IBM Optim

Optim Common Elements Manual

Version 7 Release 3

IBM Optim

Optim Common Elements Manual

Version 7 Release 3

Note
Before using this information and the product it supports, read the information in "Notices" on page 487.

Version 7 Release 3 (September 2010)

This edition applies to version 7, release 3 of IBM Optim and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

About this Guide vii

Chapter 1. Introduction 1
Common Elements and Utilities 2
Storage Options 3

Chapter 2. Main Window, Menus, and Dialogs 5
Main Window and Menus 5
Menus 6
Customize the Toolbar 13
Using a Dialog 15
Editor and Dialog Menus 16
Parts of a Grid 18
Change a Grid View 19
Using Find and Replace 23
Using the Open Dialog 25
Open from the Main Window 25
Open from an Editor or Dialog 26
Object Tree 27
Use a Pattern 27
Select and Open an Object 29
Delete an Object using the Shortcut Menu . . . 29
Optim Security 30
Object Security 30
Save an Optim Object 31
Delete an Optim Object 32
Email Notification 32
Cascading Delete/Update Confirmation Dialog . . 34
Retained Process Reports Dialog 35

Chapter 3. Optim Directory 37
Open the Optim Directory Dialog 38
Using the Optim Directory Editor 39
General Tab 40
Connection Tab 42
Server Tab 43

Chapter 4. Access Definitions 45
Open the Access Definition Editor 46
Create a New Access Definition 46
Select an Access Definition to Edit 47
Using the Editor 47
Tables Tab 49
Relationships Tab 57
Variables Tab 71
Point and Shoot Tab 73
Group Tab 74
Table Specifications 76
Columns Tab 77
Selection Criteria Tab 79
SQL Tab 81
Sort Tab 83
Archive Actions Tab 85
Archive Index Tab 100
File Attachments Tab 101
Access Definition Tools Menu 103
Indented Table Display 104
Relationship Index Analysis 106
Show Steps 108
Edit Point and Shoot List 109
Database Changes 118

Chapter 5. Column Maps 121
Open the Column Map Editor 122
Create a Column Map 122
New Column Map Dialog 123
Select a Column Map to Edit 125
Respecify Column Map Tables Dialog 126
Using the Editor 127
List Columns According to Status 129
Specify Column Values in a Column Map . . . 130
Data Privacy Functions 140
Compatibility Rules for Column Maps 168
Comparison Compatibility Rules 169
Mapping Column Data 169

Chapter 6. Column Map Procedures 175
Open the Column Map Procedure Editor 176
Create a Column Map Procedure 176
Select a Column Map Procedure to Edit . . . 177
Sample Column Map Procedures 177
Using the Editor 177
Export a Column Map Procedure 185
Import a Column Map Procedure 186
Save a Column Map Procedure 187
Delete a Column Map Procedure 187

Chapter 7. DB Aliases 189
Open the DB Alias Editor 189
From the Main Window 189
From the DB Alias Editor 190
Using the Editor 190
General Tab 191
Connection Tab 194
Server Tab 195
Table Defaults 197
Index Defaults 198
Synonym Defaults 199
Alias Defaults 200
Trigger Defaults 201

Chapter 8. Primary Keys. 203
Open the Primary Key Editor 204
Create a Primary Key 204
Select a Table for the Primary Key Definition Dialog 204
Select a Primary Key to Edit 206
Using the Editor 207

Define a Primary Key Based on a Unique Index 208
Manage Generic Primary Keys 209

Chapter 9. Relationships 213
Open the Relationship Editor 214
Create a Relationship 214
Select Relationship's Parent Table Dialog . . . 215
Select a Relationship to Edit 216
Using the Editor 217
Specify Column Values in a Relationship . . . 219
Manage Generic Relationships 224
Compatibility Rules for Relationships 226

Chapter 10. Table Maps 229
Open the Table Map Editor 229
Create a Table Map 229
New Table Map Dialog 231
Select a Table Map to Edit 232
Using the Editor 233
Tables Tab 236
Defaults Tab 237
Functions Tab 237
Packages Tab 238
Procedures Tab 238
Rules Tab 239
Sequences Tab 239
User Defined Types Tab 240
Views Tab 241
Assemblies Tab 241
Partition Functions Tab 242
Partition Schemes Tab 242
Specify Destination Objects in a Table Map . . 243
Specify Source 2 Tables in a Table Map 244
Merge Table Maps 245
Merge a Table Map Dialog 245
Create or Modify Archive Actions in a Table Map 247
Specify Column Maps in a Table Map 247
Save a Table Map 252

Chapter 11. Restart/Retry 253
Open Restart/Retry 254
Using Restart/Retry 255
Review a Pending Process 255
Restart or Retry a Pending Process 256

Chapter 12. Calendars 259
Open the Calendar Editor 259
Create a New Calendar 259
Select a Calendar to Edit 261
Using the Editor 261
General Tab 261
Dates Tab 263
Rules Tab 267
Sample Calendar Dates 268
Sample Calendar Rules 270

Chapter 13. Currency 275
Using the Currency Editor 276
General Tab 276
Rates Tab 278
Types Tab 280
Sample Rate Table 282

Chapter 14. Schedule 283
Schedule a Process Request 283
Job Details Dialog 284
General 284
Repeats 285
Steps 287
Scheduling Editor 288
Overrides Dialog 290
Scheduling Monitor 291
Jobs 291
Active 292
Completed 294

Chapter 15. Browse 299
Open a File to Browse 299
From the Main Menu 299
From a Request Editor or list of files 300
From Windows Explorer 300
Tables Tab 301
Information Tab 302
Display Table Data 303
Display Options 307
LOB Columns 311
Display Multiple Tables 314
Select Join Table Dialog 315
Specify a Relationship for Joining 316
Open Relationship Editor 317
Stack Tables 317
Unjoin Tables 317
Printing Options 317
Save File Information as an Output File 318

Chapter 16. Export and Import 319
Export 320
Export Using the Interface 320
Export Dialog 320
Command Line Export 325
Import 328
Import for UNIX 328
Import Using the Interface 329
Command Line Import 337
Output/Input File Format 339
Definition Statement Syntax 340
Access Definitions 340
Archive File Collections 349
Calendars 350
Column Maps 353
Column Map Procedures 354
Currency 355
DB Aliases 356
Edit Definition 359
Primary Keys 361
Relationships 362
Storage Profiles 362
Table Maps 367
Archive Requests 369

Compare Requests 374
Convert Requests 376
Delete Requests 380
Extract Requests 381
Insert Requests 385
Load Requests 391
Report Requests 403
Restore Requests 407

Chapter 17. Create 411
Open the Create Utility 412
Using Table Maps 413
Using the Create Utility 413
Managing the Display 416
Convert DBMS and Optim Primary Keys and Relationships 417
Create Objects 417
Drop Objects 418
Browse SQL for Objects 418
Defaults Dialog 418
Process the SQL 423
Review and Edit SQL 423
Browse Output Dialog 425

Chapter 18. Personal Options 427
Open the Personal Options Editor 427
Using the Configuration Program 427
Within Optim 427
Using the Editor 428
General Tab 429
Confirm Tab 430
Display Tab 432
Errors Tab 435
Scheduling Tab 436
Load Tab 438
Create Tab 440
Logon Tab 448
Server Tab 449
Edit Tab 452
Browse Tab 456
Archive Tab 458
Removable Media Tab 459
Actions Tab 461
Printer Tab 463
Database Tab 465
Notify Tab 468

Appendix A. Enhanced File Names 471
Using Enhanced File Names 471

Appendix B. Exit Routines for Column Maps 473
Writing Exit Routines 474
Standard Exit Routine 475
Source Format Exit 476
Destination Format Exit 478
Sample Exit Routines 481

Appendix C. Row List Files 483
Using the List 485

Notices 487
Trademarks 489

Glossary 491

Index 507

About this Guide

The IBM® Optim™ solution includes the components Archive, Compare, Edit, and Move. This user manual provides information on how to use the elements common to all the Optim components. This release runs in the Microsoft Windows environment, or in the Sun Solaris, Hewlett-Packard HP-UX, IBM AIX®, or Red Hat Linux environments supplemented with a Windows workstation. Optim supports the IBM DB2®, Oracle, Sybase Adaptive Server Enterprise (ASE), Microsoft SQL Server, and IBM Informix® database management systems. Additional database management systems may be supported in future releases.

Organization

The information in this guide is organized into the following chapters:

Chapter 1, "Introduction," on page 1
General information about Optim and an overview of the elements common to its components.

Chapter 2, "Main Window, Menus, and Dialogs," on page 5
Dialogs and features provided with Optim.

Chapter 3, "Optim Directory," on page 37
General information about the Optim Directory and procedures needed to access the Directory. This section also discusses how to connect and disconnect from a specific database.

Definitions

Chapter 4, "Access Definitions," on page 45
How to create or edit specifications for data to be processed by Optim. An Access Definition lists tables, prescribes relationship usage, and includes criteria and other parameters.

Chapter 5, "Column Maps," on page 121
How to create or edit specifications needed to match or exclude columns from processing. A Column Map matches columns in tables that have different Creator IDs or different names and can be used to modify data, age dates, or convert currency.

Chapter 6, "Column Map Procedures," on page 175
How to create or edit Column Map Procedures. A Column Map Procedure can generate values that could not otherwise be defined for the destination column.

Chapter 7, "DB Aliases," on page 189
How to browse and update parameters that Optim uses to communicate with a database. A DB Alias is a user-defined name associated with a specific database. The DB Alias serves as a prefix in the fully-qualified names of database tables, primary keys, and relationships.

Chapter 8, "Primary Keys," on page 203
How to create or edit primary keys that supplement those defined to the database. Primary keys are essential for extracting data and performing other Optim processing.

Chapter 9, "Relationships," on page 213
How to create or edit relationships that supplement relationships defined to the database. Optim allows you to define relationships that replicate implicit, or application-managed, relationships.

Chapter 10, "Table Maps," on page 229
How to create or edit specifications needed to match or exclude tables from processing. A Table Map directs the placement of data in a Convert, Create, Insert, or Restore Process and can be used to exclude one or more tables from a Compare, Convert, Create, Insert, or Restore Process.

Utilities

Chapter 11, "Restart/Retry," on page 253
How to repeat a process that did not successfully execute. You can restart a process that ends abnormally, either by user request or because of system limitations. You can retry a process that ends because the discard row limit was reached.

Chapter 12, "Calendars," on page 259
How to create or edit calendars, dates, and rules for date aging and scheduling. During an Insert Process, you can apply several techniques for date aging that can assist in processing data and testing applications.

Chapter 13, "Currency," on page 275
How to create or edit Currency Tables used to convert from one currency type to another.

Chapter 14, "Schedule," on page 283
How to schedule requests for one-time or recurring processing.

Chapter 15, "Browse," on page 299
How to browse Extract, Control, Archive, and Compare Files.

Chapter 16, "Export and Import," on page 319
How to migrate Optim objects from one Optim Directory to another.

Chapter 17, "Create," on page 411
How to create tables and their related objects using the object definitions stored in an Extract or Archive File.

Options

Chapter 18, "Personal Options," on page 427
How to customize environmental factors such as the desired font for messages, occurrence of confirmation dialogs, and maximum entries for Menu Lists and boxes. You can also specify default directories and other operational factors.

Appendices

Appendix A, "Enhanced File Names," on page 471
How to use macros to dynamically create unique file names for any file that is generated by Optim.

Appendix B, "Exit Routines for Column Maps," on page 473
The use of User exits when defining Column Maps. User exits allow you to set values for columns in the destination table that could not be established otherwise.

Appendix C, "Row List Files," on page 483
How to create a row list file without using Optim and how to define this file for use when extracting data.

"Glossary" on page 491
Definitions of terms used throughout this manual.

Chapter 1. Introduction

The IBM Optim solution manages enterprise data throughout every stage of the information lifecycle. Optim enables your company to assess, classify, subset, archive, store, and access enterprise application data. Using the archiving features in Optim, you can

- Isolate historical data from current activity and safely remove it to a secure archive.
- Access archived historical data easily, using familiar tools and interfaces.
- Restore archived data to its original business context when it requires additional processing.

The Optim test data management capabilities provide an efficient alternative to database cloning, allowing you to create development and testing environments that are sized appropriately.

Optim helps you achieve these benefits with the following components: Archive, Move, Edit, and Compare.

- Archive allows you to identify and archive sets of relationally intact data before removing selected data from your database. Archived data is indexed and stored. You can browse, search, and restore selected subsets of archived data.
- Move extracts and migrates sets of relationally intact data. Using Move, you can create test databases that are referentially intact subsets of a production database. You can also extract sets of related data and transform it as you migrate the data to the test database.
- Edit is used to browse and edit sets of relationally intact data in database tables. Using Edit, you can edit data, review logical application paths, and browse data to ensure that application test results are as expected. Edit supports your rapid development of applications, allows you to analyze the structure of your database, and facilitates your browsing of precisely defined segments of relational data.
- Compare facilitates comparisons of sets of relationally intact data. The Compare component compares data results from application tests to the original data and highlights all differences. Compare allows you to rapidly analyze the effects of your application software or modifications to it.

The Scheduler allows you to schedule Optim processes to run periodically, in sequence, or both. This feature allows you to execute selected processes at a convenient time or on a regular basis.

The Optim Server allows you to define tasks on workstations and direct any resource-intensive data processing functions to a machine more suited to the task. This machine can be one on which the database is located, eliminating associated network traffic, or a machine dedicated to the Server function.

The Configuration program provides a way to configure workstations (including machines hosting the Server) to use Optim. Using the Configuration program, you can also maintain and customize the Optim environment.

Additional features allow you to run processes from the command line, restart a process that terminates abnormally, retry a process for which all rows are not processed successfully, customize the handling of dates for aging data and scheduling processes, convert currency, browse files generated by Optim, and create database objects from definitions in generated files.

Optim is easy to use, simple in concept, yet powerful in supporting complex database structures. Intuitive dialogs simplify data entry tasks and provide options for processing relationally intact sets of data. Intelligent window handling technology allows you to display multiple dialogs, pop-up windows, context-sensitive online help, and tutorials.

Depending upon your licensing agreement, you may use one or more components of Optim.

Common Elements and Utilities

Features common to all or most of the Optim components are discussed in this book. To carry out its functions, Optim relies upon user-defined objects that supplement objects defined to the database (for example, tables, primary keys, relationships, stored procedures). These user-defined objects (collectively, Optim objects) are stored in the Optim Directory.

Optim Directory and Common Optim Objects

The Optim Directory is a set of database tables used by Optim to track processing status and store objects needed for processing. You must use the Configuration program to create or configure the Optim Directory tables and stored procedures needed to access the Directory.

Objects in the Optim Directory that are common to the Optim components include:

- **Access Definitions**. An Access Definition identifies a set of related data to be processed. It references database tables and their relationships, and provides criteria to select specific rows within tables.

  An Access Definition is required for an Archive or Extract Process and is sometimes used for a Compare, Edit, or Restore Process.
- **Column Maps**. A Column Map provides specifications needed to match columns between two tables referenced in a Table Map. Also, a Column Map can be used to transform data, age dates in tables, and exclude one or more columns from processing.

  A Column Map is used for a single table Compare Process and can be referenced in a Table Map used for a Compare, Convert, Insert, Load, or Restore Process.
- **Column Map Procedures**. In a Convert, Insert, Load, or Restore Process, a Column Map Procedure facilitates data transformations that are beyond the scope of native Column Map functions.
- **DB Aliases**. A DB Alias provides parameters needed to connect to a specific database. A DB Alias name is used as a high-order qualifier for an object or table name, providing information Optim requires to access the appropriate database.

  A DB Alias is needed any time Optim references a database object; for example, to identify an Optim primary key, Optim relationship, or a database table referenced in an Access Definition, Column Map, or Table Map.
- **Primary Keys**. Values in primary key columns uniquely identify each row in a database table.

  A primary key can be used to create an Optim relationship, and is required for a table that is changed by a Delete, Insert, or Restore Process or a table that is visited more than once in an Extract or Archive Process. A primary key is also required to enable the row selection (Point and Shoot) feature for an Access Definition or an Archive or Extract Process.

  Optim uses primary keys defined to the database, if present. You can define Optim primary keys to supplement those in the database.
- **Relationships**. A relationship is a defined connection between the rows of two tables that determines the parent or child rows to be processed and the order in which they are processed.

  Optim uses relationships to determine the data to be retrieved from related tables and relies upon relationships defined to the database, if available. However, you can also define relationships to supplement those in the database. Generally, a relationship is needed in a process that uses an Access Definition.
- **Table Maps**. A Table Map identifies and matches two tables or sets of tables in a process and can be used to exclude one or more tables from processing.

  A Table Map is required for a Compare, Convert, Insert, Load, or Restore Process.

Common Utilities

Processes are discussed in the appropriate user manuals. Utilities, except those specific to Archive (e.g., Archive Directory Maintenance and File Registry, and the Storage Profile Utility), are discussed in this book. The utilities discussed here are:

- **Restart/Retry**. Use the Restart/Retry Utility to restart a process that terminated abnormally or to retry a process for which all rows are not successfully processed.

  You can use this utility to restart or retry a Delete, Insert, or Insert/Update Process.
- **Calendar**. Use the Calendar Utility to customize handling of dates for aging data in a Convert or Insert Process and for scheduling process requests.
- **Currency**. Use the Currency Utility to customize currency conversion parameters for Convert or Insert Processes.
- **Schedule**. Use the Schedule Utility to schedule processes and monitor processing.
- **Browse**. Use the Browse Utility to review the contents of an Archive, Compare, Extract, or Control File.
- **Export/Import**. Use the Export/Import Utilities to copy Optim objects from one Optim Directory to another.
- **Create**. Use the Create Utility to create database objects from definitions in an Archive or Extract File.

Options

Options are used to maintain the Optim environment. Generally, Product Options parameters enforce site and system requirements, while Personal Options allow you to customize Optim for each workstation.

Security options allow you to establish as many as three levels of security for Optim. Functional security allows you to control user access to the interface for functions provided by Optim; object security allows you to control access to specific objects in the Optim Directory, and Archive File security allows you to control access to data in Archive Files. All security options are documented in the *Installation and Configuration Guide*.

Common Optim objects and utilities and Personal Options are discussed in ensuing chapters. (Product Options are discussed in the *Installation and Configuration Guide*.)

Storage Options

Optim supports NetApp SnapLock and Symantec VERITAS Enterprise Vault for all generated file types.

Optim supports the following backup devices for Archive Files: EMC NetWorker, IBM Tivoli® Storage Manager, and EMC Centera. You can use Optim to set Centera and WORM device minimum retention periods for Archive Files.

Chapter 2. Main Window, Menus, and Dialogs

The graphical user interface for Optim is designed for a Windows operating environment. You should be familiar with many features that are standard for Windows applications.

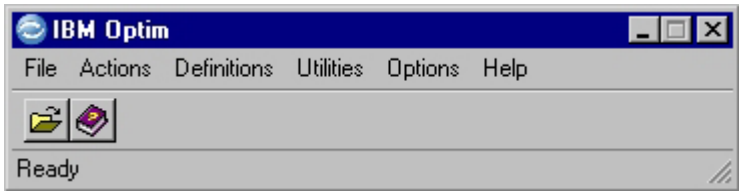
Contents

This section describes:

- Menu commands and other functions available from the main window.
- Features common to the editors and dialogs.

Main Window and Menus

When you start Optim, the main window is displayed. The main window includes a menu bar, toolbar, and status bar:



Menu bar
Processing and other menus.

Toolbar
Buttons used to choose menu commands. You can customize the toolbar for the main window, the editors and other dialogs.

To open the Customize Toolbar dialog, double-click the toolbar area (below the menu bar) or select **Customize Toolbar** from the **Options** menu.

Note: To view or hide the toolbar, select **Toolbar** from the **Options** menu. When the Toolbar command is checked, the toolbar appears below the menu bar.

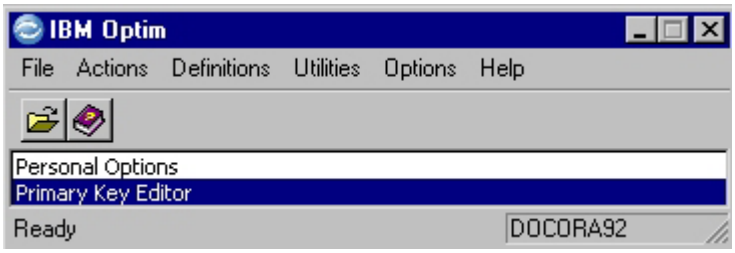
Status bar
The status bar at the bottom of the main window, the editors, and other dialogs, displays messages about a specific command or the current action.

You can view status bar messages when you position the mouse cursor over a toolbar button or a menu command, and when you select a menu command.

Note: To view or hide the status bar, select **Status Bar** from the **Options** menu. When the Status Bar command is checked, the status bar is displayed at the bottom of the main window.

Active Dialogs

As you select menu commands and open different dialogs, the main window expands to list the active dialogs:



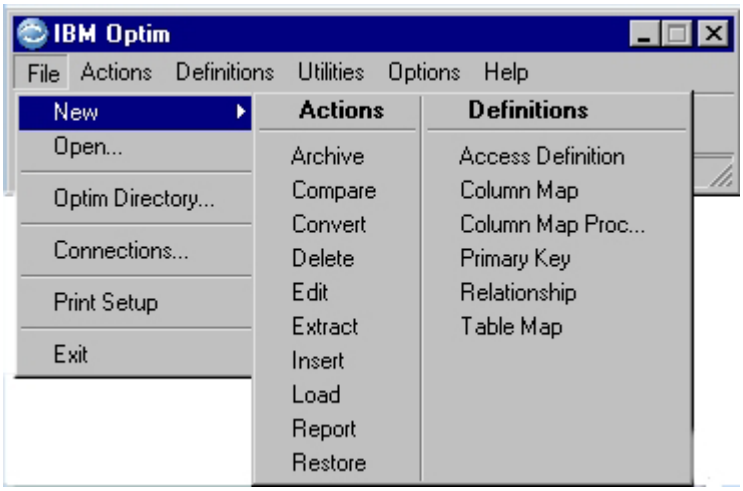
You can recall any listed editor or dialog by double-clicking the name of the item in the list.

Menus

To use a facility, select a command from the menu bar in the main window. Each menu command is described in the following paragraphs.

File Menu

The **File** menu in the main window lists standard commands for opening a new or existing object (action request or definition). It also lists unique commands for reviewing Optim Directory and database connections. Select any of the following commands:



Note: The **New** submenu lists Actions and Definitions for all products. Functionality is limited according to the site-specific license.

New Open an action or definition editor. The **New** command displays the **Actions** and **Definitions** submenus. Choose from either submenu to display an editor in which to create a new process request (action) or Optim definition.

Open Open a dialog and select an editor to display a specific process request or Optim definition. For information about the Open dialog, see "Using the Open Dialog" on page 25.

Note: To display the last action or definition that you edited, select the appropriate editor from the **Actions** or **Definitions** menu in the main window.

Optim Directory
Open the Optim Directory dialog, listing the Optim Directories to which you have access. The Optim Directory tables store Optim objects (action requests and definitions) and information needed to access a database. For details, see "Open the Optim Directory Dialog" on page 38.

Connections
Open the Connections dialog and list the DB Aliases to which you have active database connections.

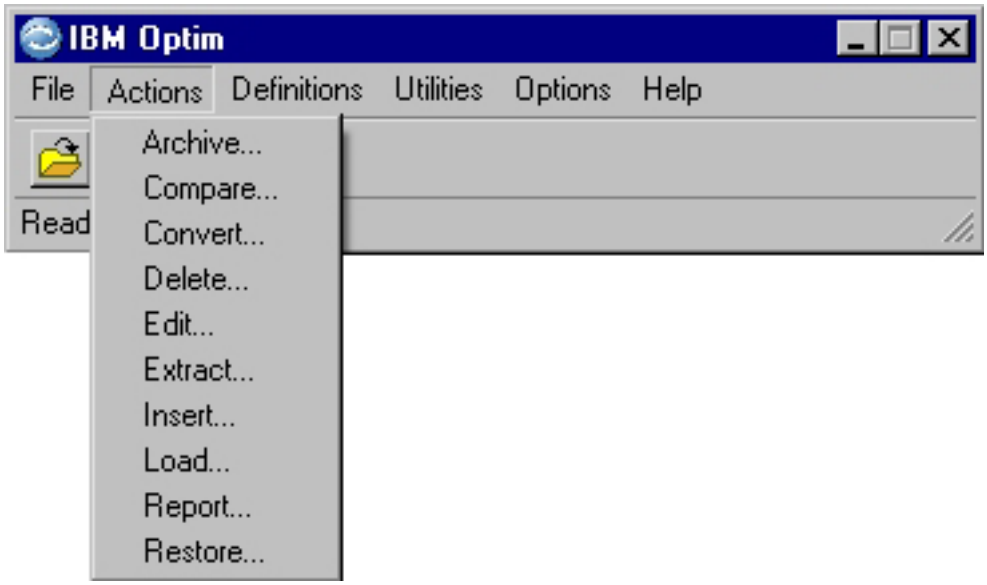
Print Setup
Open the standard Windows Print Setup dialog.

Exit Quit Optim.

Note: File menu commands available from the editors and dialogs are different from those in the main window. These differences are explained, as appropriate.

Actions Menu

The **Actions** menu in the main window lists commands for opening the editors used to create and edit process requests.



Note: The Actions menu lists Actions for all Optim components. Functionality may be limited according to the site⌂specific license.

Choose from the following commands:

Archive
Open the Archive Request Editor. Use this command to edit specifications needed to archive a set of related database rows and object definitions in an Archive File. You can use data in an Archive File as input for a Convert or Restore Process, and both data and object definitions as input for a Create Process. See the *Archive User Manual* for details.

Compare
Open the Compare Request Editor. Use this command to edit specifications needed to compare data. See the *Compare User Manual* for details.

Convert
Open the Convert Request Editor. Use this command to edit specifications needed to transform the contents of an Extract or Archive File. The transformed data can be stored in the same or another file. See the *Move User Manual* for details.

Delete Open the Delete Request Editor. Use this command to edit specifications needed to delete data from the database. Delete uses an Extract or Archive File as the source to perform a relational delete. See the *Archive User Manual* for details.

Edit Open the Table Editor. Use this command to edit or browse data in a specified table or set of tables. See the *Edit User Manual* for details.

Extract
Open the Extract Request Editor. Use this command to edit specifications needed to extract a set of related database rows and store the data in an external file, an Extract File. You use an Extract File as input for a Convert, Delete, Load, Insert, or Create Process. See the *Move User Manual* for details.

Insert Open the Insert Request Editor. Use this command to edit specifications needed to insert data from an Extract or Archive File into database tables. If the destination tables do not exist, you are given the opportunity to create them from object definitions in the file. Also, Insert allows you to decide whether existing rows are updated. (New rows are always inserted.) See the *Archive User Manual* or *Move User Manual* for details.

Load Open the Load Request Editor. Use this command to edit specifications needed to prepare an Extract or Archive File for a specific DBMS load utility and to invoke the utility. See the *Archive User Manual* or *Move User Manual* for details.

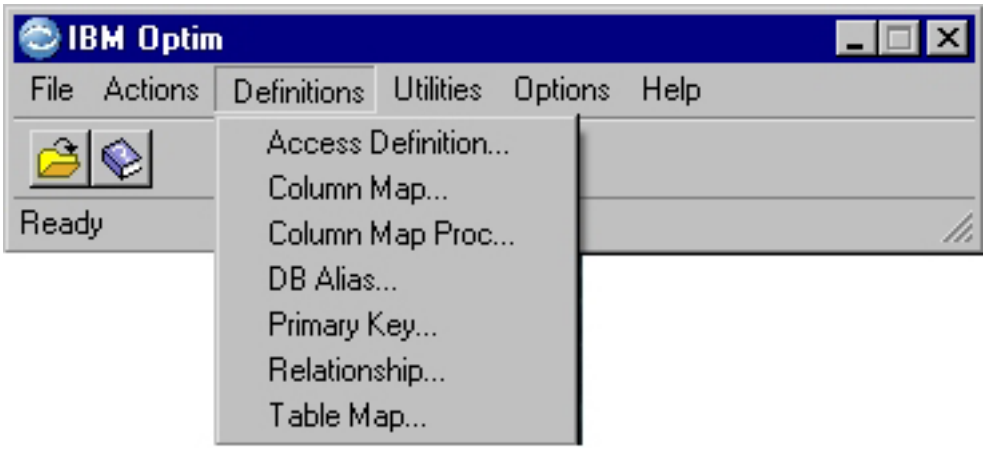
Report
Open the Report Request Editor. Use this command to edit specifications needed to prepare a report on the contents of an Archive or Compare File. See the *Archive User Manual* or *Compare User Manual* for details.

Restore
Open the Restore Request Editor. Use this command to edit specifications needed to restore archived data. See the *Archive User Manual* for details.

Note: Each command opens a process request in the selected editor. The **Action/Definition Menu Behavior** setting in Personal Options determines the default for opening the selected editor on the last edited request or to create a new request. See "Display Tab" on page 432 for information about this setting.

Definitions Menu

The **Definitions** menu in the main window lists commands to open the editors used to define (or edit) and save Optim objects. The objects are stored in the Optim Directory and are available to any authorized users.



Choose any of the following commands:

Access Definition
Open the Access Definition Editor. An **Access Definition** defines a set of related data to be

archived or extracted. An Access Definition references the tables and designates the relationships and traversal path as well as criteria to select the related data. For details, see Chapter 4, “Access Definitions,” on page 45.

Column Map
Open the Column Map Editor. A **Column Map** provides specifications for matching source columns to destination columns when performing a Compare, Convert, Insert, Load, or Restore Process. For details, see Chapter 5, “Column Maps,” on page 121.

Column Map Proc
Open the Column Map Proc Editor. Use the Column Map Proc Editorto edit or create Column Map Procedures. Column Map Procedures can be used to generate values that could not otherwise be defined for a destination column, including special processing and data manipulation. For details, see Chapter 6, “Column Map Procedures,” on page 175.

DB Alias
Open the DB Alias Editor. A **DB Alias** is a set of specifications that enables Optim to identify, locate, and access a particular database. (The DB Alias serves as a qualifier for the names of tables that are referenced, defined, or accessed.) For details, see Chapter 7, “DB Aliases,” on page 189.

Primary Key
Open the Primary Key Editor. A **Primary Key** is a column or columns containing unique values used to identify each row in a table.

Although you can view primary keys defined to the database, you can edit only those stored in the Optim Directory. For details, see Chapter 8, “Primary Keys,” on page 203.

Relationship
Open the Relationship Editor. A **Relationship** is a defined connection between the rows of two tables. This connection is generally determined by values in selected columns from a parent table that correspond to values in the child table. Optim permits more flexible “extended relationships” (Optim relationships), that need not conform to database conventions. Using extended relationships, you can relate parent and child columns without requiring a primary key, use concatenation and substring functions, and emulate data-driven relationships.

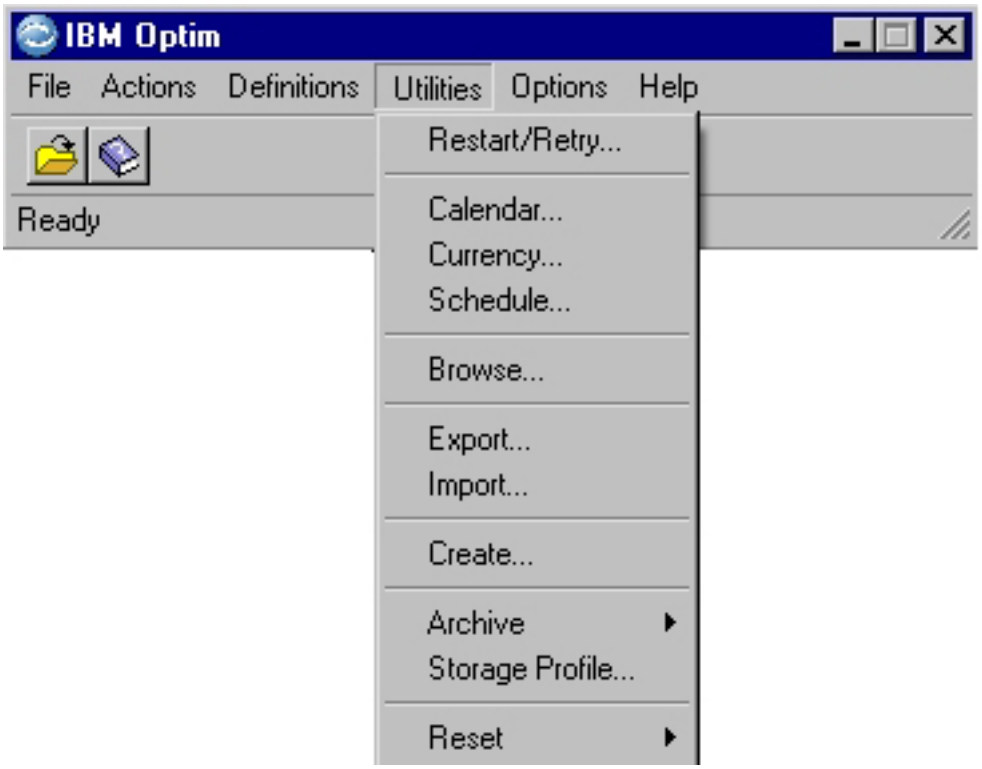
Although you can view relationships defined to the database, you can edit only those stored in the Optim Directory. For details, see Chapter 9, “Relationships,” on page 213.

Table Map
Open the Table Map Editor. A **Table Map** provides specifications for matching source and destination tables when performing a Compare, Convert, Insert, Load, or Restore Process or using the Create Utility. For details, see Chapter 10, “Table Maps,” on page 229.

Note: Each command opens a request in the selected editor. The **Action/Definition Menu Behavior** setting in Personal Options determines the default for opening the selected editor with either the last edited request or a new request. See “Display Tab” on page 432 for information about this setting.

Utilities Menu

The **Utilities** menu in the main window lists commands to open the utilities dialogs.



Select any of the following commands:

Restart/Retry
Open the Restart/Retrydialog. Use **Restart/Retry** to display a list of processes that terminated abnormally or did not process all rows successfully. Select from this list to resume (restart) a process that terminated abnormally or to reprocess (retry) rows that did not process successfully. (Restart/Retry is available for Insert and Delete Processes.) For details, see Chapter 11, “Restart/Retry,” on page 253.

Calendar
Open the Calendar Editor. Use **Calendar** to define the calendar year, dates, and business rules that comprise a Calendar. Calendars are used for date aging and process scheduling. For details, see Chapter 12, “Calendars,” on page 259.

Currency
Open the Currency Editor. Use the Currency Editor to create, browse, edit, and delete Currency Tables. Currency Tables are used to aid in the conversion of monetary amounts between different currencies. For details, see Chapter 13, “Currency,” on page 275.

Schedule
Open the Scheduling Editor. Use **Schedule** to review a list of scheduled jobs, edit scheduling specifications for individual jobs, and schedule additional jobs. For details, see Chapter 14, “Schedule,” on page 283.

Browse
Open the Browsedialog. Use the Browsedialog to review the contents of an Archive, Compare, Control, or Extract File. For details, see Chapter 15, “Browse,” on page 299.

Export Open the Exportdialog. Use **Export** to copy one or more Optim object definitions to an external file. These objects include the saved definitions and action requests stored in the Optim Directory.

The external file is used by the Import Utility to add the object definitions to another Optim Directory. Together, the Export and Import Utilities can migrate Optim object definitions between Optim Directories. For details, see Chapter 16, “Export and Import,” on page 319.

Import
Open the Importdialog. Use **Import** to insert the Optim object definitions from an external file into the specified Optim Directory. For details, see “Import” on page 328.

Create Open the Create Options dialog. Use **Create** to create database objects from the definitions in an Archive or Extract File. These objects can include: tables, primary keys, relationships, and indexes. For details, see Chapter 17, “Create,” on page 411.

Archive
Display a submenu with the **Directory Maintenance**, **Index Maintenance**, and **Register Files** commands. Use **Archive** to register archive files for the Archive Directory, and to perform Directory and index maintenance. For details, see the *Archive User Manual* .

Storage Profile
Open the Storage Profile Definition Editor. Use this editor to create, browse, edit, and delete Storage Profile Definitions. Storage Profile Definitions can be used to define parameters for saving Archive Files to fixed media (e.g., a local hard drive or network drive) and secondary media (e.g., zip drive, backup device, etc.).

For details, see the *Archive User Manual* .

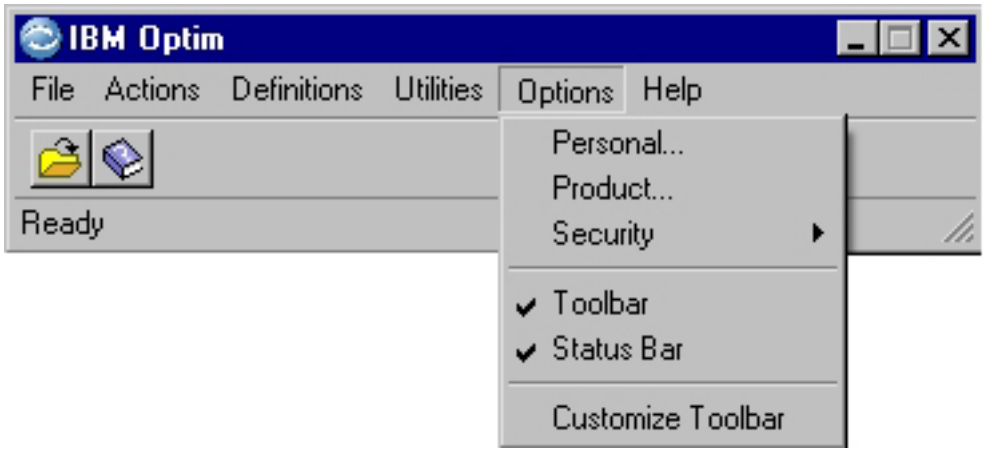
Reset Display the **Object Cache** and **Cancelled Logons** commands.

Select **Object Cache** to reset the cache and direct Optim to go to the database to obtain any changes made to database objects or configuration parameters. (Optim caches data from the database catalogs to improve performance.)

Select **Cancelled Logons** to reset all DB Alias Logon Connection dialogs that you cancelled. (When you cancel a DB Alias Logon Connection dialog, you cannot gain access to it again from the same editor unless you reset the cancelled logon.)

Options Menu

The **Options** menu in the main window lists commands for setting Personal and Product Options. These options allow you to tailor system features to your needs. Select any of the following commands:



Personal
Open the Personal Options dialog. Use Personal Options to select settings to customize your system. For details, see Chapter 18, “Personal Options,” on page 427.

Product
Open the Product Options dialog. For security, you are prompted for a password before the

dialog opens. Enter the password to access the Product Options. Product Options customize your system. For details, see the *Installation and Configuration Guide* .

Security
Display a submenu with the **Access Control Domains**, **File Access Definitions**, and **Optim Object Template ACL** commands used with Optim Security, which secures Archive Files, Optim functions, and objects in an Optim Directory.

For details about Optim Security, see the *Installation and Configuration Guide* .

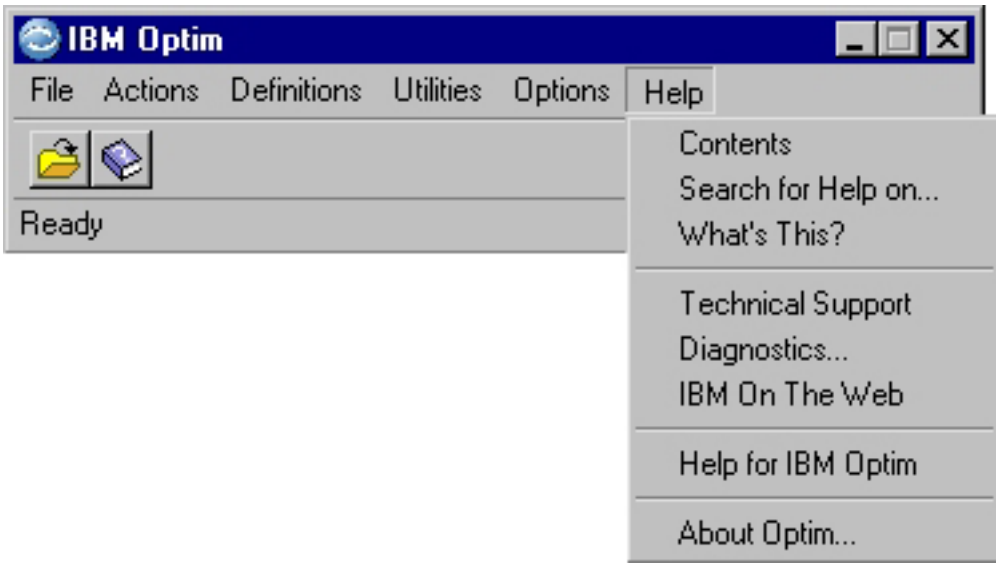
Toolbar
Toggle the toolbar display. You can customize the toolbar in the main window, editors, and other dialogs. For details, see "Customize the Toolbar" on page 13.

Status Toggle the status bar display. When active, the Status bar is at the bottom of the active window or dialog and displays messages about a specific command or the current action.

Customize Toolbar
Open the Customize Toolbardialog. Use **Customize Toolbar** to select buttons to include in the main window toolbar. For details, see "Customize the Toolbar" on page 13.

Help Menu

The **Help** menu in the main window allows you to access online help regarding Optim features. You can select any of the following commands:



Contents
Open the Help Topics dialog to the **Contents** tab. Use **Contents** to display the Help table of contents.

Search for Help on
Open the Help Topics dialog to the **Index** tab. Use **Index** to search for a specific topic.

What's This?
Display the ***What's This?*** pointer. Use this pointer to select a dialog control for context⌂sensitive help.

Technical Support
Display information needed to contact Technical Support. You can request Technical Support by phone: 1-800-IBMSERV. If it is necessary to contact Technical Support, have ready:

- Information Checklist

- Product name and build number
- Full text of all error messages
- Screen prints of internal or system errors
- Screen prints of editors or dialogs in use when the error occurred
- Description of events leading up to the error
- Location of trace files in the Temporary Work Directory (defined in Personal Options)

Diagnostics
Display the Diagnostics dialog. Use the Diagnostics dialog process list to review, save and print specific module information.

IBM On The Web
Open a connection to the World Wide Web and display the IBM home page for access to product information, news, and support. Vist the Optim web page to learn answers to frequently asked questions about Optim: http://www.optimsolution.com. The Optim web page also makes it easy to download software and learn more about the newest Optim releases.

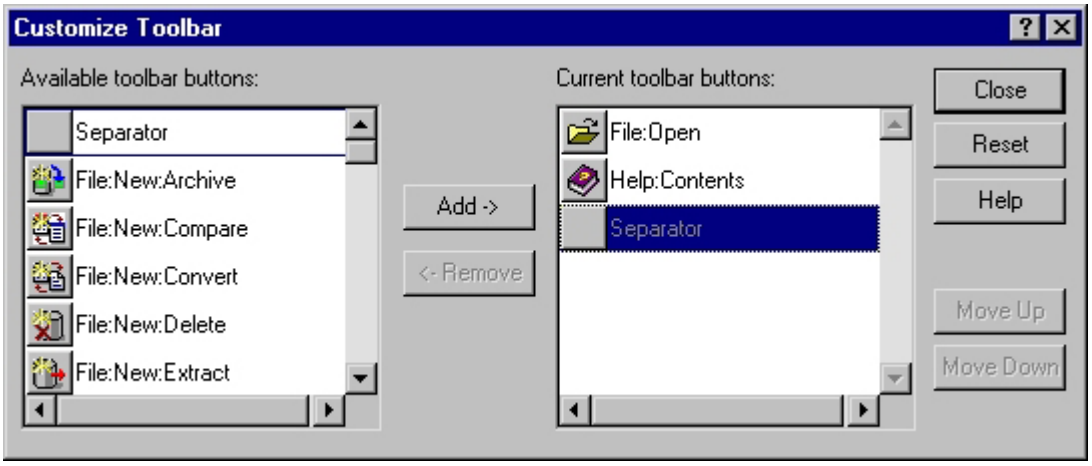
Help for IBM Optim
Open the first help topic for Optim.

About Optim
Display licensing and release information.

Customize the Toolbar

Toolbar buttons correspond to certain menu commands. You can customize toolbars in the main window, editors, and dialogs.

- To display the toolbar, select the **Toolbar** command on the **Options** menu. A check mark indicates the toolbar is active.
- To modify the toolbar, select **Customize Toolbar** on the **Options** menu (or double-click the toolbar area). The following dialog opens:



Note: Only toolbar buttons that apply to the dialog or window are listed. Changes to the toolbar for a particular dialog are profiled and apply when you reopen that dialog.

Details

Available Buttons
List of buttons available to add to the toolbar in the active window.

Current Toolbar Buttons
List of buttons on the toolbar in the order displayed.

Add Move the selected button from the **Available buttons** list to the **Current Toolbar buttons** list.

Remove
Move the selected button from the **Current Toolbar buttons** list to the **Available buttons** list.

Close Close the dialog and profile the changes for the active dialog.

Reset Restore the toolbar in the active editor or dialog to the set prior to the current editing.

Help Display online help information. For help on a specific control, select the question mark button at the top of the dialog, and select the item.

Move Up
Move the selected button up one position. On the toolbar, the button shifts left by one position. You can also drag-and-drop a button.

Move Down
Move the selected button down one position. On the toolbar, the button shifts right by one position. You can also drag-and-drop a button.

Add a Button to the Toolbar

Use the Customize Toolbar editor to add a button to a toolbar, described next:

To add a button to a toolbar:

1. In the **Options** menu, select **Toolbar** to display the toolbar.
2. Double-click the toolbar to open the Customize Toolbardialog.
3. Locate the additional button by selecting a button in the **Current Toolbar buttons** list. The added button is inserted *above* the selected button.
4. Select a button from the **Available buttons** list to add to the toolbar.
5. Click **Add**.
6. Repeat steps 3, 4, and 5 to add Separator buttons to insert spaces between buttons.
7. Click **Close** to exit the Customize Toolbardialog.

Remove a Button from the Toolbar

You can use one of the following methods to remove a button from the toolbar:

Method 1:

1. Press Shift and click the button on the toolbar. The pointer changes to:  .
2. Drag the button to the desktop.

Method 2:

1. Double-click the toolbar area to open the Customize Toolbardialog.
2. On the **Toolbar buttons** list, select the button to remove.
3. Click **Remove**.

Move a Button on the Toolbar

Use one of the following methods to move a button on the toolbar:

Method 1:

1. Press Shift and click the button on the toolbar. The pointer changes to:  .
2. Drag the button to a new position on the toolbar.

Method 2:

1. Double-click the toolbar area to open the Customize Toolbardialog.

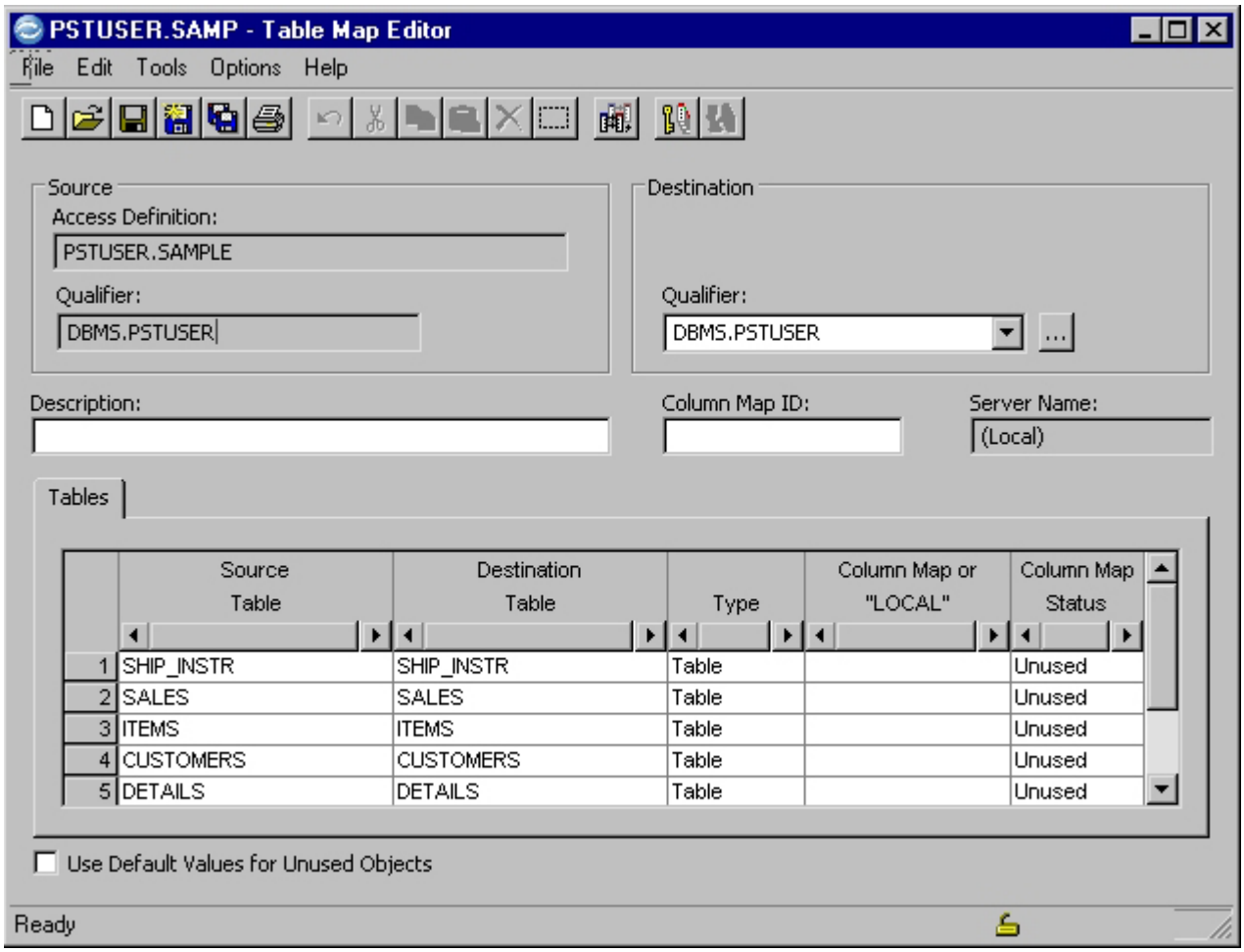
2. On the **Toolbar buttons** list, select the button you want to move.
3. Drag the button to a new position, or use **Move Up** or **Move Down**.

Toolbar Button Changes

When the Optim toolbar buttons change (that is, buttons are added to or removed from the standard set of buttons delivered with Optim), any customized toolbar affected by the changes will revert to its default buttons. In this case, you must customize the toolbar again.

Using a Dialog

The typical editor or dialog includes a title bar, menu bar, and toolbar, data entry boxes and/or a grid. Many dialogs include grids, which provide an easy way to review and enter specifications for various processing tasks. As shown in the following illustration, the grid is centered in the dialog above the message bar and status bar.



Although details vary by editor or dialog, a typical dialog includes the following:

Title bar
: Name of the object displayed in the dialog and the title of the dialog. For example, if you are using the Table Map Editor, the title bar shows the name of the Table Map you are editing.

Menu bar
: Menus for the editor or dialog.

Toolbar
Buttons that represent menu commands. You can customize the toolbar in any editor or dialog. For details, refer to "Customize the Toolbar" on page 13.

Grid Area for lists of items and attributes appropriate for the editor or dialog. You can modify grid details and display shortcut menus from grid headings or grid columns. Grids are described in more detail in "Parts of a Grid" on page 18.

Message bar
Area for information, warning, and error messages. The message bar is at the bottom of an editor or dialog directly above the status bar.

Optim displays messages when validating your entries, when you select save options, or when errors are detected. If an error is found, you can double-click the message bar to display the item in error.

Note: You can customize the font and color for each type of message, elect to hide the message bar when inactive, and personalize other message parameters by selecting **Personal** from the **Options** menu in the main window. Enter your preferences on the **Errors** tab.

Status bar
Area for messages about a specific command or the current action. The status bar is at the bottom of the main window, and each editor and dialog.

If Optim Security is initialized for the Optim Directory, the following icons are displayed:

 The object is secured by an Access Control List.

 The object is not secured by an Access Control List.

Note: Objects secured by an Access Control List are protected by Optim Security only when Object Security is enabled.

Editor and Dialog Menus

For most editors and dialogs, the menu bars are similar; however, the commands vary for particular tasks. The following is a brief summary of typical menu commands.

File Menu

The **File** menu in most editors and dialogs includes the following:

New Open an editor or dialog to create a new object (e.g., a process request or other user-defined object).

Open Display a list of objects to open one for editing.

Save Save a new object or update an existing object.

Save As
Save an object under a new name, preserving the original, and display the newly named version.

Save Copy As
Save a copy of the original object under a new name, preserve the copy, and display the original.

Set as Default
Save entries as default specifications. Select the **Set as Default** command to display the same entries the next time you open the editor to create a new object.

Delete Remove an object from the Optim Directory.

Print Open the Windows Print dialog to print a report.

Note: Use the **Printer** tab on the Personal Options dialog to set printer, font, and language preferences for printing. Use font and language settings appropriate to the character set used to create the object to help ensure that text prints correctly if the Locale settings do not match.

Redisplay Results
Redisplay the report from the previously run process or display a list of all retained reports. This option is unavailable if the current process has not been run and there are no retained reports.

For information about displaying a list of all retained reports, see "Retained Process Reports Dialog" on page 35.

History List
Display a list of recently opened requests and definitions.

Note: You can set the number of items that appear in the list by selecting **Personal** from the **Options** menu in the main window. Specify your preferences on the **Display** tab.

Close Close the active editor or dialog.

Edit Menu

The **Edit** menu in most editors and dialogs includes the following:

Undo Undo the last action or entry performed.

Cut Remove the selected value or string and save it to the clipboard.

Copy Copy the selected value or string and save it to the clipboard.

Paste Insert the contents of the clipboard at the selected place.

Clear Remove the selected value or string.

Select All
Select all items in a display.

Tools Menu

The **Tools** menu lists commands for tasks that are specific to the editor or dialog. See the discussion of the specific editor or dialog for details about the commands.

If Optim Security is initialized for the Optim Directory, the Tools menu includes the following:

Edit ACL
Open the Access Control List Editor to secure an object. Use the editor to create an Access Control List for the object or edit the existing Access Control List.

Delete ACL
Delete the Access Control List securing the object and make the object unsecured.

Note: This option is available only for objects associated with an Access Control List.

Options Menu

The **Options** menu in most editors and dialogs includes the following:

Toolbar
Display the toolbar.

Status Bar
Display the status bar.

Customize Toolbar
Open the Customize Toolbar dialog where you add or remove buttons from the toolbar in that particular editor or dialog.

Help Menu

The **Help** menu in most editors and dialogs includes the following:

• Contents	• IBM On The Web
• Search on Help	• Help for (the active editor or dialog)
• What's This	• Help for (the active tab)
• Technical Support	• About Optim
• Diagnostics	

Parts of a Grid

The grid portion of an editor or dialog displays the specifications for a definition, request, or task. In some instances (for example, when you define a relationship or create a Table Map), Optim populates these grids with information from the database or the Optim Directory. In other cases, you use the grid to enter specifications for a particular task or definition.

Note: The terms *columns* and *rows* commonly refer to database tables. To avoid confusion in this guide, the columns that make up a grid are described as *grid columns* and the rows in a grid are referred to as *grid rows*.

In most grids, you can open a shortcut menu when you right-click a grid heading or a grid column.

- The grid heading shortcut menu lists commands available to modify the grid settings.
- The grid column shortcut menu lists commands that facilitate data entry.

Details

A typical grid includes the following:

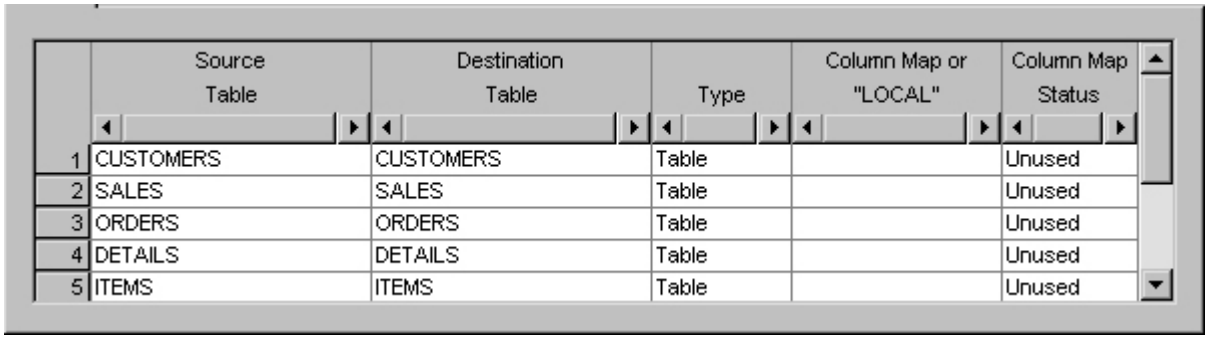
Grid Heading
Descriptive text above a grid column.

Grid Column
Vertical division of the grid, delimited by vertical lines.

Grid Row
Horizontal division within a grid, delimited by horizontal lines and often numbered.

Grid Cell
Smallest unit in a grid; the intersection of a row and a column.

The following illustrates the grid portion of the Table Map Editor:



	Source Table	Destination Table	Type	Column Map or "LOCAL"	Column Map Status
1	CUSTOMERS	CUSTOMERS	Table		Unused
2	SALES	SALES	Table		Unused
3	ORDERS	ORDERS	Table		Unused
4	DETAILS	DETAILS	Table		Unused
5	ITEMS	ITEMS	Table		Unused

Change a Grid View

At times you cannot display all grid columns and rows on your monitor. To help navigate a large grid, you can scroll, resize or move grid rows and columns manually, or use key combinations to move around a grid.

Shortcut menu commands allow you to find, replace, sort, hide, or lock grid columns and rows. Right-click a grid heading or grid column to display a shortcut menu.

Grid handling features are similar throughout Optim; any variations for specific editors and dialogs are discussed where applicable.

Adjust a Grid Manually

You can easily resize, move, and scroll grid columns and rows to view all the data in an editor or dialog. You can also delete a numbered grid row.

Resize To change the width of a grid column, position the pointer at the right boundary of the column to click and drag to the desired position.

Move You can move a grid column or row. To move a grid:

- column, select the heading and drag the column.
- row, select a numbered row and drag it to a new position. The rows renumber automatically.

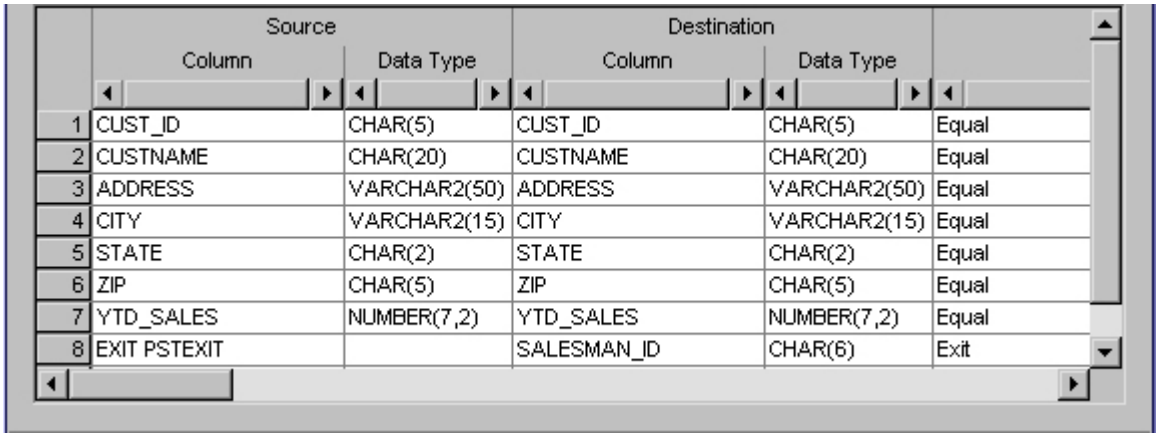
Scroll To view complete information in a grid cell, you can resize the grid column to display the entire value or use the horizontal scroll bar below the heading.

Delete To delete a grid row, click a row number and press **Delete**.

Scroll Bars

When the grid contains more information than can be viewed on your monitor, you can use the horizontal and vertical scroll bars to adjust the display left and right or to view entries on a list from top

to bottom.



	Source		Destination		
	Column	Data Type	Column	Data Type	
1	CUST_ID	CHAR(5)	CUST_ID	CHAR(5)	Equal
2	CUSTNAME	CHAR(20)	CUSTNAME	CHAR(20)	Equal
3	ADDRESS	VARCHAR2(50)	ADDRESS	VARCHAR2(50)	Equal
4	CITY	VARCHAR2(15)	CITY	VARCHAR2(15)	Equal
5	STATE	CHAR(2)	STATE	CHAR(2)	Equal
6	ZIP	CHAR(5)	ZIP	CHAR(5)	Equal
7	YTD_SALES	NUMBER(7,2)	YTD_SALES	NUMBER(7,2)	Equal
8	EXIT PSTEXIT		SALESMAN_ID	CHAR(6)	Exit

Click the scroll box to indicate the pointer position on the grid. The vertical scroll box displays the row number and the total number of rows. The horizontal scroll box displays the grid column number and the total number of grid columns.

Key Combinations

In addition to using scroll bars, you can use the following key combinations to move through grid columns and rows.

Tab Move to the next read/write cell to the right.

Shift+Tab
Move one read/write cell to the left.

Arrow Key
Move one read/write cell in the direction of the arrow key.

Ctrl+Left Arrow
Move to the beginning of a grid row.

Ctrl+Right Arrow
Move to the end of a grid row.

Ctrl+Up Arrow
Move to the top of a grid column.

Ctrl+Down Arrow
Move to the bottom of a grid column.

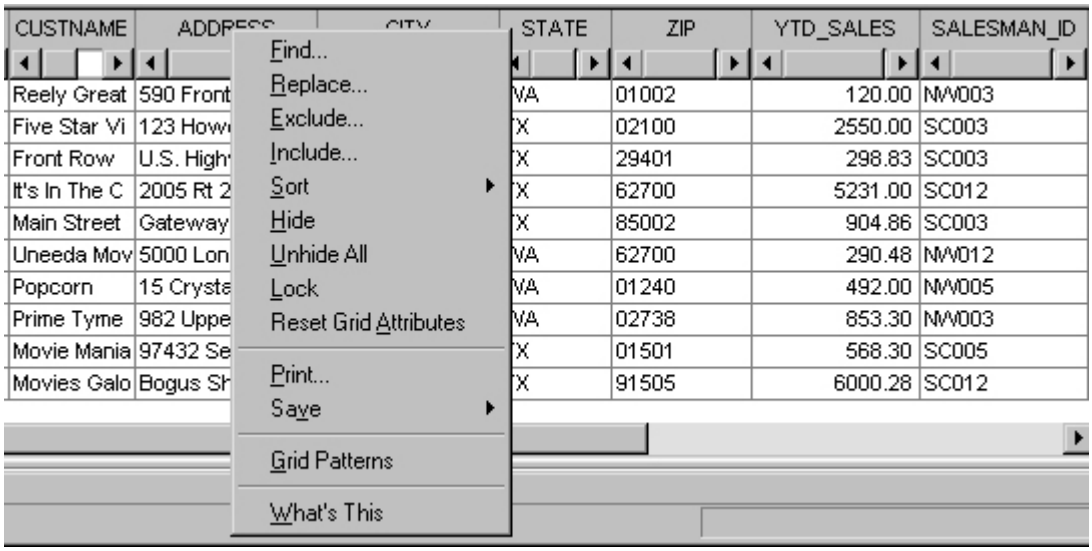
Ctrl+Home
Move to the first read/write cell in a grid.

Ctrl+End
Move to the last read/write cell in a grid.

Page Up
Scroll up through a grid.

Page Down
Scroll down through a grid.

Grid Heading Shortcut Menu



Right-click a grid column heading to display a shortcut menu. Select from the following commands to adjust the grid display:

Find Open the Find dialog where you can specify search criteria to locate a particular value in a grid column. For details on using **Find** and **Replace** commands, see "Using Find and Replace" on page 23.

Replace
Open the Replace dialog where you can enter a value or string to be replaced with a different value or string (applies only to grid columns that can be updated).

Exclude
Open the Exclude dialog where you can enter a value or string used to select rows to be excluded.

Include
Open the Include dialog where you can enter a value or string used to select rows to be included.

Sort Open the **Sort** menu where you can rearrange the grid rows according to a value or string in a grid column.

Hide Hide a grid column. To redisplay the grid column, select **Unhide All** or **Reset Grid Attributes** from the shortcut menu.

Unhide All
Display hidden grid columns.

Lock Move a grid column to the left and lock in place. When you lock more than one column, the locked columns are positioned in the order locked. The menu command is **Unlock** for previously locked columns. To unlock a column, select **Unlock**.

Note: To unlock all locked grid columns, select **Reset Grid Attributes** from the shortcut menu.

Reset Grid Attributes
Return the grid components to the original settings, display hidden grid columns, and unlock locked grid columns. Sorted data remains unchanged.

Print Open the Print dialog to print the rows in the grid.

Save Open the Save dialog to save all or selected rows.

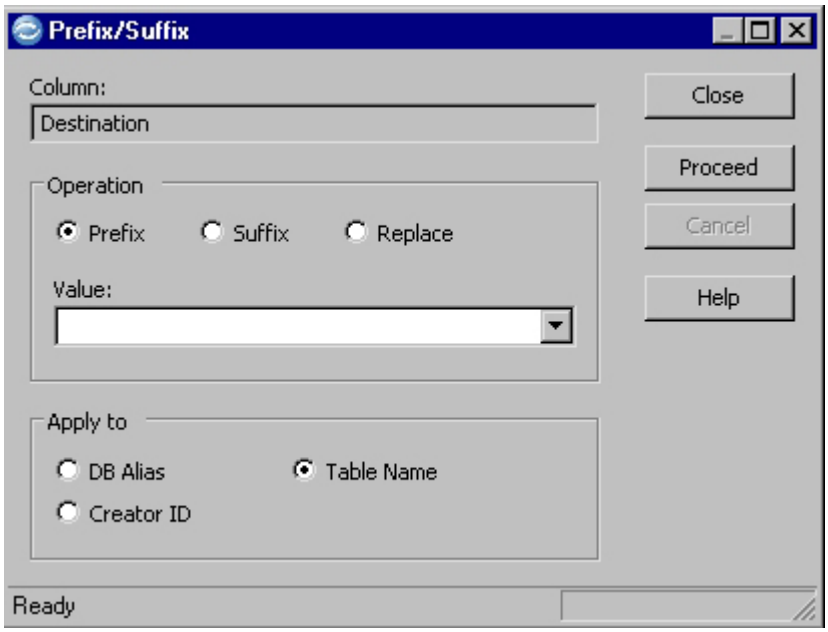
Grid Patterns
Display the Grid Patterns dialog. Refer to this dialog for a brief description of any cross-hatching patterns used in the dialog from which you right-clicked.

What's This
Display the *What's This?* pointer. Click a control in an editor or dialog with this pointer to obtain context-sensitive help.

Prefix/Suffix

The Prefix/Suffix dialog is accessible from the grid header shortcut menu in the following dialogs:

- Access Definition Editor table grid (Table grid column).
- Table Map Editor table grid (Destination Table grid column).
- Archive Request Editor Extended Object List grid (Object name grid column).
- Extract Request Editor Extended Object List grid (Object name grid column).
- Create Utility grid (Object name grid column). (You can add a prefix or suffix to the Creator-ID or object-name parts of the Index, Primary Key, Relationship, Alias or Synonym database objects. All other database objects in the Create dialog (e.g., table, default, function, package, procedure, rule, sequence, user defined type, view) are optionally prefixed or suffixed in the Table Map Editor.)



Right-click a grid column header and select Prefix/Suffix to display the Prefix/Suffix dialog.

Use the Prefix/Suffix dialog to add a prefix or suffix to any part of a database object name (i.e., table, default, function, package, procedure, rule, sequence, user defined type, or view) in the grid column beneath the header.

The following guidelines apply:

- In the Table Map Editor, the Prefix/Suffix dialog applies to the Destination name column on all tabs (Tables, Defaults, Functions, Packages, Procedures, Rules, Sequences, User Defined Types, and Views).
- The Prefix/Suffix dialog does not apply to the Column Map column on the **Tables** tab.
- The Prefix/Suffix dialog is not available for a Compare Table Map.
- In the Extract Request Editor, the Prefix/Suffix dialog applies to the Object Name column on the **Object List** tab.

The Prefix/Suffix dialog includes the following:

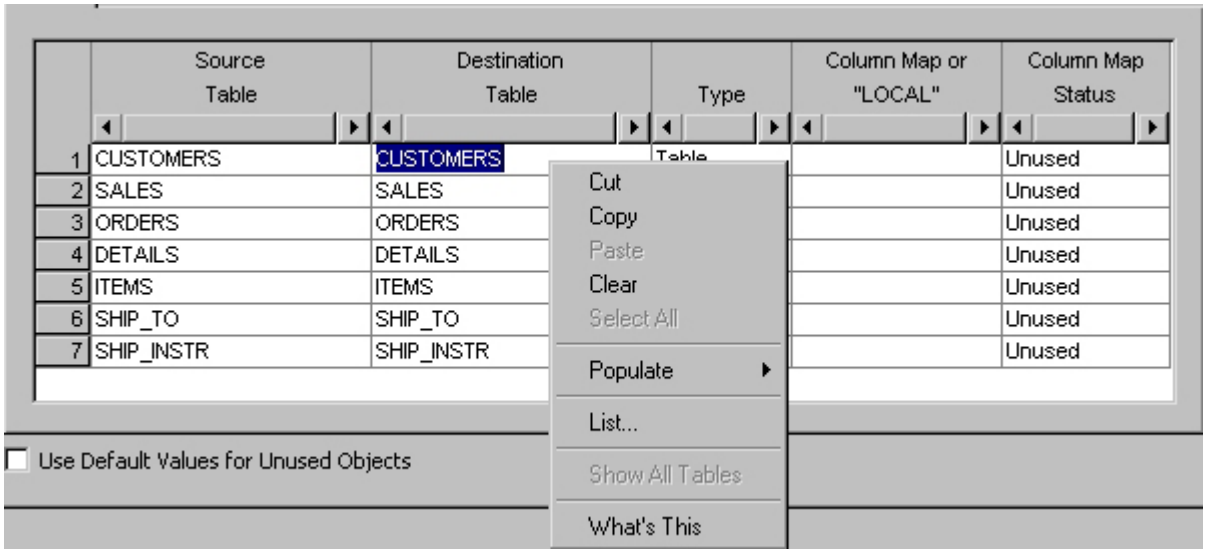
Column
Displays the name of the column header you right⌂clicked.

Operation
Select whether to add a prefix or suffix to, or replace, each item in the column, then enter the value to use.

Apply to
Select the part of the three part object name, as applicable, to apply the prefix or suffix, or to replace.

Grid Column Shortcut Menu

Right-click a grid column to select from shortcut menu commands to edit the entry in a grid cell. The typical field editing function commands are available and correspond to the specific grid column.



Using Find and Replace

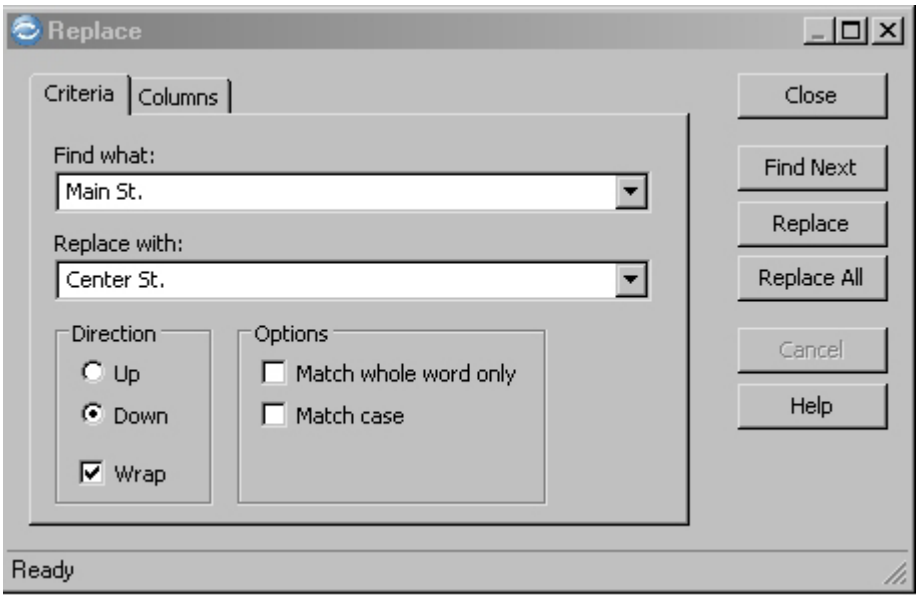
The **Find** and **Replace** commands are available from a right-click menu in any grid column heading.

- In the Find dialog, you enter a search string and direct Find to locate the string in one or more cells in the grid.
- In the Replace dialog, you enter a search string and a replacement string, and direct Replace to locate and replace occurrences of the search string in the grid.

You select the **Find** and **Replace** options on the **Criteria**, **Columns**, and **Selection** tabs. The **Selection** tab is available only during a Point and Shoot session. For details on using Point and Shoot, refer to "Edit Point and Shoot List" on page 109.

Find/Replace Dialog

To open the Find/Replace dialog, right-click a grid column heading and select **Replace** (or **Find**) from the shortcut menu.



The Find/Replace dialog displays the string from your last search or replace.

Tabs

Use the appropriate tabs to provide search **Criteria** and select the target **Columns**.

Command Buttons

Command buttons specific to the Find/Replace dialog are:

Find Next
Locate the next instance of the search string specified in the **Find what** box.

Replace
Replace the search string specified in the **Find what** box with the replacement string specified in the **Replace with** box.

Replace All
Replace all instances of the search string with the replacement string.

Criteria Tab

Use the **Criteria** tab (shown in the previous example) on the Find/Replace dialog to provide search criteria. You indicate what to find and the direction of the search, and can select options for applying the criteria.

Details

Enter search criteria, supply details, and make selections as follows:

Find what
Search string. Type a value or click the down arrow to select from a list of recent search strings.

Replace with
Replacement string. Type a value or click the down arrow to select from a list of recent replacement strings.

Direction

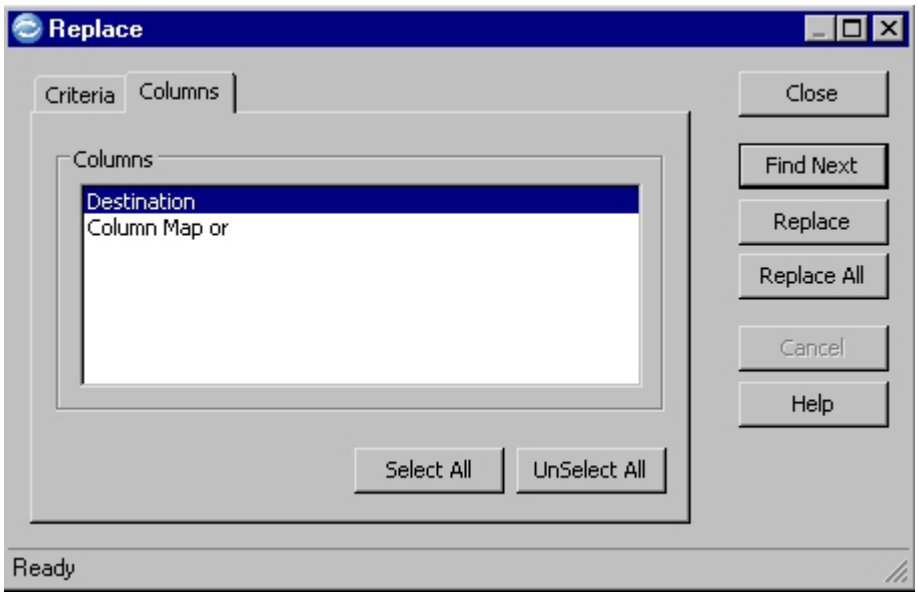
Direction of the search from the position of the cursor in the grid. Select **Up** or **Down** and **Wrap** to continue until all the data has been searched.

Options

Instructions for applying the search string. Select **Match Whole Word Only** to limit the search to exact word matches and **Match Case** to perform a case-sensitive search.

Columns Tab

Use the **Columns** tab on the Find/Replace dialog to limit the search to selected grid columns.



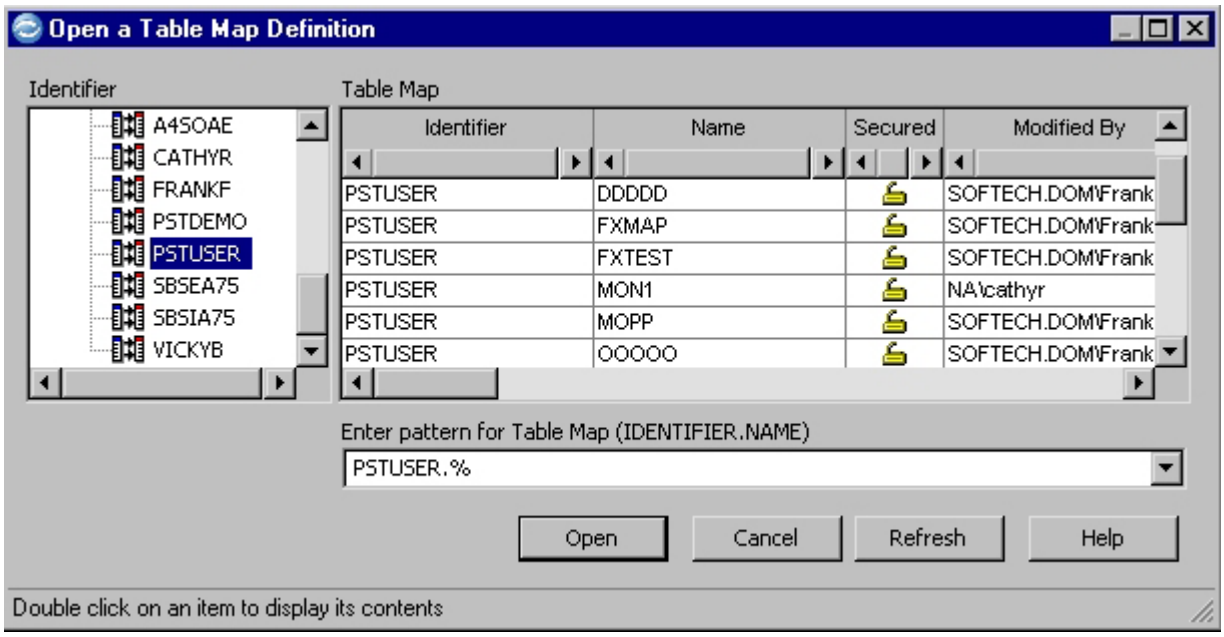
The **Columns** tab lists all the grid columns. Select the columns you want to target. If targeting most columns, click **Select All** and select specific grid columns for exclusion. To exclude all grid columns, click **UnSelect All**. Then, select individual columns to search.

Using the Open Dialog

When you select **Open** from the **File** menu in the main window or an editor, the Opendialog is displayed, listing object definitions to select and open.

Open from the Main Window

In the main window, select **Open** from the **File** menu to list all objects in the Optim Directory. You can open as many editors as you require and can have several instances of the same editor open at one time.



The Open dialog is divided into two areas. The object types and identifiers (object tree) are arranged alphabetically on the left and the corresponding objects are listed on the right. Objects are listed according to the appropriate naming convention. See “Use a Pattern” on page 27 for further information on “Naming Conventions” on page 27.

Note: When opened, the display is focused on the object identifier and object definition list last displayed. The descriptive text in the title bar and the labels above the object tree and grid change according to the object type you select.

Secured Objects

If Optim Security is initialized for the Directory, the **Secured** column is displayed and includes the following icons:

The object has an Access Control List.

The object does not have an Access Control List.

For more information, see “Object Security” on page 30.

Open from an Editor or Dialog

When you select **Open** from a **File** menu on an editor or dialog, the appropriate list of objects is displayed. In all other ways, the appearance and functions of the Open dialog are the same as when the dialog is opened from the main window.

Note: Several instances of an editor can be open at one time, but if you open an object definition from within an object editor, the new information replaces the current display, so you should save the current definition to prevent losing your changes.

Object Tree

The object tree lists object types and identifiers. Objects associated with a selected identifier are listed on the right. The object list varies to match the identifier you select. When you select an identifier, the Pattern value changes to the identifier name followed by the symbol, %.

To list identifiers for an object type:

- Double-click the object type.
- Click the plus sign to the left of the object type.
- Highlight the object type, and then press plus sign (on the numeric keypad) or right arrow.

To display the object list associated with an identifier:

- Double-click the identifier.
- Highlight the identifier, and then click **Open** or press Enter.
- Select the identifier and click **Refresh** or press Alt+R.

To close a list of identifiers for an object type:

- Double-click the object type.
- Click the minus sign to the left of the object type.
- Highlight the object type, and then press minus sign (on the numeric keypad) or Alt+Enter.

Use a Pattern

You can use a Pattern to search for a particular object in the Optim Directory or to limit the list of objects displayed in the Open dialog or other list. When entering a pattern, you must use a naming convention appropriate for the object or list of objects you want to display.

Naming Conventions

When you create objects, you may want to use logical naming conventions to identify the use for each and to organize the objects for easy access. In general, Optim supports the use of _, @, # or letters in object names and qualifiers; numeric digits also permitted for characters other than the first. The naming conventions for objects follow.

Database Table or Primary Key

The fully qualified name of a primary key matches the name of the database table for which it is defined: *dbalias.creatorid.tablename*

dbalias Alias that identifies the database where the table resides (1 to 12 characters).

creatorid
Creator ID assigned to the table (1 to 64 characters).

tablename
Base table name (1 to 64 characters).

Note: When Object Security is enabled, the second and third parts of a Primary Key (i.e., the *creatorid* and *tablename*) are restricted to a combined total of 64 characters.

DB Alias

The fully qualified name of a DB Alias is *dbalias*.

dbalias Base name assigned to the DB Alias (1 to 12 characters).

Access Definition

The fully qualified name of an Access Definition is *identifier.name*

identifier
Qualifier assigned to the Access Definition (1 to 8 characters).

name Base name assigned to the Access Definition (1 to 12 characters).

Relationship

The fully qualified name of a relationship is *dbalias.creatorid.tablename.constraint*

dbalias Alias that identifies the database where the child table resides (1 to 12 characters).

creatorid
Creator ID assigned to the child table (1 to 64 characters).

tablename
Base name of the child table (1 to 64 characters).

constraint
Name of the relationship (1 to 64 characters).

Note:

- When Object Security is enabled, the second, third, and fourth parts of a Relationship (i.e., the *creatorid,tablename*, and *constraint*) are restricted to a combined total of 64 characters.
- If **Enforce DBMS Rel. Name Lengths** is selected on the Product Options dialog **Database** tab, the *constraint* length must comply with the database management system's limit. See the *Installation and Configuration Guide* for detailed information on the Product Options dialog.

Table Map

The fully qualified name of a Table Map is *identifier.name*.

identifier
Identifier assigned to the Table Map (1 to 8 characters).

name Name assigned to the Table Map (1 to 12 characters).

Column Map

The fully qualified name of a Column Map is *identifier.name*.

identifier
Identifier assigned to the Column Map (1 to 8 characters).

name Name assigned to the Column Map (1 to 12 characters).

Request Name

The fully qualified name of a process request (for example, an Archive Request) is *identifier.name*.

identifier
Identifier assigned to the request (1 to 8 characters).

name Name assigned to the request (1 to 12 characters).

Wildcard Characters

Use percent (%) to represent any number of characters and underscore (_) to represent a single character in a pattern. (You cannot use % in *identifier* or *dbalias* and must select an option on the **General** tab of

Personal Options to use the underscore as the SQL LIKE character.) The following examples demonstrate the use of wildcard characters to list tables with names that meet specific criteria:

dbalias.creatorid.%
: List all tables identified by a specific DB Alias and Creator ID.

dbalias.%.tablename
: List all occurrences of a specific table identified by a particular DB Alias, regardless of the Creator ID.

dbalias.%.%
: List all tables identified by a particular DB Alias, regardless of the Creator ID.

dbalias.creatorid.TAB%
: List all tables identified by a specific DB Alias and Creator ID beginning with TAB in the name.

*dbalias.creatorid.TAB_*
: List all tables identified by a specific DB Alias and Creator ID having a four⌂character name beginning with TAB.

dbalias.creatorid.T_ _B
: List all tables identified by a specific DB Alias and Creator ID having a four⌂character name beginning with T and ending with B.

Show Pattern for

When using the **Open** command from the Primary Key Editoror Relationship Editor, you can further limit the list of objects by using **Show pattern for** options to display a specific type of object. When you select an option, the list is refreshed.

Database
: List primary keys or relationships defined to a database.

Optim List explicit and generic primary keys or relationships defined to the Optim Directory.

Both List primary keys or relationships defined to both a database and the Optim Directory.

Relationships Related to Table

When using the **Open** command from the Relationship Editor, you can also limit the list of relationships using the **Relationships Related to Table** options. Select the **Relationships Related to Table** check box to list relationships for a particular table. Enter the table name. If you do not specify the Creator ID, the default Creator ID is assumed.

Select the **Show Only Directly Related Tables** check box to limit the list further. If you want to display all possible relationships for the table, clear the check box.

Select and Open an Object

In the Open dialog, you can use any of the following methods to select and open an object in the appropriate editor or dialog:

- Select the object and press **Enter**.
- Select the object and click **Open**.
- Double-click the object.
- Type the entire object name directly into the **Pattern** box and click **Open**.

Delete an Object using the Shortcut Menu

Right-click an object name to display the shortcut menu. To remove an object from the list, select **Delete**. If you delete all objects associated with an identifier, the identifier is deleted when you close the dialog.

Optim Security

Optim Security may secure objects and functions in Optim. Optim Security is in three types: Functional Security, Object Security, and Archive File Security.

- Functional Security controls access to features and functions in Optim.
- Object Security controls access to Optim Directory objects such as Column Maps and Access Definitions. An object is secured by associating it with an Access Control List. Object Security controls the ability to perform the following functions on a secured object: Delete, Open, Save, and Run.
- Archive File Security controls access to tables and columns in Archive Files. Access is defined in File Access Definitions.

Note: To use Optim Security, the Security Administrator must initialize security for the Optim Directory and enable the features. See the *Installation and Configuration Guide* for further information about Functional Security, Object Security, and Archive File Security.

Object Security

An object in the Optim Directory (for example, a Column Map or Archive Request) is secured by associating it with an Access Control List (ACL), which provides access permissions for the object. Secured objects are then protected by settings in the ACL if Object Security is enabled by the Security Administrator.

The status bar in an object editor and the **Secured** column in the Open dialog display the following icons to indicate if an object is secured by an ACL:

The object is secured by an ACL.

The object is not secured by an ACL.

When you print a secured object, the secured object icon is also displayed at the bottom of the report.

Editing Optim Object ACLs

You can modify the ACL for an object using the following commands available from the **Tools** menu in the object editor or the shortcut menu in the Open dialog:

Edit ACL
Open the Access Control List Editor. Use the editor to create an ACL for the object or to edit the existing ACL.

Delete ACL
Delete the ACL for the object.

Note:

- This option is available for secured objects only.
- This option is available only to users who have permission to delete the ACL for an object, or if the object is automatically secured when saved.

Object Security and Object Names

The size of the fully qualified name for a Primary Key and a Relationship is restricted when Object Security is enabled.

- When Object Security is enabled, the second and third parts of a Primary Key (i.e., the *creatorid* and *tablename*) are restricted to a combined total of 64 characters. When Object Security is not enabled, each of those parts may consist of up to 64 characters.
- When Object Security is enabled, the second, third, and fourth parts of a Relationship (i.e., the *creatorid*, *tablename*, and *constraint*) are restricted to a combined total of 64 characters. When Object Security is not enabled, each of those parts may consist of up to 64 characters.

Other object names — such as those for Access Definitions, Table Maps, and Column Maps — are not affected by this restriction because their names do not exceed the above character limit.

Importing and Exporting Secured Objects

You must have read access to export a secured object. When you import automatically secured objects, the new ACLs are modeled after the Optim Object Template ACL. If the Optim Object Template ACL has not been defined, you must define an ACL for each object that requires an ACL.

Save an Optim Object

Forms of the **Save** command are as follows:

Save

Use the **Save** command to save a new Optim object or update an existing one:

- To save a new Optim object, select **Save** from the **File** menu in the Editor to open the Save the <*objecttype*> dialog. Type a name in the **Pattern** box and click **Save**.
- To save an existing Optim object, select **Save** from the **File** menu in the Editor. The current version replaces the original version. You can continue editing the current version.

Save As

Use the **Save As** command to save an Optim object under a new name, preserve the original, and display the newly named version for editing. To use this command, select **Save As** from the **File** menu in the Editor. In the **Save the <*objecttype*>** dialog, type a name in the **Pattern** box and click **Save**.

Save Copy As

Use the **Save Copy As** command to save a copy of an Optim object under a new name, preserve the copy, and continue editing the original. To use this command, select **Save Copy As** from the **File** menu in the Editor. In the **Save the <*objecttype*>** dialog, type a name in the **Pattern** box and click **Save**.

Secured When Saved Objects

When an object is secured when saved, an Access Control List (ACL) automatically secures the object. The ACL is modeled after the Optim Object Template ACL. After saving the object, the ACL can be modified.

The Security Administrator determines which objects are secured when saved and also defines the Optim Object Template ACL.

If the Optim Object Template ACL has not been defined, you will be prompted to define an ACL for a secured-when-saved object.

Delete an Optim Object

When you delete an Optim object, the definition is removed from the Optim Directory. You can delete an object from the Editor or from the Open dialog.

From the Editor

While in the appropriate Editor, display the object you want to delete and select **Delete** from the **File** menu. In the confirmation popup, click **Yes** to confirm the delete.

From the Open Dialog

From the Open dialog, choose one of the following:

- In the Editor, select **Open** from the **File** menu. Left-click to select the object you want to delete. Press **Delete** or right-click to open a menu and select **Delete**. In the confirmation popup, click **Yes** to confirm the delete.
- In the main window, select **Open** from the **File** menu. In the Open dialog, select the object type (e.g., Table Map) to expand the list of identifiers. Select the object you want to delete. Press **Delete** or right-click to open a menu and select **Delete**. In the confirmation popup, click **Yes** to confirm the delete.

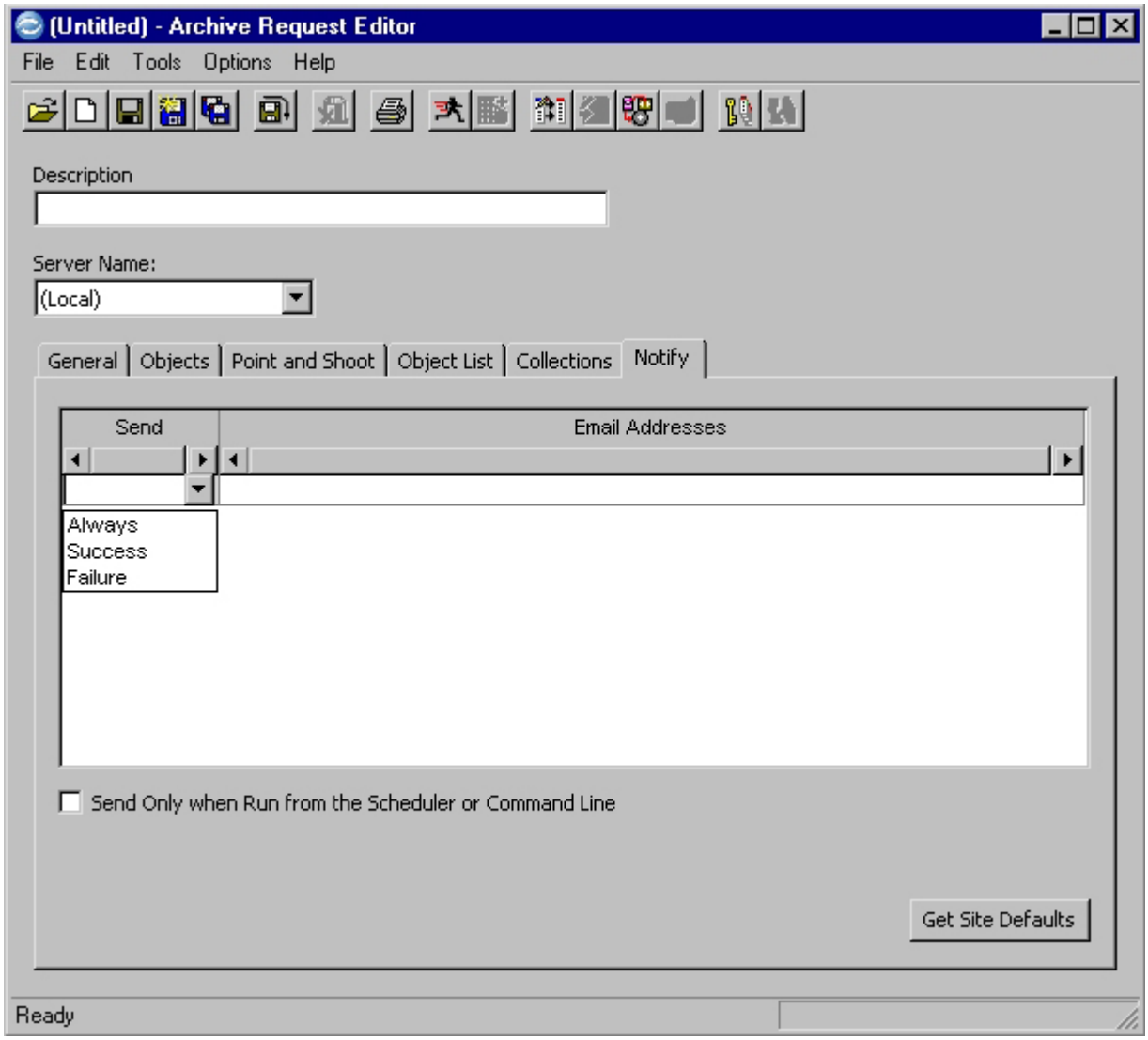
Confirm Option

By default, Optim prompts you to confirm before deleting an object. To disable this feature, select **Personal** from the **Options** menu in the main window. See “Confirm Tab” on page 430 for more information.

Email Notification

Use the **Notify** tab on an action editor to provide options and email addresses for automatic notification of the outcome of the process. The process report generated when the process completes is automatically sent as an attachment.

Note: Before using email notification, the desired email program must be installed. For Windows, the email client must be defined as the default, and set up to interface with MAPI. For UNIX or Linux, a valid copy of SENDMAIL must be configured correctly.



Send

For each listed email address, select an option to send a message as determined by the outcome of the process. Click the **Send** column to select **Always**, **Success**, or **Failure** from a drop-down list.

Email Addresses

Enter an email address for automatic notification at the completion of the process. Enter one address per line.

Send Only when Run from the Scheduler or Command Line

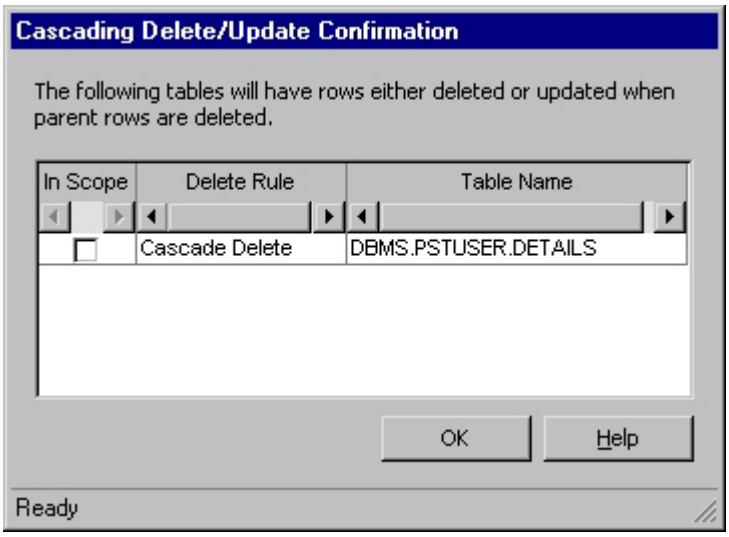
Select this check box to limit automatic email notification to a process run from the Scheduler or the command line interface.

Get Site Defaults

Click this button to add default email addresses listed on the **Notify** tab in Personal Options to the grid (duplicate entries are not repeated). See “Notify Tab” on page 468 for more information.

Cascading Delete/Update Confirmation Dialog

Optim performs a cascading delete/update check when you save an Access Definition or execute a process. If a cascading delete or update may affect a table that is not explicitly referenced in an Access Definition or a process, the Cascading Delete/Update Confirmationdialog is displayed.



The Cascading Delete/Update Confirmation dialog lists the names of all tables for which rows may be deleted or updated because of deletes or updates to the parent table. This dialog is provided for information only and cannot be modified.

Note: The **Warn on Cascade Delete/Update** setting in Product or Personal Options determines the display of this dialog. See "General Tab" on page 429 in this manual or Product Options in the *Installation and Configuration Guide* for further information.

In Scope

When selected, the **In Scope** box indicates a table is explicitly included in the Access Definition or process. When cleared, it indicates a table is not explicitly included in the Access Definition or process.

Note: The Cascading Delete/Update Confirmation dialog is displayed if the **In Scope** box is cleared.

Delete Rule

Delete Rule indicates the rule for deleting from the child table when a row in the parent table is deleted or updated.

Cascade Delete
All child rows are deleted when the parent row is deleted.

Cascade Set Null
Foreign key columns are set to NULL when the parent row is deleted.

Note: When you delete or update a row in a parent table for which a Cascade Delete or Cascade Set Null rule is defined, the related rows in the child table will be adjusted appropriately, whether or not explicitly included in the Access Definition or process.

Table Name

Table Name identifies the table affected by the delete or update of parent rows.

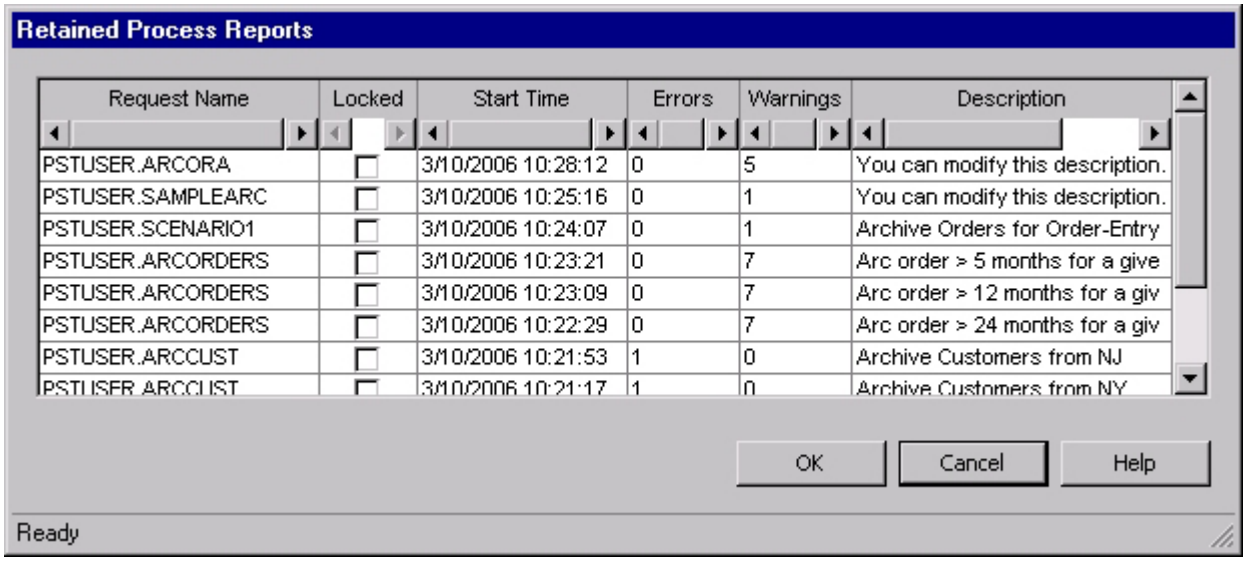
OK or Cancel

Select **OK** to continue saving the Access Definition or executing the process. Select **Cancel** to abort the process.

Retained Process Reports Dialog

Use the Retained Process Reports dialog to review and manage retained process reports for a process. Select **Redisplay Results** on the **File** menu in a request editor to display the **Current** and **All** commands. Select **Current** to redisplay the report from the previously run process. Select **All** to display the Retained Process Reports dialog.

Note: Redisplay Results is unavailable if a process has not been run and no reports are retained. The **Current** command is unavailable until a process is run. The **All** command is unavailable if no reports are retained.



The Retained Process Reports dialog includes the following details:

Request Name
The fully qualified name of the process request for which the report was generated.

Locked
The check box indicates whether the report is locked. (A locked report cannot be deleted.)

Note: To lock or unlock a report, right-click a grid row and select from the shortcut menu.

Start Time
The date and time processing began. **Start Time** is useful for distinguishing reports generated for a single process request.

Errors The number of errors that occurred during processing.

Warnings
The number of warnings that occurred during processing.

Description
Text from the **Description** on the request editor. You can modify this description to provide more detail on the contents of the retained process report. Modifying the description here does not change the description on the request editor.

You can display any report by double-clicking the grid row.

Note: The availability of the Retained Process Reports dialog and the maximum number of reports you can retain for an action are determined by the **Report Retention** settings in Personal Options. See "Actions Tab" on page 461 in Chapter 18, "Personal Options," on page 427 for further information.

Shortcut Menu Commands

You can use shortcut menu commands on the Retained Process Reports dialog to lock, unlock, delete, or display retained reports. Right-click a grid row to select from the following shortcut menu commands:

Lock Lock the selected report. A locked report is not deleted when a new report is saved or when you select **Remove All** from the shortcut menu.

Note: A warning message is displayed if the number of locked reports equals or exceeds the **Report Levels** setting in Personal Options. You can increase that value or unlock one or more reports. Optim retains all locked reports and one additional report (for the previously run process), until the conflict is resolved.

Unlock
Unlock the selected report.

Unlock All
Unlock all reports.

Remove
Delete the selected report.

Remove All
Delete all unlocked reports.

Display Results...
Display the selected report.

Chapter 3. Optim Directory

The Optim Directory is a set of tables in which Optim stores objects needed for processing and tracking processing status. Before using Optim, you must use the **Configuration** program to create or configure the Optim Directory tables and stored procedures needed to access the Directory.

See the *Installation and Configuration Guide* for more information.

Important Optim Directory objects include:

- **DB Aliases**. A DB Alias provides parameters needed to connect with a specific database. It is used as a high-order qualifier for an object or table name and tells Optim how to access the appropriate database.
- **Access Definitions**. An Access Definition identifies the set of related data to be processed by Optim. It identifies the database tables and their relationships, and provides criteria to select specific rows within tables.
- **Table Maps**. A Table Map defines the correlation between two tables or sets of tables in an Insert, Update, Restore, or Compare Process and can be used to exclude one or more tables from processing.
- **Column Maps**. A Column Map defines the correlation between columns in two tables. Additionally, a Column Map can be used to transform data, age dates in tables, and exclude one or more columns from processing.
- **Column Map Procedures**. A Column Map Procedure is a custom program that is referenced by a Column Map and is used for special processing and data manipulation that is beyond the scope of native Column Maps.
- **Primary Keys**. Values in primary key columns uniquely identify each row in a database table.
- **Relationships**. A Relationship determines the parent or child rows to be processed and the order in which they are processed.
- **Processing Definitions**. A processing definition provides parameters needed to run a process, including the names of generated files, the Access Definition used for the process, and the source of processed data. You can sometimes use parameters in a process definition to override parameters defined for the Access Definition. Process definitions include:

Archive Requests	Load Requests
Extract Requests	Delete Requests
Compare Requests	Report Requests
Insert Requests	Edit Requests
Convert Requests	Restore Requests

- **Utility Definitions**. A utility definition is used for some type of specialized processing. Utility definitions include:

Calendars	Storage Profiles	Currency Tables

- **Security Definitions**. A security definition provides parameters needed for Optim Security.
  - Access Control Domains (ACDs) map roles to network accounts.
  - The **(Default)** ACD provides Functional Security parameters.
  - Access Control Lists (ACLs) provide parameters used with Object Security.
  - File Access Definitions (FADs) provide parameters used with Archive File Security.
- **Archive File Registrations**. Archive File registrations provide all information needed to locate and process data on Archive Files.

After one or more Optim Directories are created, you can use the Optim Directory Editor to:

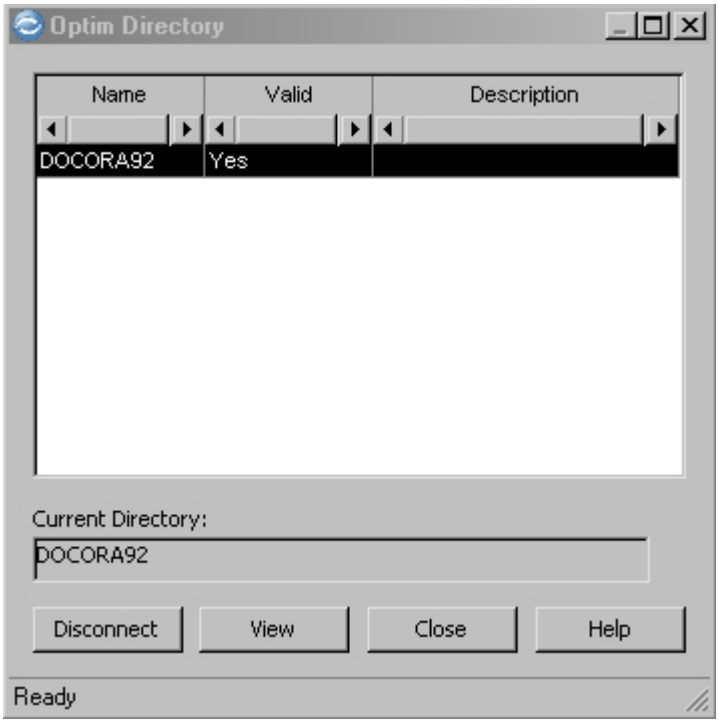
- Browse a list of Optim Directories available to the workstation.
- Establish or change the Default Directory or connection for the workstation.
- View Optim Directory attributes while connected.
- Modify the code page or connection string for an Optim Directory connection.

Contents

This section describes the specifications on the Optim Directory Editor.

Open the Optim Directory Dialog

To open the Optim Directory dialog, select **Optim Directory** from the **File** menu on the main window.



A list of Optim Directories to which the workstation can connect is displayed. The **Valid** value indicates whether the Optim Directory is a valid Directory created using version 6.0 or 6.1 of Archive or the Relational Tools, or version 6.2 or later of Optim. If "No", the Directory is a 5.x version, and can be viewed, but not connected to or modified.

Current Directory indicates the Optim Directory to which the workstation is currently connected. If you disconnect from a Directory, the title changes to **Default Directory** and indicates the last Directory to which you were connected. The name of the Default Directory appears until you connect to a different Directory.

Command Buttons

Command buttons on the Optim Directory dialog change to reflect the status of the workstation. For example, when you first open the dialog, the workstation is not connected to an Optim Directory; only **Close** and **Help** are available to use. After you select an Optim Directory on the list, these commands are available:

Connect
Connect to the Optim Directory.

Modify
Open the Optim Directory Editor to edit attributes for the Optim Directory. (You cannot modify attributes while connected to a Directory.)

After you connect to an Optim Directory, these commands are available:

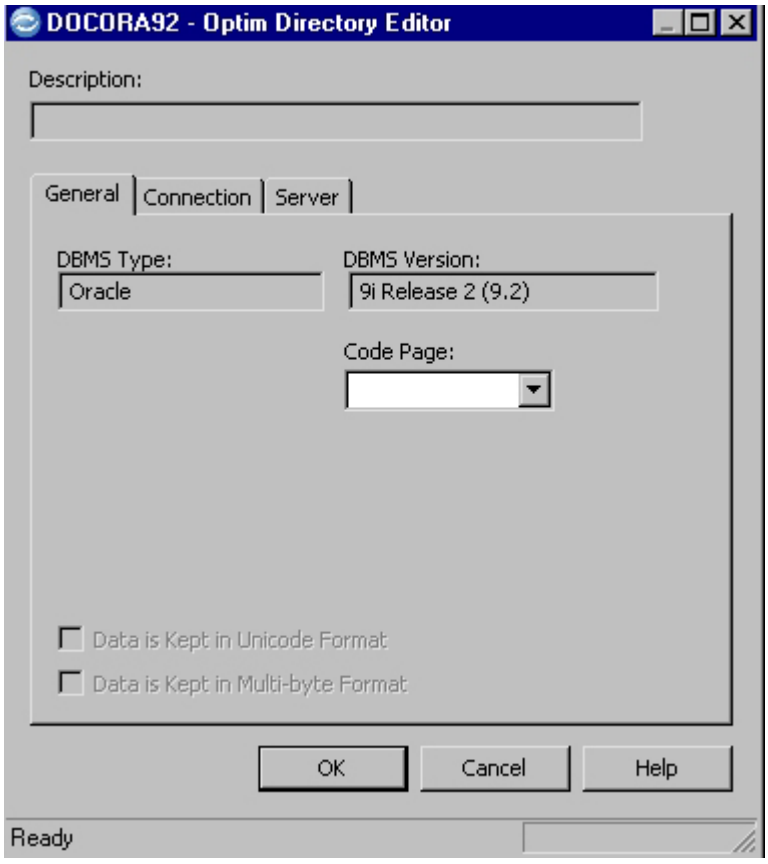
Disconnect
Disconnect from the Optim Directory.

Note: You must close all active dialogs listed in the main window to disconnect from the Optim Directory.

View Open the Optim Directory Editor to view attributes of the Optim Directory.

Using the Optim Directory Editor

In the Optim Directory dialog, click **View** or **Modify** to display the Optim Directory Editor.



Description

Text that describes the Optim Directory (up to 40 characters).

Tabs

Tabs on the Optim Directory Editor allow you to view or modify information pertaining to connection parameters for an Optim Directory:

General
Displays the DBMS type and version, and code page and character set values.

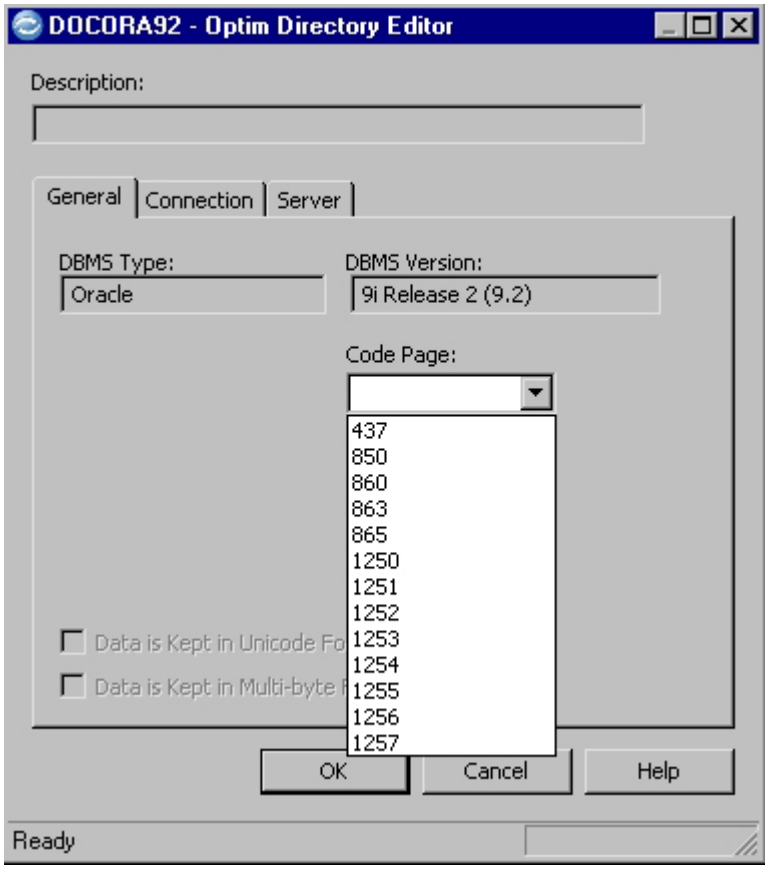
Connection
Displays the Connection String and Database Qualifier.

Server Displays the Server Connection Name.

Note: You cannot modify attributes of an Optim Directory to which you are connected.

General Tab

The **General** tab includes:



DBMS Type and Version

Type and version of the DBMS for the database within which the Optim Directory is stored.

Code Page

You can select the code page for the specified DBMS, however Optim can generally ascertain the proper code page from the database. Supported code pages are:

437 U.S. English

850 International Multilingual

860 Portuguese

863 Canadian, French

865 Nordic

1250 Eastern European

1251 Cyrillic

1252 U.S./Western European ANSI or Latin 1 (ISO 8859-1)

1253 Greek

1254 Turkish

1255 Hebrew

1256 Arabic

1257 Baltic Rim

blank Microsoft Windows 95 or Windows NT

Data is Kept in Unicode Format

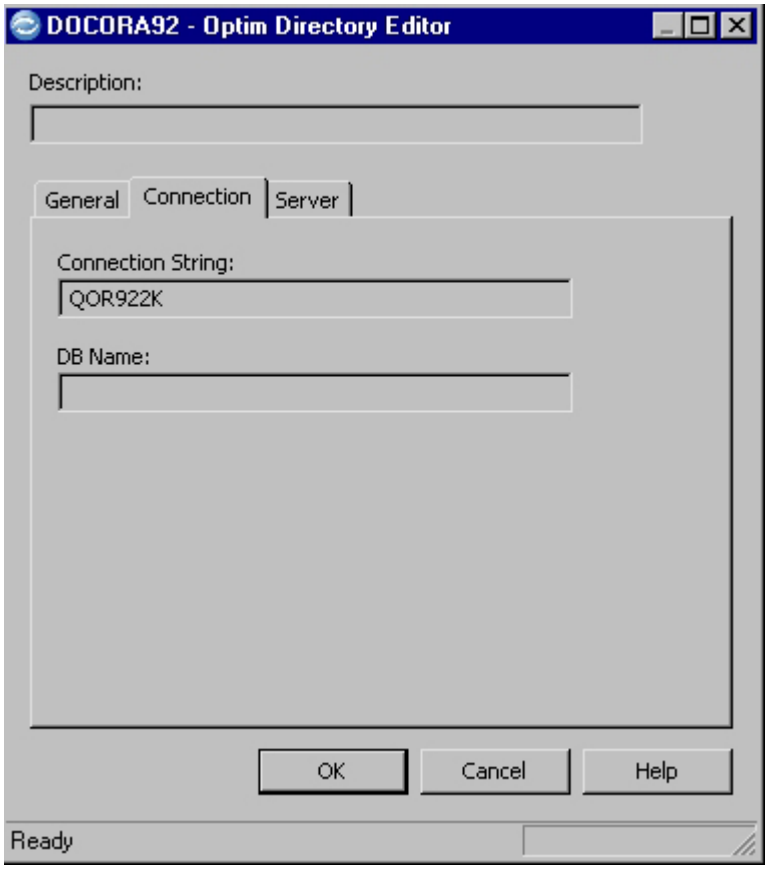
A check indicates that Optim Directory data is kept in Unicode format. This setting cannot be changed and **Data is Kept in Unicode Format** is displayed only when you are connected to an Optim Directory in a DBMS for which Unicode is supported.

Data is Kept in Multi-byte Format

A check indicates that Optim Directory data is kept in multi-byte format. This setting cannot be changed and **Data is Kept in Multi-byte Format** is displayed only when you are connected to an Optim Directory in a DBMS for which multi-byte is supported.

Connection Tab

The **Connection** tab includes:



Connection String

The value used by Optim to connect to the database that contains the selected Optim Directory. This value varies according to database management system:

DB2 Database Name or Alias

Oracle Host Machine Identifier

Sybase ASE
Server Name

SQL Server
Server Name

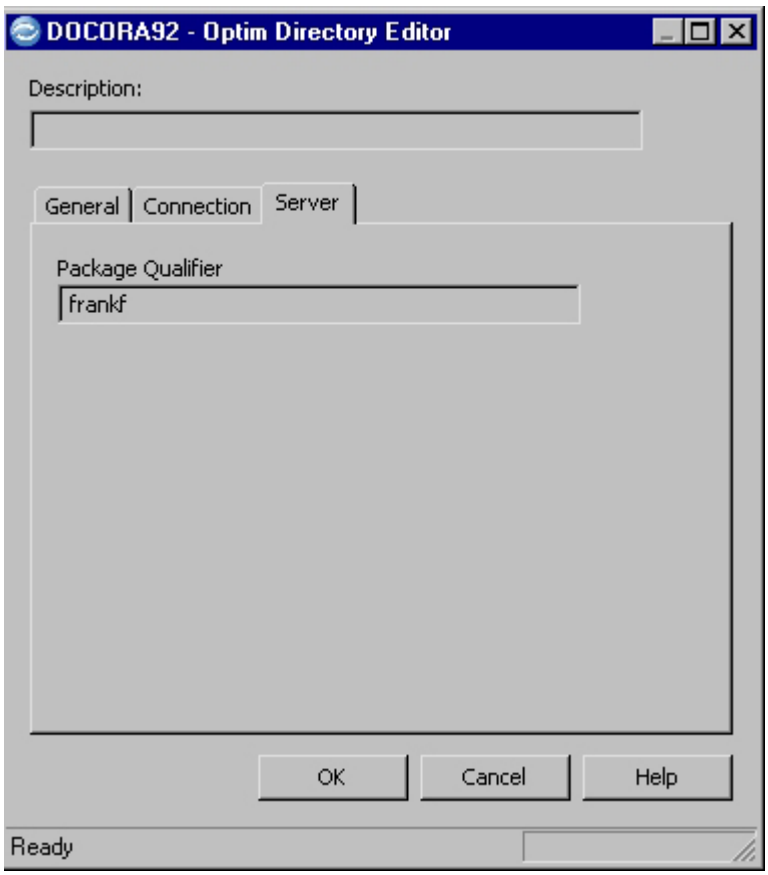
Informix
Server Name

DB Name

The DB Name (1 to 64 characters) identifies the database. This parameter applies to database management systems that support multiple databases within a single database server (for example, Sybase ASE and SQL Server).

Server Tab

The **Server** tab includes:



Package Qualifier

The Package Qualifier is the high-level qualifier for names of plans, packages or stored procedures used to access the Optim Directory tables.

Note: The label may vary according to the DBMS for the database in which the Optim Directory resides. For example, Sybase ASE, SQL Server, and Informix use Procedure Qualifier; DB2 uses Collection Name.

Chapter 4. Access Definitions

An Access Definition defines a set of related data to be processed by Optim.

Use an Access Definition to select a set of related data that:

- Archive copies to an Archive File or restores to a database.
- Move or Compare copies to an Extract File for processing.
- Edit displays for editing.

Note: Access Definitions can be stored in the Optim Directory and shared among users.

An Access Definition references tables, prescribes relationship usage, and includes criteria and other parameters:

Tables An Access Definition must reference at least one table, view, alias or synonym. The table, view, alias or synonym from which rows are selected first is called the Start Table. You can enter the name of a Start Table and easily include the names of all tables related to the Start Table (to a maximum of 24,000 tables) in the Access Definition.

Relationships
Relationships determine the traversal path for selecting data from tables. By default, relationships are traversed from parent to child, but you can control the direction of traversal using settings in the Access Definition. Relationships among tables referenced by the Access Definition are listed on the Relationship tab (to a maximum of 24,000 tables) so that you can select those to be used in processing and the direction in which they are traversed.

Table Specifications
Table Specifications define the data and provide special instructions for processes that use the Access Definition. Table Specifications include:

- Selection Criteria for focusing on a specific set of related data. You can visually select rows (Point and Shoot), specify SQL operators and values, and use substitution variables with default values.
- Column and Sort options for displaying data for Point and Shoot selection or for editing.

Additional Parameters
Additional parameters in an Access Definition include:

- Sampling parameters for extracting rows that correspond to a particular column value in the Start Table, or using a specified sampling rate (every nth row).
- Specifications for indexes that enable quick access to data in Archive Files.
- Specifications for custom SQL statements that can be executed at selected action phases of an Archive, Extract, Delete, or Restore Process.

Naming Conventions

A fully qualified Access Definition name is in the form *identifier.name*, where:

identifier
Qualifier assigned to the Access Definition
(1 to 8 characters).

name Base name assigned to the Access Definition
(1 to 12 characters).

A logical set of naming conventions can identify the use for each Access Definition and be used to organize them for easy access.

Contents

This section explains how to create and maintain Access Definitions, including how to:

- Reference the tables that contain the desired set of data.
- Select the relationships to be traversed between each pair of tables.
- Define selection criteria to limit the set of data.

Open the Access Definition Editor

Use the Access Definition Editor to create or edit Access Definitions. There are different ways to open the editor, depending on whether you want to create a new Access Definition or edit an existing Access Definition.

Create a New Access Definition

You can create an Access Definition from the main window or from the Access Definition Editor.

From the Main Window

To create an Access Definition from the main window:

1. In the main window, select **New** from the **File** menu.
2. Select **Access Definition** from the **Definitions** submenu to open the Access Definition Editor.
3. Specify the **Default Qualifier** for table names.
4. Specify the name of the **Start Table**.
5. Specify other tables/views and other values in the list of tables.
6. **Optional** – Type a brief description of the Access Definition.
7. In the Access Definition Editor, select **Save** from the **File** menu to open the Save the Access Definitiondialog.
8. In the **Pattern** box, type a unique name for the new Access Definition and then click **Save**.

From the Access Definition Editor

To create an Access Definition from the Access Definition Editor:

1. In the main window, select **Access Definition** from the **Definitions** menu to open the Access Definition Editorand last edited Access Definition.
2. Your next step depends on your purpose:
   - To create a new Access Definition, select **New** from the **File** menu in the Access Definition Editor.
   - To create a new Access Definition modeled on an existing one, open the desired Access Definition and select **Save As** from the **File** menu in the Access Definition Editor.
   - To create and store a copy of the active Access Definition and continue editing, select **Save Copy As** from the **File** menu in the Access Definition Editor.

These steps are the minimum required to create an Access Definition. See “Using the Editor” on page 47 for complete details.

Note: In addition, you can open the Access Definition Editorfrom the **Tools** menu in the Extract Request, Archive Request, or Compare Request Editor. For details, see the appropriate chapter in the corresponding user manual.

Select an Access Definition to Edit

You can select an Access Definition for editing from the main window or from the Access Definition Editor.

From the Main Window

To select an Access Definition to edit from the main window:

1. In the main window, select **Open** from the **File** menu to display the Open (object selection) dialog. (See "Using the Open Dialog" on page 25 for more information on the Open dialog.)
2. Double-click **Access Definition** to expand the Identifier list.
3. Double-click the desired identifier to display a list of Access Definitions.
4. **Optional** – Specify a Pattern to limit the list based on your criteria and click **Refresh**.
5. Double-click the desired Access Definition to open the Access Definition Editor.

Note: To select the last Access Definition you edited, select **Access Definition** from the Definitions menu in the main window to open the Access Definition Editor and the last edited Access Definition.

From the Access Definition Editor

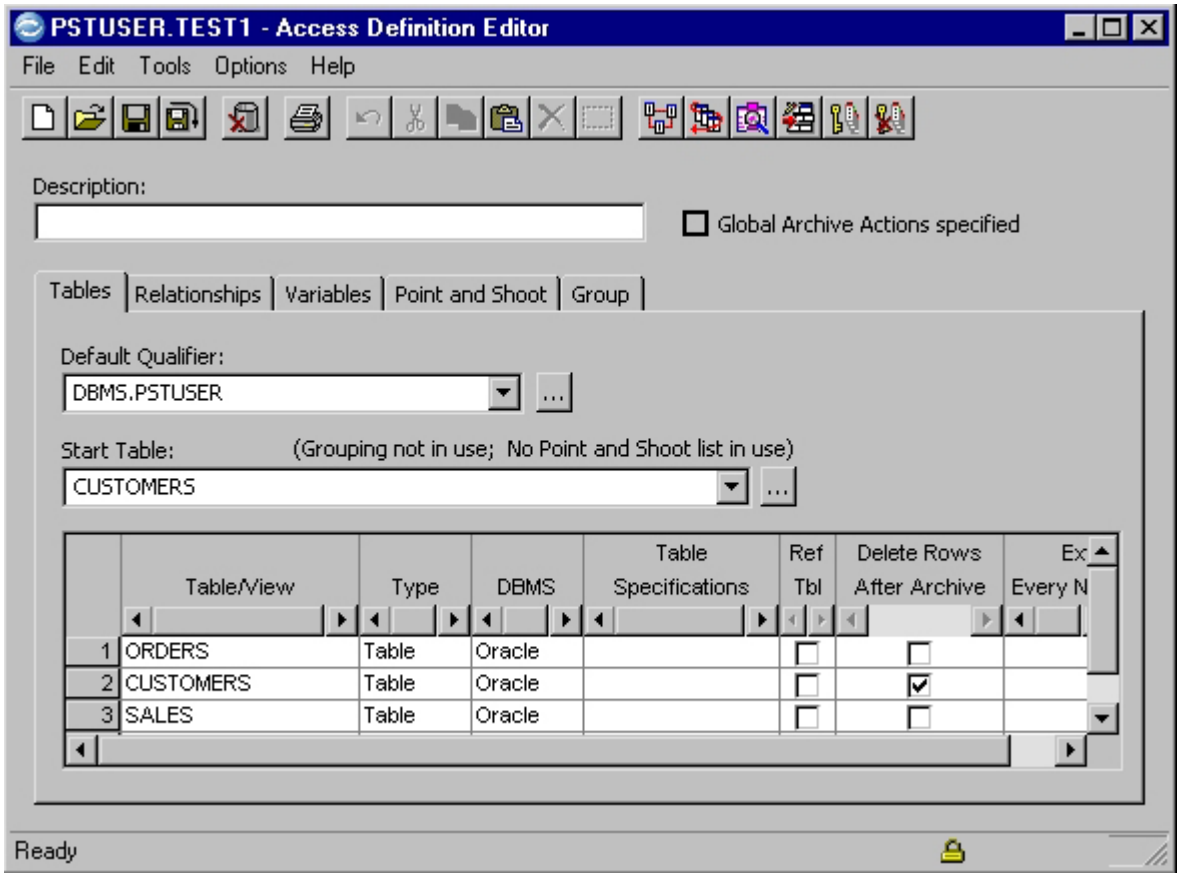
To select a different Access Definition:

1. In the Access Definition Editor, select **Open** from the **File** menu to display the Open an Access Definitiondialog. (See "Using the Open Dialog" on page 25 for more information on the Open dialog.)
2. Double-click the desired identifier to display a list of Access Definitions.
3. **Optional** – Specify a **Pattern** to limit the list based on your criteria and click **Refresh**.
4. Double-click the desired Access Definition to open the Access Definition Editor.

Using the Editor

Use the Access Definition Editor to create or edit an Access Definition.

Note: When you open the Access Definition Editor from a process request editor, parameters unique to other types of processes may be hidden or unavailable.



Description

Text that identifies the purpose of the Access Definition (up to 40 characters).

Global Archive Actions specified

A read-only check box that indicates whether Global Archive Actions are defined within the Access Definition. When table-specific Archive Actions are defined, an icon is displayed in the Table Specifications column.

Tabs

The Access Definition Editor allows you to enter specifications and select options used to define an Access Definition. It includes the following tabs:

Tables List tables, views, aliases or synonyms to be included in a process using the Access Definition. Designate the Start Table and define Table Specifications. Each time you open the editor, the **Tables** tab displays first.

Relationships
Designate the relationships to be traversed to select the set of related data from the referenced tables. (If editing or browsing data in a database, you can only review the relationships.)

Variables
Define substitution variables to be used with selection criteria, SQL statements, or Extract or Restore Archive Actions.

Point and Shoot
Provide parameters for a Point and Shoot list used to select specific rows from the Start Table. This tab is not available and settings on it do not apply when editing or browsing data in a database.

Group Provide parameters to select rows based on values in a particular column in the Start Table. This tab is not available and settings on it do not apply when archiving, editing, or browsing data in a database.

Menu Commands

In addition to the standard **File**, **Edit**, and **Tools** menu commands, you can select these commands from the **Tools** menu:

Convert to Local
Convert parameters in a named Access Definition, referenced in a process request, to Local. You can edit the Local Access Definition for the process without affecting the named Access Definition.

Indent
Display the tables listed in the Access Definition in a format that emphasizes the relationships between the tables.

Relationship Index Analysis
Analyze relationships between tables listed in the Access Definition to determine if new DBMS indexes would enhance performance. If additional DBMS indexes are needed, you can create them in Optim. See “Relationship Index Analysis” on page 106.

Show Steps
Describe the extract steps and traversal paths defined in the Access Definition in a narrative form. See “Show Steps” on page 108. This command is not available when editing or browsing data in a database.

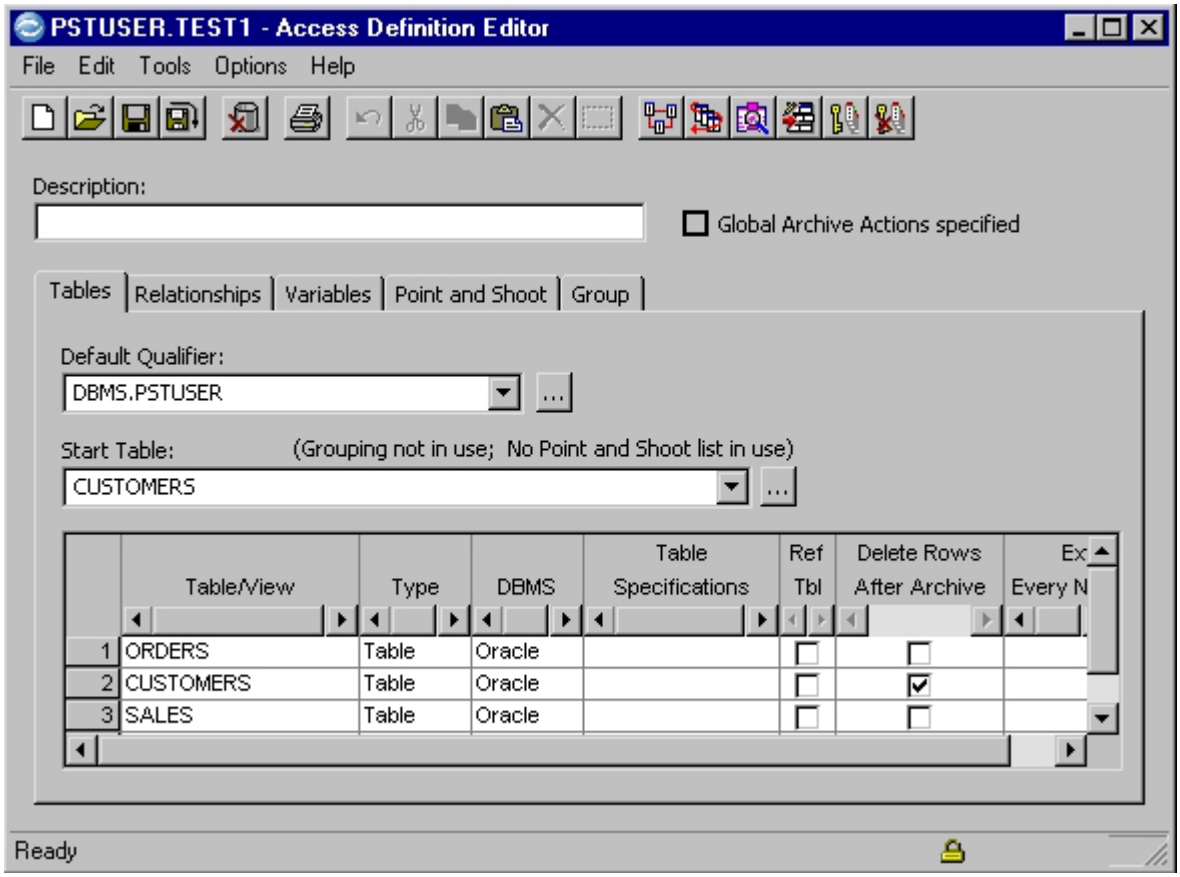
Edit Point and Shoot
Open the Point and Shoot Editor to allow you to select individual rows from the Start Table and save the selected rows in a Point and Shoot list. (See “Edit Point and Shoot List” on page 109.) This command is not available when editing or browsing data in a database.

You can select the following command from the **Options** menu:

Advanced Options
Display the **Uncommitted Read** column on the **Tables** tab and the **Table Access** and **Key Lookup Limit** grid columns on the **Relationships** tab. If you do not select this command, these columns are displayed only if a column contains a value other than the default. (See“Table Access” on page 60 and “Key Lookup Limit” on page 60.)

Tables Tab

Use the **Tables** tab to list the names of tables that contain the desired data and to define specifications for data in the listed tables, including selection criteria, sampling parameters, and archive options.



Default Qualifier

You must provide a Default Qualifier, which determines the prefix for unqualified or partially qualified table names. A fully qualified table name is in three parts — *dbalias.creatorid.tablename*. The Default Qualifier is: *dbalias* or *dbalias.creatorid*.

dbalias DB Alias of the database where a table is defined (1 to 12 characters).

creatorid
Creator ID, Owner ID, or Schema Name (depending on DBMS) assigned to the table (1 to 64 characters).

Start Table

The name of the table from which data is selected first. Any listed table, except a table designated as a reference table, can be the Start Table. If you enter a table name that is not listed, it is automatically added to the list. If you do not specify a Start Table, the first name entered in the Table List becomes the Start Table.

- To designate a listed table as the Start Table, right-click the table name and select **Set as Start Table** from the shortcut menu. You can also click the Start Table box arrow to select a table name.
- For Point and Shoot data selection, the Start Table must have a primary key. Use the Primary Key Editor to define a primary key, if needed. See Chapter 8, “Primary Keys,” on page 203for further information.

Selection Criteria Indicator

A notation above the Start Table box indicates whether Group Selection or a Point and Shoot list applies to the Access Definition. Possible notations are:

Grouping in use
The Access Definition uses Group Selection factors. For information on Group Selection, see "Group Tab" on page 74.

Grouping not in use
The Access Definition does not use Group Selection factors.

Point and Shoot List in use
The Access Definition uses a named Point and Shoot file. See "Edit Point and Shoot List" on page 109 for information on Point and Shoot data selection.

Local Point and Shoot List in use
The Access Definition uses a Local Point and Shoot list (saved with the Access Definition).

No Point and Shoot List in use
The Access Definition does not use a Point and Shoot list.

Table List Details

The grid on the **Tables** tab lists tables and views referenced by the Access Definition. Enter names directly in the **Table/View** grid column, or right-click to use shortcut menu commands to enter or edit entries on the list. The Access Definition can reference only one version of a table; you cannot reference a view or an alias of a table referenced by name. Also, an Access Definition can reference no more than 24,000 tables.

- General

Table/View
The name of a table or view.

Note: If a table name is not qualified with a DB Alias or Creator ID, the Default Qualifier is used.

Type The type of object referenced by the table name is automatically displayed:

Table Table name.

A-Table
Alias for a table.

S-Table
Synonym for a table.

View View name.

A-View
Alias for a view.

S-View
Synonym for a view.

View Error
View is invalid and unusable.

Unknown
Object is unknown. Unknown may indicate the:

- Referenced table no longer exists.
- Fully qualified name that results when the Default Qualifier is applied does not reference a table.

- Table name is entered incorrectly.

Inaccessible
Optim cannot connect to the database referenced by the DB Alias.

Note: From the Restore Request Editor and Archive Directory Maintenance Utility, the table may be inaccessible because it is restricted by Archive File Security.

DB Alias Unknown
DB Alias does not exist or is entered incorrectly.

DBMS
The name of the DBMS for a table is automatically displayed.

Table Specifications
Icons indicate the presence of selection criteria or other specifications for a table. Click an icon in this column to display the corresponding tab on the Table Specifications dialog. The icons are:

blank No selection criteria are specified.

Columns are rearranged or headings or LOB options are specified for data displays, or application associations for LOB data are specified.

Selection criteria are specified.

An SQL WHERE clause is specified.

Sort criteria are specified for data displays.

Archive Actions are defined for action phases of an Archive, Delete or Restore Process.

Archive Index parameters are specified.

File Attachments are specified.

Ref Tbl (Reference Table)
Select the check box to designate a table as a reference or look-up table. Unless selection criteria are specified for the reference table, all rows are selected. Specify any table as a reference table, except the Start Table.

Note: Although relationships associated with reference tables appear on the **Relationships** tab (with **Ref** status) they are not traversed during an Extract or Archive Process.

- Archive Process

Delete Rows After Archive

Select the check box to delete the selected rows from the database table after the rows are copied to an Archive File. Archive Request options allow you to override this Access Definition specification for an Archive Process or defer execution of the Delete Process.

Note: When you save the Access Definition, Archive performs a cascading delete/update check and displays the Cascading Delete/Update Confirmation dialog if:

- The Warn on Cascade Delete/Update option in either Product or Personal Options is set to Saving Access Definition or Always. (See Chapter 18, "Personal Options," on page 427 or refer to the*Installation and Configuration Guide*)
- **Delete Rows After Archive** is selected for at least one table in the Access Definition.
- The cascade delete or update affects at least one table that is not explicitly included in the Access Definition.

Click **OK** to return to the **Access Definition Editor**. (See "Cascading Delete/Update Confirmation Dialog" on page 34 for more information about this dialog.)

Row Limit

Enter a numeric value to limit the number of rows extracted from the Start Table. Valid values are 1 through 99999999 and apply to the Start Table only. This feature is useful for limiting the duration of an Archive Process, when timing is a consideration.

- Extract Process

Extract Parms

Enter numeric values to extract a sampling of rows or to limit the number of rows to extract.

Every Nth

Enter a numeric value to specify a sampling factor for a table. For example, if you enter 5, the process extracts every 5th row in the table, beginning with the 5th row. Valid values are 1 through 9999.

Row Limit

Enter a numeric value to limit the number of rows extracted from a table. Valid values are 1 through 99999999.

- Archive or Extract Process

Uncommitted Read

Select the check box to extract uncommitted rows from the database table. Selecting this option for tables with known performance problems may increase the speed of your Archive or Extract processes.

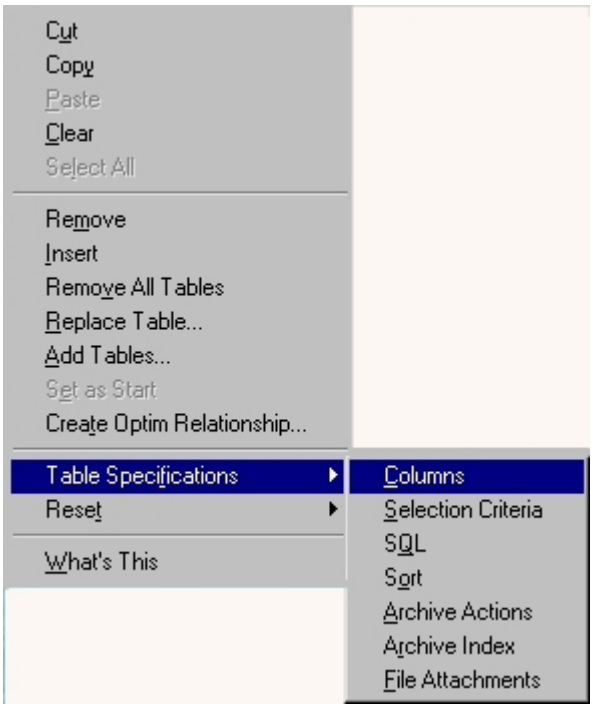
This option is available if the **Extract using Uncommitted Read** option in Product Options is set to **Default Active** or **Default Inactive** and the DBMS or version supports it. (Refer to the *Installation and Configuration Guide* .) Optim disables the option if regardless of the Product Options setting.

Note: If you choose to extract uncommitted rows, the relational integrity of the data in the Archive or Extract File may be compromised. Use caution when inserting or restoring from any Archive or Extract File with uncommitted rows.

Note: If using a Point and Shoot list to select Start Table rows, the Extract Process ignores any **Every Nth** and **Row Limit** parameters for the Start Table.

Shortcut Menu

Use the shortcut menu to edit entries on the Table List and define table specifications. Select a table by positioning the pointer in the appropriate **Table/View** grid cell. Right-click to open a shortcut menu for the selected table. Choose from the following:



Remove
Remove the grid row.

Insert Insert a blank grid row above the selected row.

Remove All Tables
Remove all rows from the grid.

Replace Table
Open the Select Table(s) dialog to choose a table name to replace the selected entry.

Add Tables
Open the Select Table(s) dialog to select one or more table names to add to the Table List.

Set as Start
Designate the table as the Start Table.

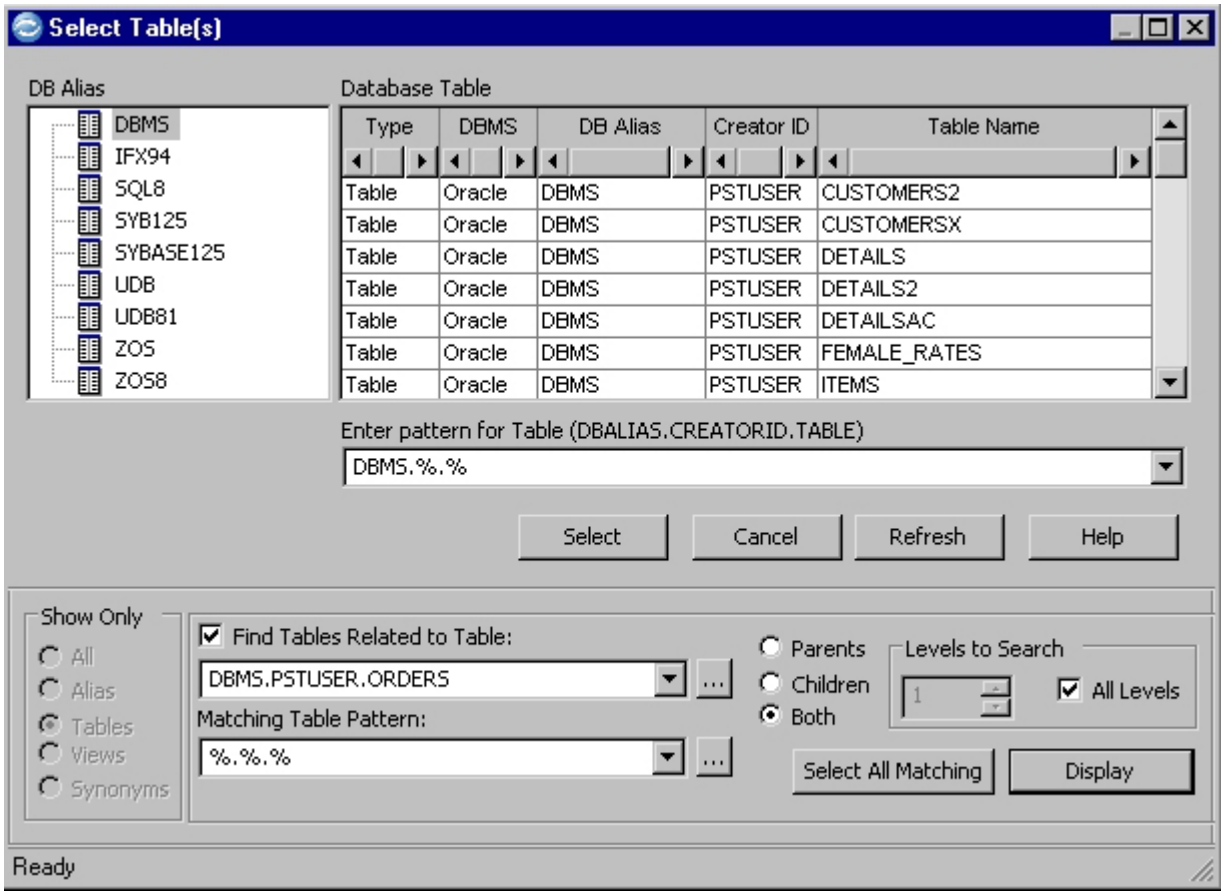
Create Optim Relationship
Open the Relationship Editor and designate the selected table in the Access Definition as the Parent table in a new relationship. Select a Child table from the Table Open dialog. If necessary, you can select **Reverse Parent/Child tables** from the **Tools** menu, before saving the new Optim Relationship.

Table Specifications
Open the **Table Specifications** submenu. See “Table Specifications” on page 76 for more information.

Reset Open the **Reset** submenu and select a table specification for removal.

Select Table(s) Dialog

When you select **Add Tables** or **Replace Table** from the shortcut menu, the Select Tables dialog is displayed. This dialog is also displayed when you use the Join command from the Table Editor.



The Select Table(s) dialog provides a list of tables for the selected database:

- DB Aliases for available databases are listed on the left. To list tables in a database, double-click the DB Alias or overtype the DB Alias in the **Pattern** box.
- Objects referenced by the selected DB Alias are listed in the Database Table grid in alphabetical order by Creator ID and Table Name. The type of object (table, view, alias, synonym), DBMS, and fully qualified name are provided.

Pattern

Use a **Pattern** to limit the list of database tables in the Select Table(s) dialog. After you specify a pattern, click **Refresh** to redisplay the list based on your criteria. See “Use a Pattern” on page 27 for more information.

Show Only

Select a **Show Only** option to determine the type of object listed in the Select Table(s) dialog. Select **All**, **Alias**, **Tables**, **Views**, or **Synonyms**. The list is refreshed when you make a selection.

Note: The **Show Only** options are disabled if **Find Tables Related to Table** is selected.

Find Tables Related to Table

Select **Find Tables Related to Table** and enter a fully qualified table name to retrieve the names of related tables. You can use a pattern, specify the type of relationship, and limit the number of generations to consider.

Matching Table Pattern

Use a three-part pattern, *dbalias.creatorid.tablename*, to insert names of tables into the Access Definition table list. You can use the % (percent) wildcard to represent one or more characters or use the _ (underscore) wildcard to represent a single character in a table name.

Relationship

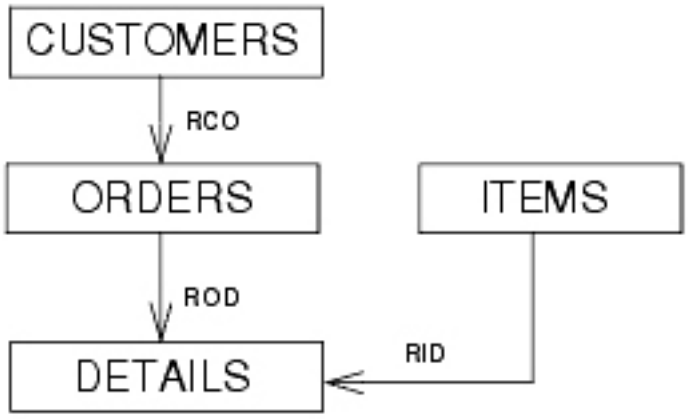
Select an option to include **Parents**, **Children**, or **Both**.

Levels to Search

Enter an explicit number of **Levels to Search**, or select the **All Levels** check box to include all levels. The value in the **Levels to Search** box is combined with the relationship option (**Parents**, **Children**, or **Both**) to determine the related tables.

Example

To understand how **Find Tables Related to Table** functions, assume that the following four tables are related as shown. The arrows indicate the direction of the relationships, from parent to child.



Following are examples of the sets of tables that match different combinations of relationship options and **Levels to Search**.

Relationship	Levels	Results
Children	**1**	Retrieve names of one generation of children. If the **Related to Table** name is CUSTOMERS, ORDERS is retrieved.
Parents	**1**	Retrieve names of one generation of parents. If the **Related to Table** name is DETAILS, ORDERS and ITEMS are retrieved.
Both	**1**	Retrieve names of one generation each of children and parents. If the **Related to Table** name is CUSTOMERS, ORDERS is retrieved. If the **Related to Table** name is ORDERS, CUSTOMERS and DETAILS are retrieved.

Relationship	Levels	Results
Children	**ALL**	Retrieve names of children, grandchildren, and so on. If the **Related to Table** name is CUSTOMERS, ORDERS and DETAILS are retrieved. If the **Related to Table** name is ORDERS, DETAILS is retrieved.
Parents	**ALL**	Retrieve names of parents, grandparents, and so on. If the **Related to Table** name is CUSTOMERS, which has no parents, no names are retrieved. If the **Related to Table** name is ORDERS, CUSTOMERS is retrieved. If the **Related to Table** name is DETAILS, CUSTOMERS, ORDERS and ITEMS are retrieved.
Both	**ALL**	Retrieve names of all tables (children, parents, grandparents, grandchildren, and so on). Use any table name in the example as the **Related to Table** name to obtain the same set of table names. This default option is the one used most often to retrieve table names for an entire data model.

Buttons

Select Select one or more table names and use **Select** to add the names to the list of tables on the **Tables** tab of the Access Definition Editor.

Cancel
Use **Cancel** to return to the Access Definition Editor.

Refresh
Use **Refresh** to update the list of table names displayed to match the Pattern.

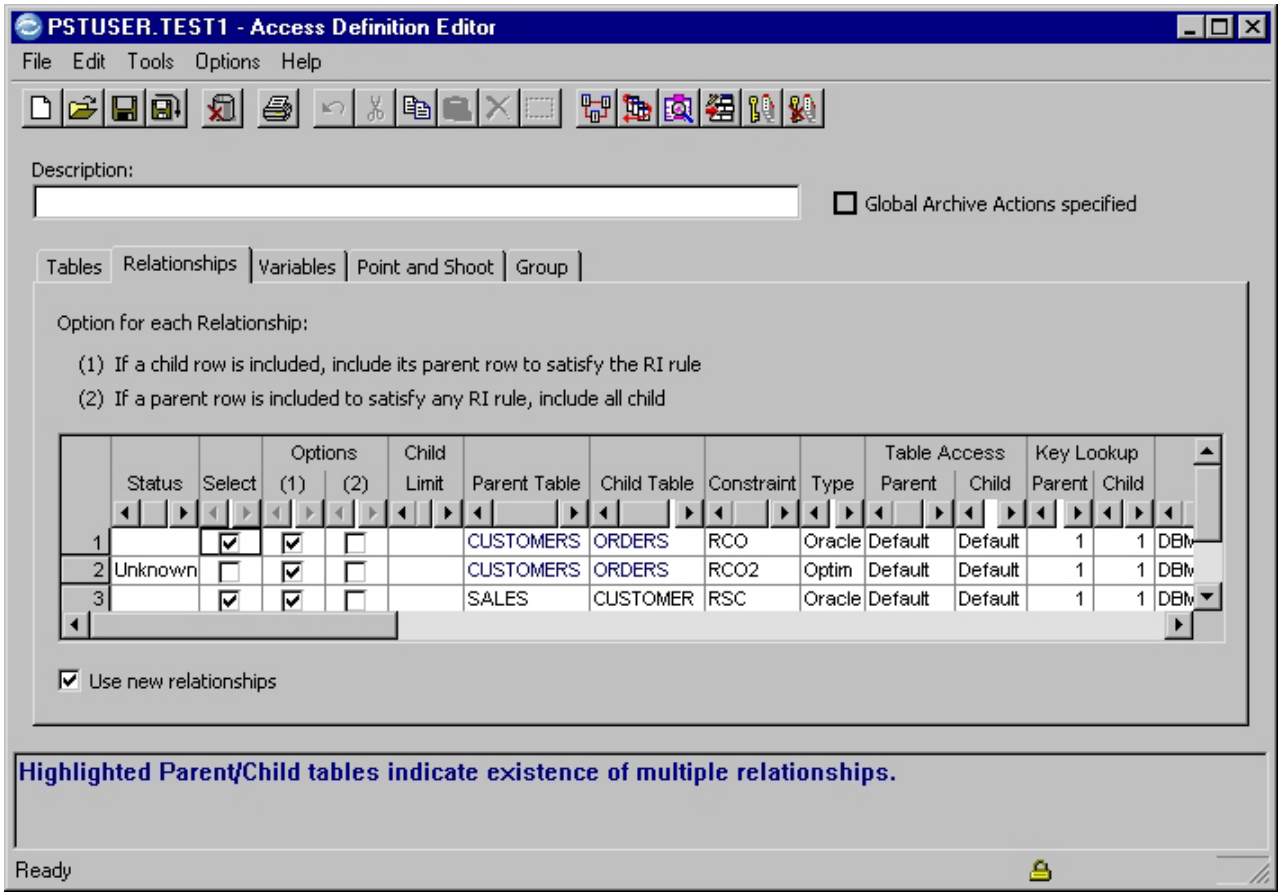
Select All Matching
Use **Select All Matching** to add all table names that match the criteria to the list of table names on the **Tables** tab of the Access Definition Editor.

Display
Use **Display** to display the names of all related tables that match the pattern.

Relationships Tab

Use the **Relationships** tab to define the traversal path for selecting data from tables referenced in an Access Definition. The traversal path determines the sequence in which a process visits each table. If editing or browsing database data, the information on this tab cannot be edited.

The Start Table and any reference tables listed on the **Tables** tab are always included in a process. However, in order to select related data from other tables listed in an Access Definition, you must define the traversal path.



Status

The status of the relationship. Changes to the DB Alias or Creator ID for a table referenced in the Access Definition can affect the status of a relationship. **Status** is populated automatically and cannot be modified.

Note: Two or more relationships that have the same parent and child are considered to be duplicates. The names of duplicate relationships are displayed in a different color. To open the Relationship Editor and change a duplicate relationship, right-click the grid row and select **Open Relationship** from the shortcut menu.

blank Relationship is defined and is not new to the list.

New Relationship is new and status is not Unknown or Ref. The status of a relationship is New the first time the list is displayed after:

- A new relationship between tables on the list is defined.
- A table is added to the list.

When you run a process using the Access Definition, a warning message advises you of New relationships. Although you may proceed despite the warning, it may be prudent to review the relationship usage list.

Unknown
Relationship does not exist. This condition can occur when a change in Default Qualifier causes a new set of tables and relationships to be referenced in the Access Definition. Relationships that have Unknown status are ignored during a process. To remove a relationship in Unknown status, right-click to open the shortcut menu and select Remove or Remove All Unknown.

Ref Table is a reference or look-up table, as specified on the **Tables** tab.

Select

Select the check box to include a relationship in the process. Clear the check box to exclude the relationship. To use relationships with New status automatically, select the **Use new relationships** check box in the lower left corner of the dialog. Right-click the grid column to select or clear the check boxes for all listed relationships.

To edit a relationship, right-click a grid row and select **Open Relationship** from the shortcut menu.

Options

Option (1)
Select this check box to traverse the relationship from child to parent, in order to select a parent row for each child row. This option ensures the relational integrity of the selected data. Option **(1)** is selected by default.

Option (2)
Select this check box to traverse the relationship to select additional child rows for each parent row selected as a result of a traversal from child to parent. When the process follows the path from child to parent, and a parent row is selected, Option **(2)** ensures that additional child rows are selected for that parent. Option **(2)** is cleared by default.

- Options **(1)** and **(2)** are relevant when the Start Table is a child table or when a table has more than one parent table that is referenced by the Access Definition.
- Option **(2)** is relevant only if you traverse a relationship from child to parent. For example, if a process traverses from child to parent [Option **(1)**] and a parent row is selected, selecting Option **(2)** causes the process to select additional child rows for that parent row.
- If you select Option **(2)** for a relationship, consider a Child Limit on the number of child rows to extract for any given relationship.

Note: You can right-click the Options grid column to select commands from the shortcut menu to **Set All...** or **Clear All...** check boxes.

To review the traversal path, select the **Show Steps** option from the **Tools** menu. For details, see "Show Steps" on page 108.

Child Limit

Maximum number of rows from the child table to be selected for a relationship (for example, five orders for each customer).

Parent Table

Name of the parent table in the relationship. You cannot modify this value.

Child Table

Name of the child table in the relationship. You cannot modify this value.

Constraint

Base name for the relationship. You cannot modify this value.

Type

The type of relationship, indicated by DBMS name, Optim, or Generic. You cannot modify this value.

Table Access

Allows you to override the default method (scan or key lookup) for accessing the parent or child table for each relationship. A scan reads all rows in a table at one time; whereas a key lookup locates rows using a WHERE clause to search for primary or foreign key values. (Note that you should override the default method only if the statistical information in the process report indicates the default method is less efficient.) Specify:

Default
Optim determines the best method.

Note: A key lookup is used when a DBMS index is available, and a scan when an index is not available. However, if accessing a significant portion of the table, the default is to scan, even if an index exists.

Force Scan
Force Optim to use a scan.

Force Key Lookup
Force Optim to use a key lookup.

Note: To verify that a DBMS index exists or to create indexes for the parent and child tables in each selected relationship, use the Relationship Index Analysis dialog. (See "Relationship Index Analysis" on page 106.)

To set a method for all parent or child tables at once, right⌂click a **Table Access** cell and select **Set All Default**, **Set All Force Key Lookup**, or **Set All Force Scan** from the shortcut menu.

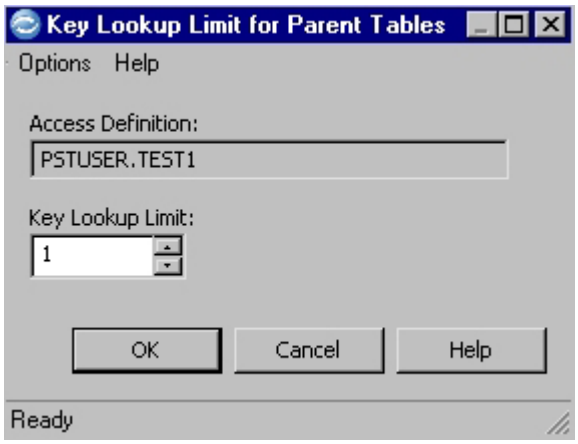
Note: Table Access and **Key Lookup Limit** are displayed automatically if a value differs from the default. To display these settings at all times, select **Advanced Options** from the **Options** menu.

Key Lookup Limit

The maximum number of key lookups performed at one time for a table. Valid values are 1 through 100. By default, Optim looks up one key at a time.

Note that increasing the **Key Lookup Limit** may significantly improve performance. For example, if you specify 5 as the **Key Lookup Limit** and the key has a single column, 5 key values are searched in a single request to the DBMS.

To display the Key Lookup Limit dialog, right-click a **Key Lookup Limit** cell and select the **Set All...** command from the shortcut menu.



Use the Key Lookup Limit dialog to set all values in the parent or child tables at once.

Note: Table Access and **Key Lookup Limit** are displayed automatically if a value differs from the default. To display these settings at all times, select **Advanced Options** from the **Options** menu.

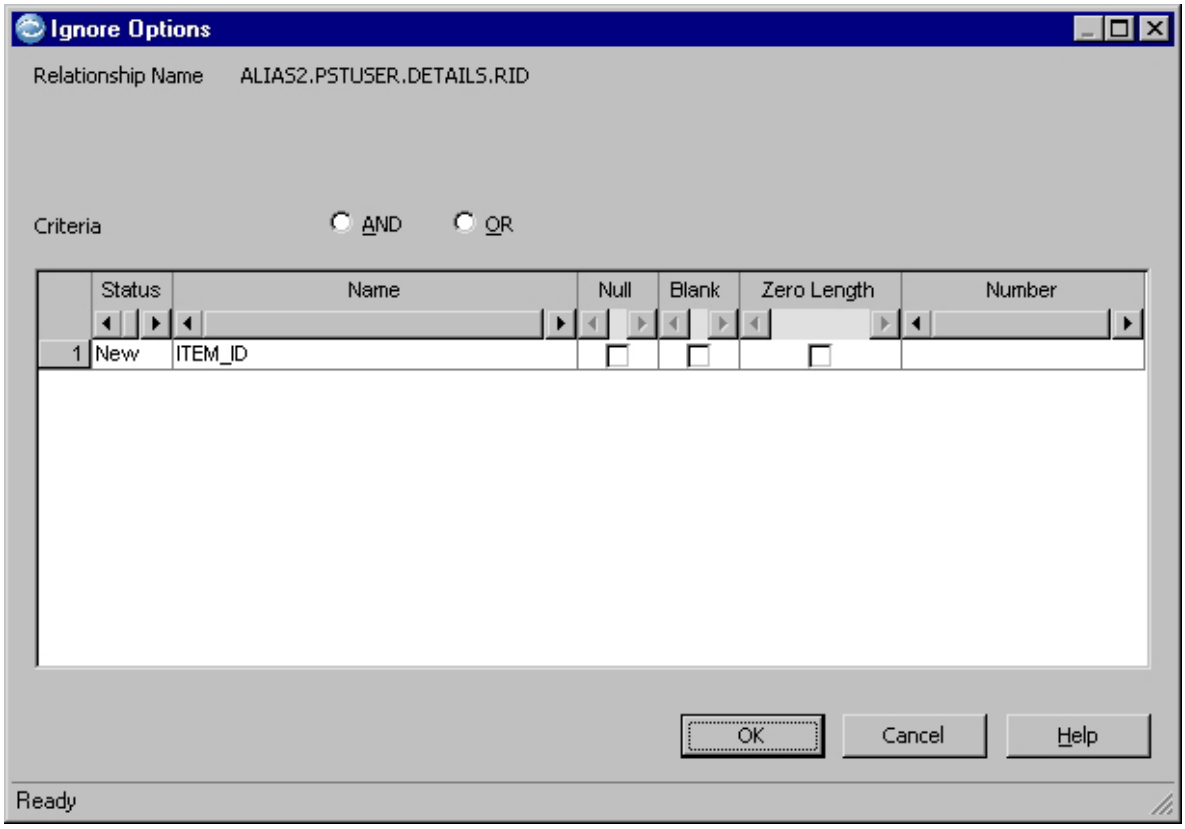
Ignore Empty Relationship

Option to exclude rows from processing if their relationship has an empty value. Optim will not attempt to fetch related rows for this relationship. You can ignore relationships with values equal to null, blank, zero length, any numeric value you choose, or any combination of these options. This option is available if you select **Advanced Options** from the **Options** menu.

To use this option, in the **Ignore Empty Relationship** grid column, right-click the cell for a relationship and this menu displays:

Open Relationship
Opens the Relationship Editor. For details, see “Open the Relationship Editor” on page 214

Change Ignore Options
Opens the Ignore Options dialog.



Criteria

AND

OR Determines whether or not criteria are combined. Selecting AND causes a relationship to be ignored if all relationship columns match the criteria. Selecting OR causes a relationship to be ignored if any columns match the criteria. To select a value, click the button next to it.

Status The status of the relationship. Status is populated automatically and cannot be modified.

Name Name of the foreign key for the relationship. You cannot modify this value.

Null Select the checkbox in this column to ignore null values. This option is valid for all data type columns.

Blank Select the checkbox in this column to ignore blank values. This option is valid for fixed and variable length character columns.

Zero Length
Select the checkbox in this column to ignore zero length values. This option is valid for variable length character columns.

Number
To ignore a numeric value, enter the value in this column. This option is valid for INTEGER type numeric columns.

Set All to → Selected Options
Sets ignore options for all relationships in the Access Definition. If you use this option, it supersedes any ignore option set previously for this Access Definition.

Clear → Clear All
Removes ignore options for a relationship. Use **Clear All** to remove ignore options for all relationships in the Access Definition.

Relationship

Fully qualified name of the relationship. You cannot modify this value. A relationship name is in four parts: *dbalias.creatorid.tablename.constraint*.

dbalias Alias for the database in which the child table is defined (1 to 12 characters).

creatorid
Creator ID assigned to the child table (1 to 64 characters).

tablename
Name of the child table (1 to 64 characters).

constraint
Base name assigned to the relationship (1 to 64 characters).

Use new relationships

This check box is selected by default. Use it to include all relationships with New status in a process.

If you clear this check box, new relationships must be selected manually to be included in a process.

Note: Review the Relationship List each time you edit the Table List, create a relationship for a listed table, or change the Default Qualifier to ensure that the desired relationships and traversal path are selected. If a table is in New status and you do not review the Relationship List, a warning prompt allows you to proceed or abort when you run a process using the Access Definition.

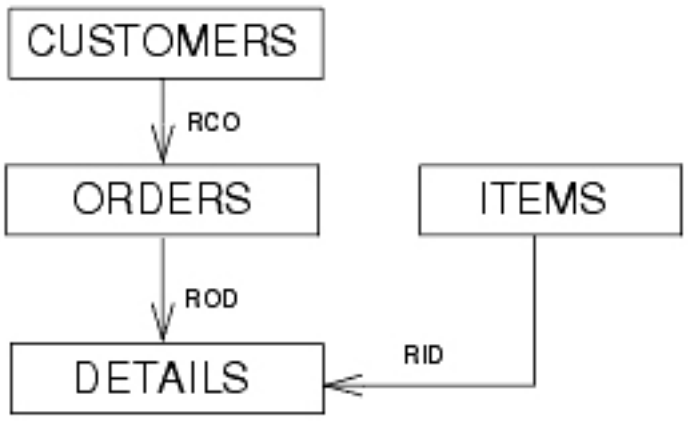
Examples

The following examples use a simple database structure to explain how relationships are traversed using an Access Definition to select a subset of data. This database structure includes four tables: CUSTOMERS, ORDERS, DETAILS, and ITEMS. Based on the relationships between each pair of tables:

- CUSTOMERS is the parent to ORDERS.
- ORDERS and ITEMS are the parents to DETAILS.

Example 1: Basic Traversal Path

The Basic Traversal path is from parent to child as shown next:



The traversal path begins at the Start Table and proceeds through the data model, traversing relationships from parent to child. For example, if the Start Table is CUSTOMERS, the process traverses from ORDERS to DETAILS.

Example 2: Multiple Parent Tables

To select all the ORDERS, DETAILS, and ITEMS rows related to a specific set of CUSTOMERS rows:

1. Specify CUSTOMERS as the Start Table.
2. Specify selection criteria for the desired set of customers.
3. Select Option **(1)** for the relationship RID between ITEMS and DETAILS to traverse from child to parent.

The process selects:

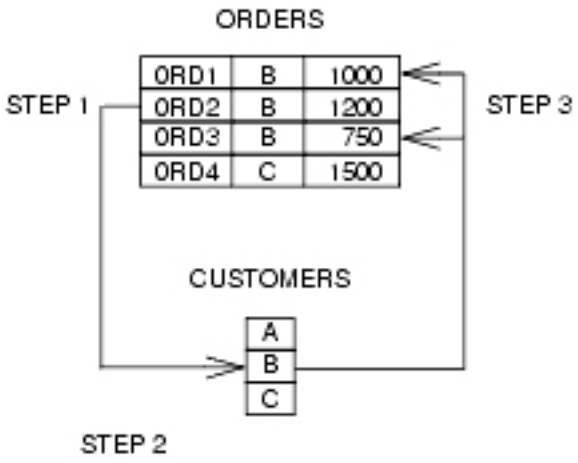
1. Rows from the CUSTOMERS table.
2. Related child rows in the ORDERS table to satisfy the relationship RCO between CUSTOMERS and ORDERS.
3. Related child rows in the DETAILS table to satisfy the relationship ROD between ORDERS and DETAILS.
4. Related parent rows from the ITEMS table for every child row selected from the DETAILS table. This step satisfies the relationship RID between ITEMS and DETAILS, according to Option **(1)**.

Example 3: Child as a Start Table

To select all ORDERS rows for CUSTOMERS that placed specific ORDERS:

1. Specify ORDERS as the Start Table.
2. Use Point and Shoot to select the specific set of ORDERS rows.
3. Select Option **(1)** and Option **(2)** for the relationship RCO between CUSTOMERS and ORDERS. The process traverses from ORDERS (child) to CUSTOMERS (parent) and back to ORDERS (child) to select all orders for each selected customer.

In this example, use Point and Shoot with the ORDERS table as the Start Table to select ORD2. The related CUSTOMERS row B has three related ORDERS rows, ORD1, ORD2, and ORD3. The rows are selected in the sequence shown:



STEPS

1. ORDERS row ORD2 is selected using Point and Shoot.

2. The CUSTOMERS row B is selected because of Option **(1)**.
3. ORDERS rows ORD1 and ORD3 are selected because of Option **(2)**.

Because you selected Option **(2)**, the process selects additional ORDERS table rows for every parent row selected from the CUSTOMERS table.

Note: Any rows selected from a table because of Option **(2)** must satisfy selection criteria specified for that table and are subject to statistical controls.

Example 4: Multiple Relationships

In this example, orders are shipped only when all items are available. If one ordered item is out of stock, the order is not shipped to the customer. To identify orders that are outstanding because of an out-of-stock item, the set of selected data must include full details on all orders.

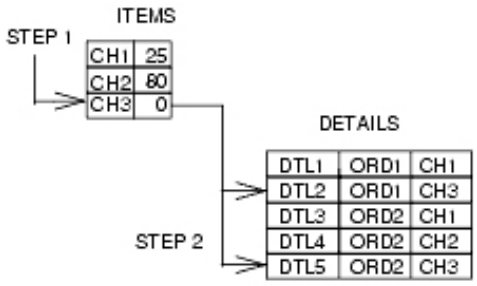
To obtain complete details, select data from the CUSTOMERS, ORDERS, DETAILS, and ITEMS tables:

1. Specify the ITEMS table as the Start Table.
2. Specify selection criteria to select out-of-stock-ITEMS rows. (Select items with a quantity of 0.)
3. Select Option **(1)** for all relationships.
4. Select Option **(2)** for the relationship ROD between ORDERS and DETAILS.

The process begins as described in the next illustrations.

After the specified ITEMS rows are selected, the related DETAILS are selected by traversing the relationship RID from parent to child.

ITEMS to DETAILS



STEPS

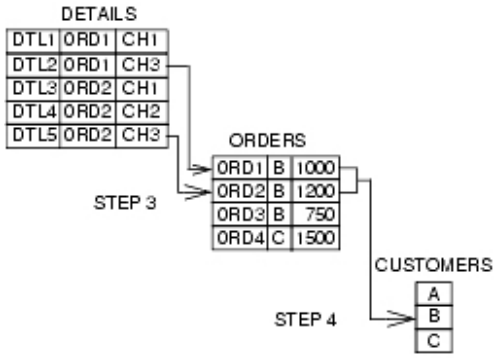
Step 1–The ITEMS row with a quantity of zero is selected based on the selection criteria.

Step 2–The DETAILS rows containing the item are selected because of the parent to child traversal of relationship RID.

The process selects ORDERS related to DETAILS, followed by CUSTOMERS related to ORDERS.

DETAILS to ORDERS

ORDERS to CUSTOMERS

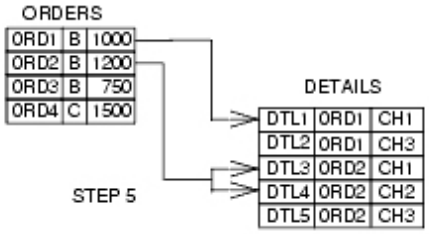


Step 3–The ORDERS that are related to these DETAILS rows are selected by traversing the relationship ROD from child to parent because Option **(1)** is selected.

Step 4–The CUSTOMERS that placed these ORDERS are selected by traversing the relationship RCO from child to parent because Option **(1)** is selected.

The process selects additional DETAILS related to these ORDERS.

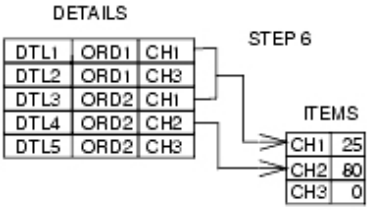
ORDERS to DETAILS



Step 5–The additional DETAILS that are part of the ORDERS are selected because Option **(2)** is selected for the relationship between ORDERS and DETAILS.

The process selects ITEMS for the additional DETAILS.

DETAILS to ITEMS



Step 6–The ITEMS for the additional DETAILS are selected because Option **(1)** is selected for the relationship between ITEMS and DETAILS.

Example 4 Completes

The process selects only the ORDERS for selected ITEMS. The process does not select additional ORDERS for selected CUSTOMERS because you did not select Option **(2)** for the relationship RCO between CUSTOMERS and ORDERS.

Example 5: Multiple Children

Some tables are parent to more than one child table. Options **(1)** and **(2)** apply only to each pair or tables that share a relationship. For example, if the process selects a row in the CUSTOMERS table because you selected Option **(1)** for the relationship between CUSTOMERS and ORDERS, then Option **(2)** applies only to related rows from the ORDERS table. To select data from other child tables, you must select Option **(2)** for those relationships.

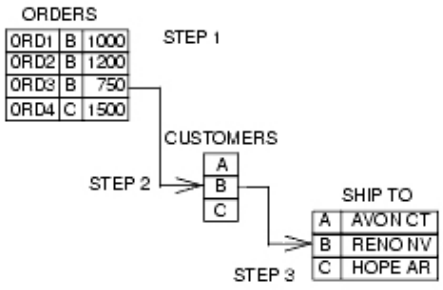
In this example, the SHIP_TO and ORDERS tables are children of the CUSTOMERS table. To obtain the shipping information for specific orders:

1. Specify the ORDERS table as the Start Table.
2. Specify the selection criteria for the desired set of ORDERS.

3. Select Option **(1)** for the relationship RCO between CUSTOMERS and ORDERS.
4. Select Options **(1)** and **(2)** for the relationship RCST between CUSTOMERS and SHIP_TO.

The process begins as described in the next illustration:

ORDERS to CUSTOMERS to SHIP_TO



STEPS

1. A single ORDERS row is selected based on the selection criteria.
2. The CUSTOMERS row is selected because Option **(1)** is selected for the relationship RCO between CUSTOMERS and ORDERS.
3. The SHIP_TO row is selected because Option **(2)** is selected for the relationship RCST between CUSTOMERS and SHIP_TO.

Example 6: Traversal Cycles

You can direct a process to cycle through (revisit) tables to select data. These traversal cycles depend on the options you specify for each relationship.

Extending Example 4 (Multiple Relationships) to select all related data about orders for an out-of-stock item, a traversal cycle results by selecting all items that have a quantity of zero. To obtain a complete set of orders:

1. Specify ITEMS as the Start Table.
2. Clear the check box for the relationship between CUSTOMERS and ORDERS (The CUSTOMERS table is not needed in this example.)
3. Select Option **(1)** for the relationship ROD between ORDERS and DETAILS.
4. Select Option **(1)** for the relationship RID between ITEMS and DETAILS.
5. Select Option **(2)** for the relationship ROD between ORDERS and DETAILS.

The process performs the following steps to complete a traversal cycle:

1. Selects the ITEMS that have a quantity of zero.
2. Selects DETAILS related to those ITEMS by traversing the relationship RID from parent to child.
3. Selects ORDERS related to the DETAILS by traversing the relationship ROD from child to parent. (Option **(1)** for that relationship is Yes.)
4. Selects additional DETAILS for those ORDERS by traversing the relationship ROD from parent to child. (Option **(2)** for that relationship is Yes.)
5. Selects ITEMS related to the additional DETAILS by traversing the relationship RID from child to parent. (Option **(1)** for that relationship is Yes.)

Multiple Traversal Cycles

In Example 6, the process starts and ends at the ITEMS table to complete one cycle, but a process can also include multiple traversal cycles.

Example 6 can be extended to show multiple, although unlikely, traversal cycles. After selecting ITEMS rows for additional items in Step 5, it is necessary to select ORDERS and DETAILS rows for these ITEMS. This is done by selecting Option **(2)** for the relationship RID.

After Step 5 in the example, the process traverses the relationship RID from parent to child to select the remaining DETAILS rows related to these additional ITEMS. Then the process repeats Steps 3, 4, and 5 to select additional ORDERS, the DETAILS rows related to those ORDERS, the ITEMS rows related to those DETAILS, and so on.

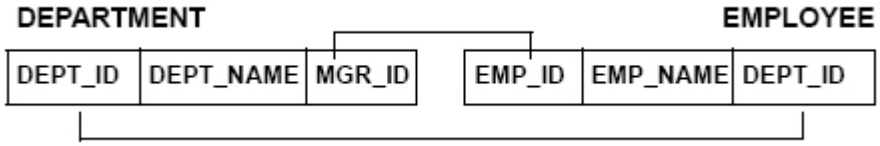
Example: Selection of Referential Cycles

You can select data from tables that are related to each other by referential cycles. In a referential cycle, the process starts at one table and returns to it after traversing one or more relationships. (In contrast, for many databases, a process proceeds up or down through the hierarchy or the network and never returns to the Start Table.)

The following examples use two tables — DEPARTMENT and EMPLOYEE. The tables are defined as:

DEPARTMENT			**EMPLOYEE**		
DEPT_ID	DEPT_NAME	MGR_ID	EMP_ID	EMP_NAME	DEPT_ID

The DEPT_ID is the primary key of the DEPARTMENT table; the EMP_ID is the primary key of the EMPLOYEE table. The relationships between the DEPARTMENT and EMPLOYEE tables are shown next:



The relationships used in the following examples are defined as:

Relationship	**Parent Table.Column**	**Child Table.Column**
MANAGER	EMPLOYEE.EMP_ID	DEPARTMENT.MGR_ID
MEMBER	DEPARTMENT.DEPT_ID	EMPLOYEE.DEPT_ID

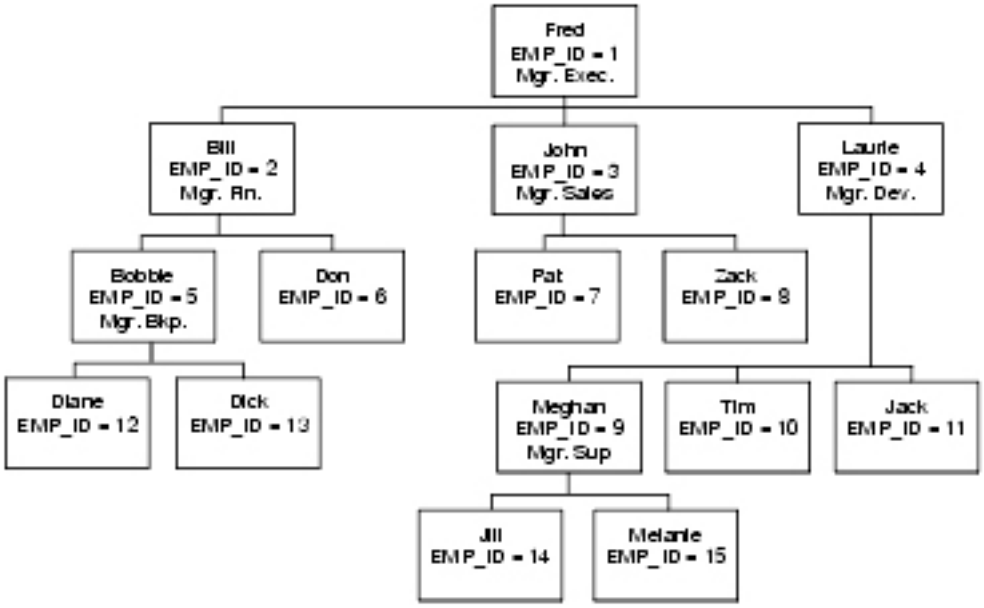
The data in each table is as follows:

DEPARTMENT		
DEPT_ID	DEPT_NAME	MGR_ID
A	Executive	1
B	Finance	2
C	Sales	3
D	Development	4
E	Bookkeeping	5
F	Support	9

EMPLOYEE		
EMP_ID	EMP_NAME	DEPT_ID
1	Fred	
2	Bill	A
3	John	A
4	Laurie	A
5	Bobbie	B
6	Don	B
7	Pat	C
8	Zack	C
9	Meghan	D
10	Tim	D
11	Jack	D
12	Diane	E
13	Dick	E

	14	Jill	F
	15	Melanie	F

The following is a sample company organization chart:



In the next examples, a sample query demonstrates how to handle cyclic relationships by selecting Options **(1)** and **(2)** for different relationships. Changing these options affects the set of data selected from each table.

Example 7

To select the names of all employees in the Development Department, including employees that are members of subordinate departments:

1. Specify the DEPARTMENT table as the Start Table.
2. Use Point and Shoot to select the row in the DEPARTMENT table, where DEPT_ID = D.
3. Do not select Options **(1)** and **(2)** for relationships because the process always follows the existing relationships using the basic traversal path from parent to child.

The process performs these steps:

1. Selects the row containing the Development Department, Dept_ID='D', from the DEPARTMENT table.
2. Using the relationship MEMBER, search the EMPLOYEES table for all employees in the Development Department, EMPLOYEE.DEPT_ID='D'.
   Three EMPLOYEE rows are extracted.
   EMP_ID = 9
   EMP_ID = 10
   EMP_ID = 11
3. Use the relationship MANAGER to search the DEPARTMENT table to determine if any of the selected employees are managers. The employee with EMP_ID = 9, Meghan, is a manager of the Support Department, DEPT_ID = 'F'. Select this row from the DEPARTMENT table.
4. Use the relationship MEMBER to search the EMPLOYEE table to select all employees that belong to the Support Department, DEPT_ID = 'F'.

Two EMPLOYEE rows are selected.
EMP_ID = 14
EMP_ID = 15

5. Use the relationship MANAGER to search the DEPARTMENT table to determine if either of the two selected rows in step 4 are managers. They are not, and the process stops.

Example 7 Completes

When the process completes, two rows are selected from the DEPARTMENT table and five rows are selected from the EMPLOYEE table:

DEPARTMENT	EMPLOYEE
DEPT_ID='D'	EMP_ID = 9
DEPT_ID='F'	EMP_ID = 10
	EMP_ID = 11
	EMP_ID = 14
	EMP_ID = 15

Example 8

To select the manager of the Development Department in addition to all employees in the Development Department and subordinate departments (Example 7):

1. Specify the DEPARTMENT table as the Start Table.
2. Use Point and Shoot to select the row in the DEPARTMENT table, where DEPT_ID = D.
3. Select the relationship MANAGER between EMPLOYEE (parent) and DEPARTMENT (child).
4. Select Option **(1)** for the relationship MANAGER. This option selects the manager of the Development Department by traversing from child to parent.

The process follows the steps described for Example 7. In addition, Step 4 traverses the relationship MANAGER from DEPARTMENT (child) to EMPLOYEE (parent), based on Option **(1)**. As a result, the process selects MGR_ID = 4 to identify the manager of the Development Department (Laurie) and selects a row from the EMPLOYEE table, EMP_ID = 4 (Laurie).

Example 9

To select any other departments this manager manages in addition to all employees in the Development Department and the manager of the Development Department:

1. Specify the DEPARTMENT table as the Start Table.
2. Use Point and Shoot to select the row in the DEPARTMENT table, where DEPT_ID = D.
3. Select the relationship MANAGER between EMPLOYEE (parent) and DEPARTMENT (child).
4. Select Option **(1)** for the relationship MANAGER. This option selects the manager of the Development Department by traversing from child to parent.
5. Select Option **(2)** for the relationship MANAGER. This option selects other departments headed by the manager of the Development Department by traversing from parent to child.

The process follows the steps described for Example 8. In addition, after Laurie is selected from the EMPLOYEE table, the process follows the MANAGER relationship from EMPLOYEE (parent) to DEPARTMENT (child). The process traverses the DEPARTMENT table to see if Laurie is manager of any other department. Based on the employee hierarchy for these examples, no additional rows are selected. Laurie manages only the Development Department.

Example 10

In addition to all employees in the Development Department, the manager of the Development Department, and any other departments under that manager, extend the example to select any other departments of which the manager of the Development Department is a member. To do this:

1. Specify the DEPARTMENT table as the Start Table.
2. Use Point and Shoot to select the row in the DEPARTMENT table, where DEPT_ID = D.
3. Select the relationship MANAGER between EMPLOYEE (parent) and DEPARTMENT (child).
4. Select Option **(1)** for the relationship MANAGER. This option selects the manager of the Development Department by traversing from child to parent.
5. Select Option **(2)** for the relationship MANAGER. This option selects other departments under the manager of the Development Department by traversing from parent to child.
6. Select the relationship MEMBER between DEPARTMENT (parent) and EMPLOYEE (child).
7. Select Option **(1)** for the relationship MEMBER. This option selects other departments to which the manager of the Development Department is a member by traversing from child to parent.

In this example, after Laurie is selected from the EMPLOYEE table, the process follows the MEMBER relationship from EMPLOYEE (child) to DEPARTMENT (parent) based on Option **(1)**. The process traverses the DEPARTMENT table to see if Laurie is a member of any other department. Laurie is a member of the Executive Department. The process selects an additional row from the DEPARTMENT table, DEPT_ID = A.

In addition, because you selected Option **(1)** for the MANAGER relationship, the process traverses from DEPARTMENT (child) to EMPLOYEE (parent) to select an additional row from the EMPLOYEE table, EMP_ID = 1. (Fred is the manager of the Executive Department.)

Example 11

In this example, select only employees that are members of the Development Department. Unlike Example 7, do not select employees in subordinate departments. This process disables the referential integrity (RI) cycle:

1. Specify the DEPARTMENT table as the Start Table.
2. Use Point and Shoot to select the row in the DEPARTMENT table, where DEPT_ID = D.
3. Clear the check box for the relationship MANAGER between EMPLOYEE (parent) and DEPARTMENT (child).
4. Select the relationship MEMBER between DEPARTMENT (parent) and EMPLOYEE (child), which represents the basic traversal cycle.
5. Clear the Option **(1)** check box for the relationship MEMBER. This option is not necessary.

In this example, Step 3 disables the RI cycle. The DEPARTMENT row for the Development Department is selected. Then, using the relationship MEMBER, the process traverses from parent to child to select the three employees in the department. Because the MANAGER relationship is unselected, it is not traversed. The process is complete.

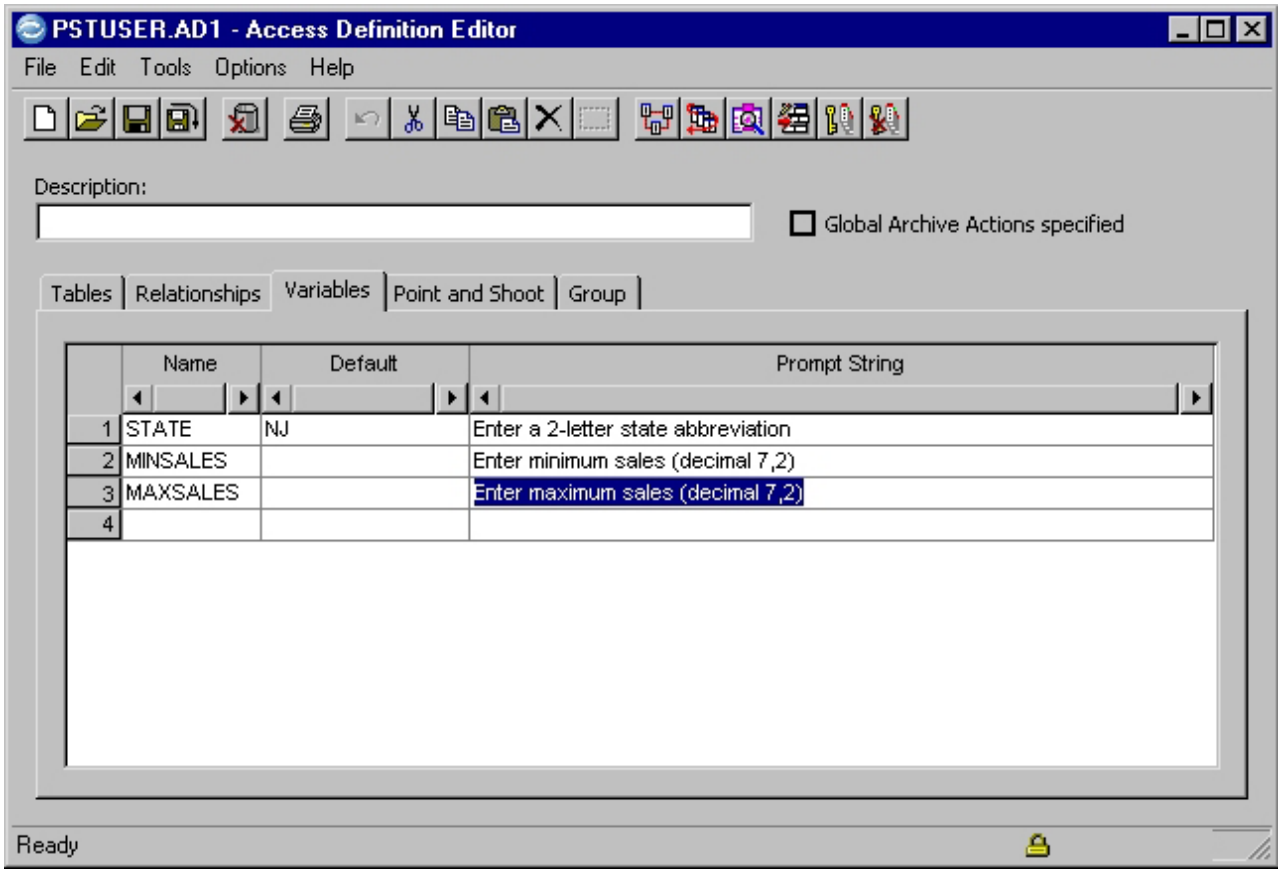
Variables Tab

Use the **Variables** tab to assign substitution variables for selection criteria or SQL statements in an Access Definition. Additionally, you can use any defined variables when creating Extract or Restore Archive Actions.

For detailed information about selection criteria, SQL, or Archive Actions, refer to "Table Specifications" on page 76.

By assigning variables, you provide values for the variables each time the Access Definition is processed. As an option, you can provide default values for substitution variables. The variables are saved with the Access Definition.

Note: You can also specify or override variables (programatically or in batch) referenced in an Archive, Extract, or Restore Request using the command line interface. For detailed information about the command line interface, refer to the *Archive User Manual* or the *Move User Manual* .



Name

Enter a name for the variable (1 to 12 characters). The name is embedded in the selection criteria or SQL. Logical naming conventions help identify variables.

Default

Enter an optional default value for the variable to be used when no value is specified for the variable at run time.

Default values must be of the appropriate data type and size for the column and must conform to SQL syntax. For example, assume a variable name is **ST** (state), the variable delimiter is a colon (**:**), and the column requires character data.

- If you use the variable with single quotes in the selection criteria, you must specify the value without single quotes:

Selection Criteria	Value
= ':ST'	CA

- If you use the variable without single quotes in the selection criteria, you must specify the value with single quotes:

Selection Criteria	Value
= :ST	'CA'

Note: Default values are not validated until run time. If a value is the incorrect data type or size for the column or does not conform to SQL syntax, processing errors may result.

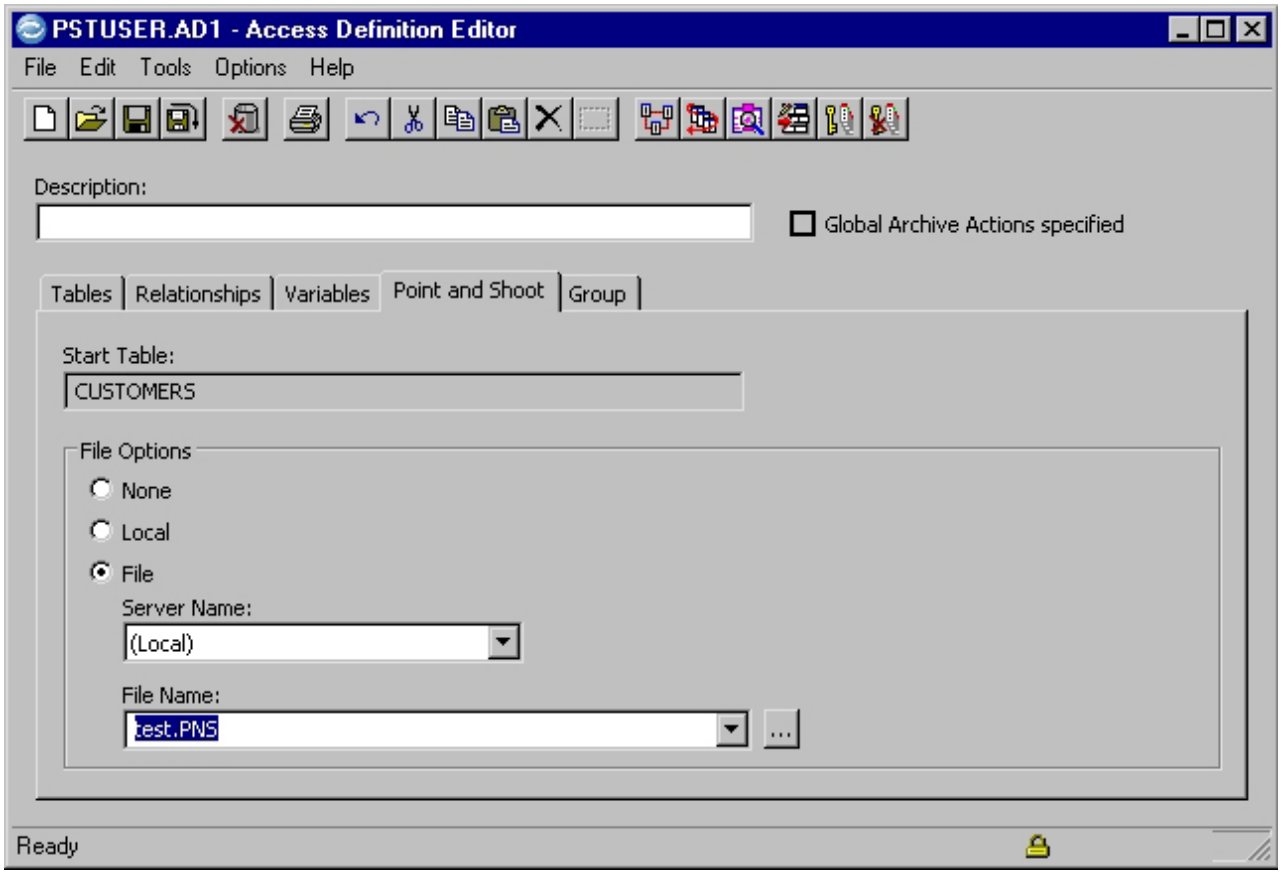
Prompt String

Enter the text that prompts for the variable value at run time. Type the prompt string exactly as you want it to appear in the process request dialog (up to 50 characters). This prompt is displayed before you execute the request.

Point and Shoot Tab

Use the **Point and Shoot** tab to select Point and Shoot list options for the Access Definition. A Point and Shoot list is a list of primary key values that is used as criteria to select individual Start Table rows.

Note: Settings on this tab do not apply when editing or browsing database tables.



Start Table

Name of the table selected as the Start Table on the **Tables** tab.

File Options

None A Point and Shoot list is not referenced in the Access Definition.

Local The Point and Shoot list is stored with the Access Definition. A Local list is unavailable to other process requests or Access Definitions. However, you can save a Local Point and Shoot list as a named Point and Shoot list.

File The Point and Shoot list is named and saved as a file that can be referenced by other Access Definitions and process requests, and shared with other users. A Point and Shoot file is stored in the directory you specify. To use an existing file or to create a new, named Point and Shoot file, select **File**, select the server, and type the file name.

Server Name
Machine from which the Point and Shoot file is accessed. Select **Local** if the file is accessed from the workstation or click the down arrow to select the Optim Server from which the file is accessed. If you are editing the Access Definition from an Archive or Extract Request, the Server specification on the editor applies. Server is disabled on the Access Definition Editor.

File Name
The default **.pns** extension is added to the file name, by default.

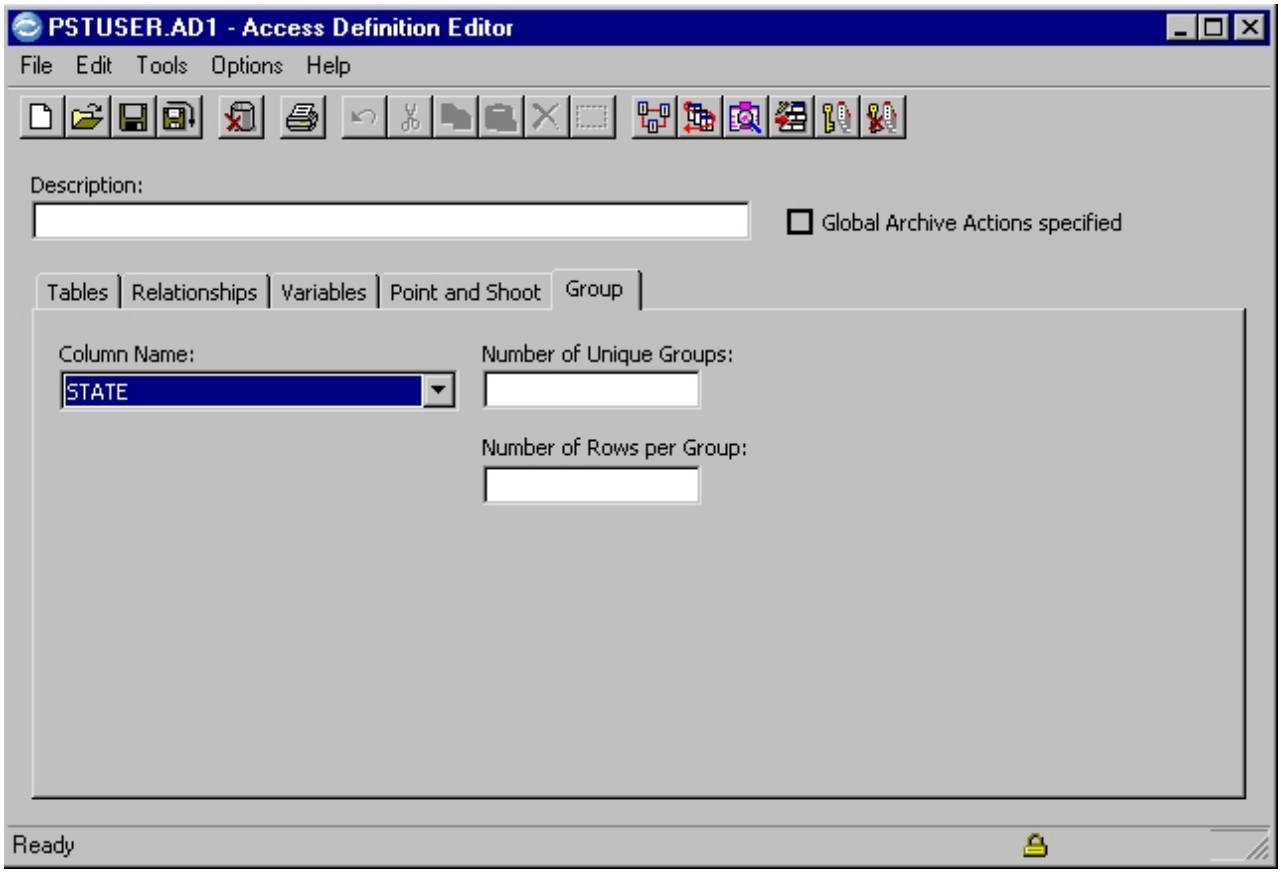
To create a new **Local** Point and Shoot list or to edit a named Point and Shoot file, select **Edit Point and Shoot** from the **Tools** menu to display the Point and Shoot Editor. See "Edit Point and Shoot List" on page 109.

Note: You can override Point and Shoot settings in the Access Definition with settings in an Archive or Extract Request.

Group Tab

Use the **Group** tab for group selection, which extracts a number of rows based on values in a particular column in the Start Table.

Note: Group selection is typically used in an Extract Process to select data for a test database. The **Group** tab is not available and the settings on it do not apply when archiving, editing, or browsing database tables.



Column Name

Name of the Start Table column used to select the data to extract. For example, you can extract a set of customer data based on values in the STATE column in the CUSTOMERS table. Click the arrow to select a column name.

Number of Unique Groups

Number of groups to extract. Rows in each group have the same value in the selected Start Table column, but the value is different for each group. To select a group for each unique value, leave the box blank.

Number of Rows per Group

Maximum number of rows to extract for each unique value in the selected Start Table column. To select all rows per group, leave the box blank.

Note: For group selection, you must specify a **Number of Unique Groups**, a **Number of Rows per Group**, or both.

Examples

As an example of group selection, you can extract a set of data based on values in the STATE column in the CUSTOMERS table:

	Number of Unique Groups	Number of Rows per Group
To extract all rows of customer data from any 10 states.	10	*blank*

To extract 50 rows of customer data from all states.	*blank*	50
To extract 50 rows of customer data from any 10 states.	10	50

You can combine selection criteria or an SQL WHERE clause with Group options. In this example, you can specify selection criteria to target up to 10 specific states.

Note: When criteria are used in combination with Group Selection parameters, the criteria are applied first with Group Selection parameters applied to the result. In the example, if you provided selection criteria for three states, three groups would be selected. If, however, you provided criteria for 11 states, only 10 groups would be selected.

For details on selection criteria, see "Table Specifications."

Table Specifications

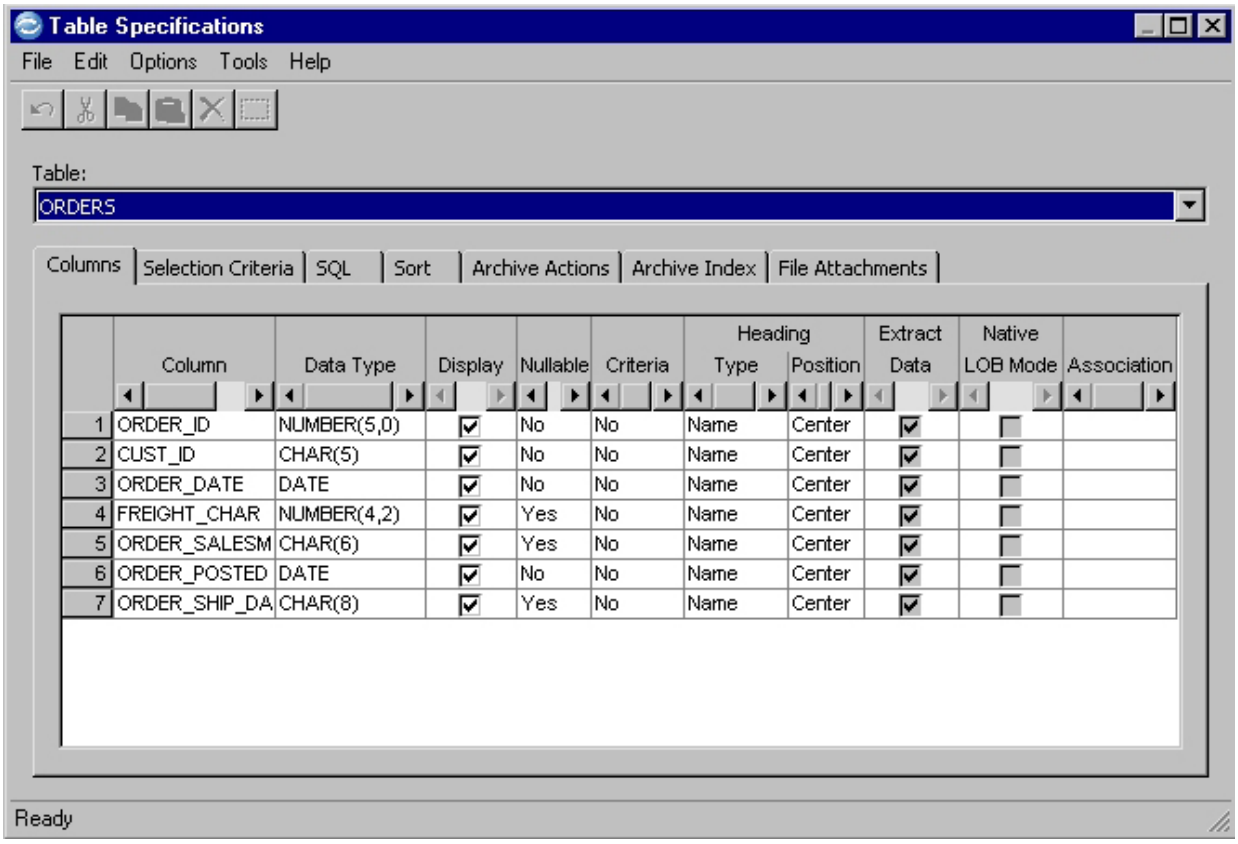
Table specifications provide display and criteria parameters for data. Display parameters apply when editing or browsing data, and when selecting rows for a Point and Shoot list. Criteria apply when selecting rows for display, or archiving, extracting, or restoring data.

You can:

- Include or exclude table columns from a data display.
- Display or select specific data by assigning selection criteria.
- Create SQL WHERE clauses to define complex selection criteria.
- Arrange displayed rows in ascending or descending order according to values in one or more specified columns.

To add table specifications to an Access Definition:

1. Position the pointer on the desired table name in the Table/View grid in the Access Definition Editor.
2. Right-click to open the shortcut menu.
3. Select a choice for **Table Specifications** from the shortcut menu. The choices correspond to the tabs on the Table Specifications dialog.



Initially, the **Table** box shows the name of the table you selected on the **Access Definition Editor** table List. To edit table specifications for another table referenced in the Access Definition, click the down arrow and select a table name from the list.

Columns Tab

The **Columns** tab displays information about the table and enables you to choose display options and other criteria, as follows:

Column

Name of each column in the specified table. To rearrange the order in which the columns are displayed in the Table Editor or Point and Shoot Editor, drag the grid row number. You cannot modify column names.

Data Type

Data type and length for each column. You cannot modify this value.

Display

Check box to identify columns to display in the Table Editor or the Point and Shoot Editor. At least one column per table must be selected for display. By default, all columns are included for display. Clear the check box to exclude a column from the display. This option has no effect on a process using the Access Definition.

Nullable

Null indicator. The value is Yes if the column is defined to accept a NULL value and No, if it is not. You cannot modify this value.

Criteria

Criteria indicator. This value is Yes if selection criteria are specified for a column and No if no selection criteria are present. You cannot modify this value. (Select the **Selection Criteria** tab or **SQL** tab of the Table Specifications dialog to edit or define criteria.)

Heading Type

Type of heading for a column. To set the heading type:

- For an individual column, left-click the Type grid cell and choose **Name** or **Label**.
- For all columns, right-click the Type grid column and choose **Set All Name** or **Set All Label**.

Heading Position

Justification for column headings. To change the justification:

- For an individual column, left-click the Position grid cell and choose **Left**, **Center**, or **Right**.
- For all columns, right-click the Position grid column and choose **Set All Left**, **Set All Center**, or **Set All Right**.

Extract Data

All check boxes in the Extract Data column are selected by default. When you clear a check box, data in the corresponding column is not extracted when the Access Definition is used for an Extract Process.

Note: Only columns with CLOB or BLOB data types are permitted to be excluded. Columns are NOT excluded when the Access Definition is used in an Archive Process.

Native LOB Mode

Select the check box to display LOBs in the Table Editor as normal LOB data. Clear the check box to display LOBs as VARCHAR or VARBIN data. If you clear this check box, a limitation on the amount of data retrieved from the LOB applies. The default setting and size limitation is set on the **Display** tab of Personal Options.

Association

Use this column to associate a LOB column with the application required to view or edit the LOB data, in one of two ways:

- Type the file name extension in the column that corresponds to the type of LOB (for example, type the extension *.doc* to associate a LOB Word document with Microsoft Word).

  OR

- If the table contains a reference column used to identify the LOB data, you can enter the name of the reference column in the **Association** column. The first three characters in the reference column are used as the file name extension for the LOB data in the corresponding row of the LOB data column. The reference column must be a character-type column. Click the **Association** column to display a drop-down list of the names of character-type columns in the table, and select the reference column name.

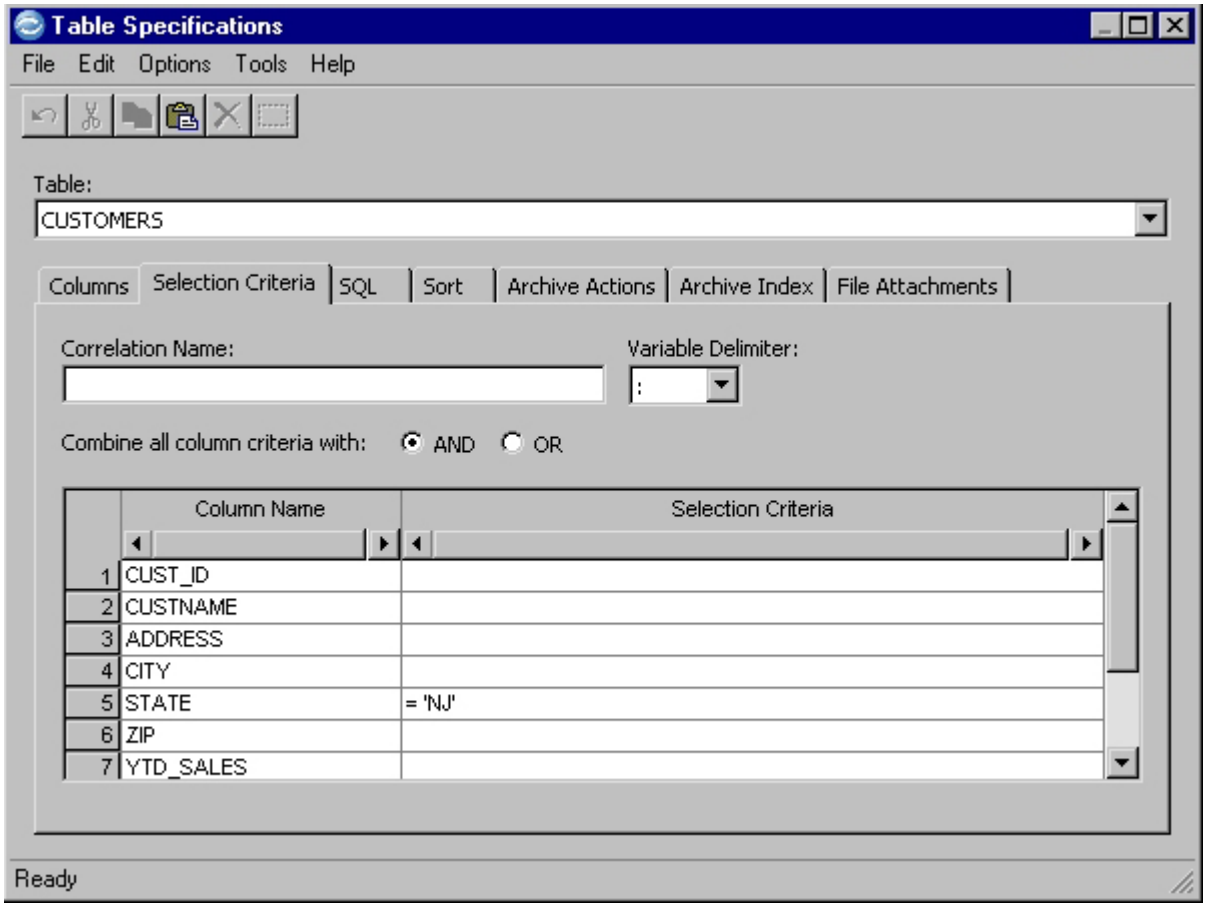
Column Specifications Identified

The columns icon  in a Table Specifications grid cell on the **Access Definition Editor** table List indicates that columns are rearranged, headings or LOB options are specified for data displays, or LOB column associations are specified.

Selection Criteria Tab

Use the **Selection Criteria** tab to provide selection criteria for one or more columns in any table in the Access Definition Table List. The criteria is used to select rows to display, archive, or extract.

Note: If you provide selection criteria and an SQL WHERE clause for a table, the specifications are logically ANDed (rows must meet both conditions). If a Point and Shoot list is also used, it is logically ORed with the other criteria.



Correlation Name

Abbreviation for the name in the **Table** box. Enter an easily typed substitute for the fully qualified table name to use in selection criteria or an SQL WHERE clause.

Note: Changes to the Correlation Name apply on the **Selection Criteria** and **SQL** tabs.

Variable Delimiter

Character required to identify substitution variables in selection criteria and SQL WHERE clauses. To change the delimiter, click the down arrow.

Note: Optim automatically revises selection criteria and SQL WHERE clauses to reflect the change.

Combine all column criteria with

Option for combining criteria for multiple columns:

AND A row must match selection criteria for all columns. For example:

```
CUSTNAME > 'M' AND STATE = 'NJ'
```

OR A row must match selection criteria for one column. For example:

```
CUSTNAME > 'M' OR STATE = 'NJ'
```

Column Name

Name of each column. You can rearrange the order in which the columns are displayed in the Table Editor or Point and Shoot Editor by dragging the grid row number. You cannot modify column names and you cannot search an SQL Variant column.

Selection Criteria

Selection criteria for any column.

Note: When a relationship is traversed from child to parent, any selection criteria for the parent table are ignored.

Use an appropriate operator and value or substitution variable specification. Selection criteria must conform to SQL syntax and include relational or logical operators. Logical operators and syntax vary among DBMSs. Refer to the appropriate DBMS documentation for information. Lists of commonly used operators, appropriate to the current DBMS, are provided on the **SQL** tab.

For information on defining substitution variables, see “Variables Tab” on page 71.

Validate selection criteria by clicking outside the grid. If the DBMS returns an error on an SQL Prepare statement, that message is displayed.

Note: To close this dialog, you must correct any errors.

Remove selection criteria by right-clicking the **Selection Criteria** cell to open a shortcut menu and selecting **Clear**, **Remove** or **Remove All**. You may also overtype with blanks or use the Delete or Backspace key.

Date Criteria

A unique operator allows you to select data on the basis of values in a DATE column. The syntax for this operator is:

BEFORE (*n*D *n*W *n*M *n*Y)

Use the D, W, M, and Y arguments in any combination to indicate the number of days, weeks, months, or years subtracted from the date at runtime. If no arguments are specified, the current date is used. Rows with a date older than the calculated date are extracted or archived. The *n* multiplier is an integer and can optionally be preceded by + or -.

Note: You can analyze the effect of selection or date criteria and confirm that it works as expected by selecting the **Access Definition Editor**. If the table for which you are defining criteria is not the Start

Table, temporarily specify it as the Start Table and select **Edit Point and Shoot List** from the **Tools** menu. Optim displays data that corresponds to the selection criteria. If no data is displayed, the syntax for the selection criteria may be incorrect for the DBMS.

Selection Criteria Identified

A selection criteria icon  in the Table Specifications column on the Access Definition Editor Table Listindicates that selection criteria are specified for the table.

Examples

The following examples use operators and syntax that may not be valid for all DBMSs. The functions demonstrated in the examples are universal, however.

Example 1

To obtain data for all customers with names that begin with the letter M, from the state of New Jersey, specify:

CUSTNAME	**LIKE 'M%'**
STATE	**= 'NJ'**

This example uses explicit values for each column. However, you can also use substitution variables.

For example, to obtain data for the same customers using a variable (ALPHA) for the alphabetic range and a variable (ST) for the state, specify:

CUSTNAME	**LIKE :ALPHA**
STATE	**= :ST**

Select **AND** to include all customers that satisfy both conditions. Select **OR** to include all customers that satisfy either condition.

Example 2

To obtain data for all customers with names that begin with the letters M, N, O, P and Q from the states of California, Arizona and New Mexico, specify:

CUSTNAME	**BETWEEN 'M' AND 'Q'**
STATE	**IN ('CA','AZ','NM')**

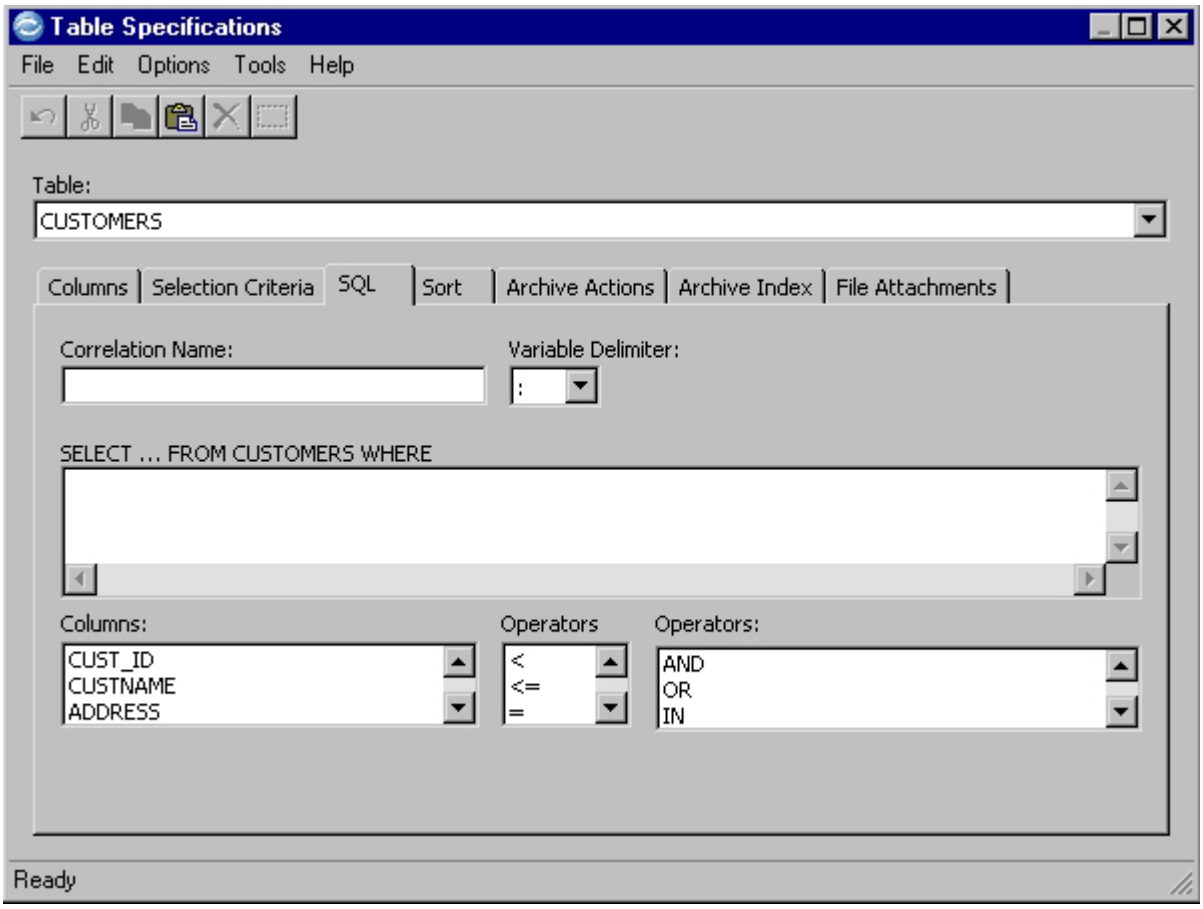
As in the previous example, select **AND** to include all customers that satisfy both conditions. Select **OR** to include all customers that satisfy either condition.

Note: Values are validated at run time or you can display data in the Point and Shoot Editor to confirm selection criteria. Processing errors may occur if values are not the correct data type or size, or if the specifications do not match the requirements of the selection criteria SQL.

SQL Tab

Use the **SQL** tab to create an SQL WHERE clause to handle selection criteria that are more complex than can be defined on the **Selection Criteria** tab. For example, to select the desired set of data for a table, you may need a combination of AND and OR logical operators.

Note: If you specify an SQL WHERE clause and selection criteria for a table, the specifications are logically ANDed (rows must meet both conditions). If a Point and Shoot list is also used, it is logically ORed with the other criteria.



Correlation Name

Abbreviation for the table name in the **Table** box. Enter an easily typed substitute for the fully qualified table name to use in criteria or an SQL WHERE clause.

Note: Changes to the Correlation Name apply on the **SQL** and **Selection Criteria** tabs.

Variable Delimiter

Character required to identify substitution variables in SQL WHERE clauses and selection criteria. To change the delimiter, click the down arrow.

Note: Optim automatically revises the SQL WHERE clauses and selection criteria to reflect the change.

SELECT. . . FROM (table) WHERE

Enter the SQL WHERE clause portion of the SELECT Statement. To:

- Create an SQL WHERE clause, type directly into the SELECT text box or select from Columns, Relational Operators, and Logical Operators lists.
- Remove an SQL WHERE clause, right-click the SELECT text box and select **Remove SQL**. You may also overtype with blanks, or use the Delete or Backspace key.

- Validate the syntax of an SQL WHERE clause, right-click the SELECT text box and select **Validate SQL**, or click outside the text box. If the DBMS returns an error on an SQL Prepare statement, a message is displayed.

Columns

A list of the columns in the table. Select a column name to add it to the SQL WHERE clause at the cursor position. You cannot search an SQL Variant column.

Operator Symbols

A list of valid operator symbols that you can use in the SQL WHERE clause. Select an operator symbol to insert it at the cursor position.

Operators

A list of valid operators that you can use in the SQL WHERE clause. Select an operator to insert it at the cursor position.

BEFORE Operator

A unique operator allows you to select data on the basis of date. The syntax for this operator is:
colname **BEFORE (***n***D** *n***W** *n***M** *n***Y)**

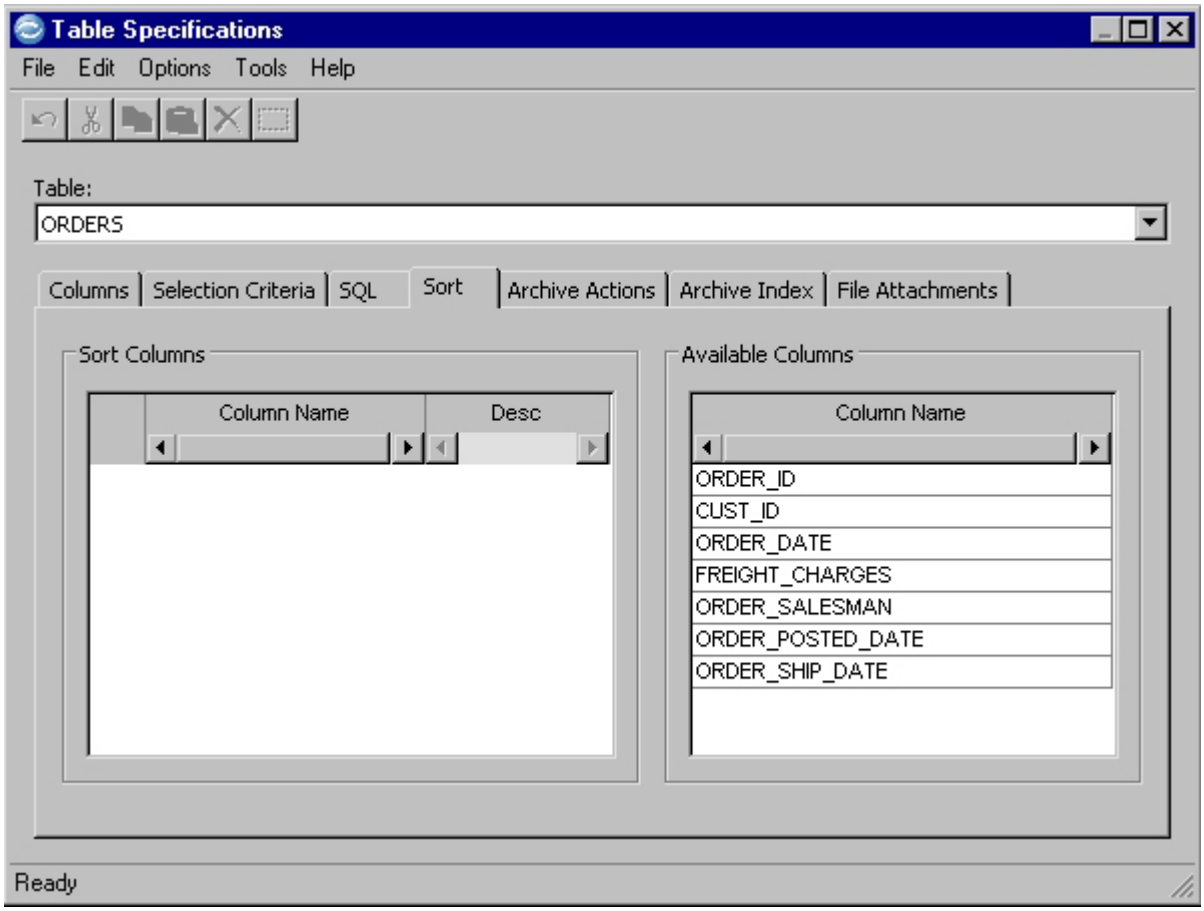
The column referenced by *colname* must be a DATE column. Use the D, W, M, and Y arguments in any combination to indicate the number of days, weeks, months, or years subtracted from the date at runtime. If no arguments are specified, the current date is used. Rows with a date in *colname* that is older than the calculated date are extracted or archived. The *n* multiplier is an integer and can optionally be preceded by + or -.

SQL WHERE Specifications Identified

An SQL icon  in the Table Specifications column on the Access Definition Editor table List indicates that SQL WHERE clause specifications apply for the table.

Sort Tab

Use the **Sort** tab to arrange the display of rows in the Table Editor or Point and Shoot Editor. To arrange rows according to values in one or more columns, drag column names from the Available Columns list to the Sort Columns list.



Sort Columns

Names of columns, in priority order, used for arranging the display of rows in the table. The arrangement has no effect in a process using the Access Definition.

Column Name

Name of column. Select a column name in the **Available Columns** box and drag it to the **Sort Columns** box. (You can also double-click a column name to move it.)

- To rearrange the order of priority in the **Sort Columns** box, drag a column name to a new position.
- To remove a column name from the **Sort Columns** box, drag the column name to the **Available Columns** box, or right-click the name and select **Remove**.

Desc Sequence of values in the column:

- To arrange in descending order, select the check box.
- To arrange in ascending order, clear the check box.

Available Columns

Names of columns that are not involved in the sort sequence.

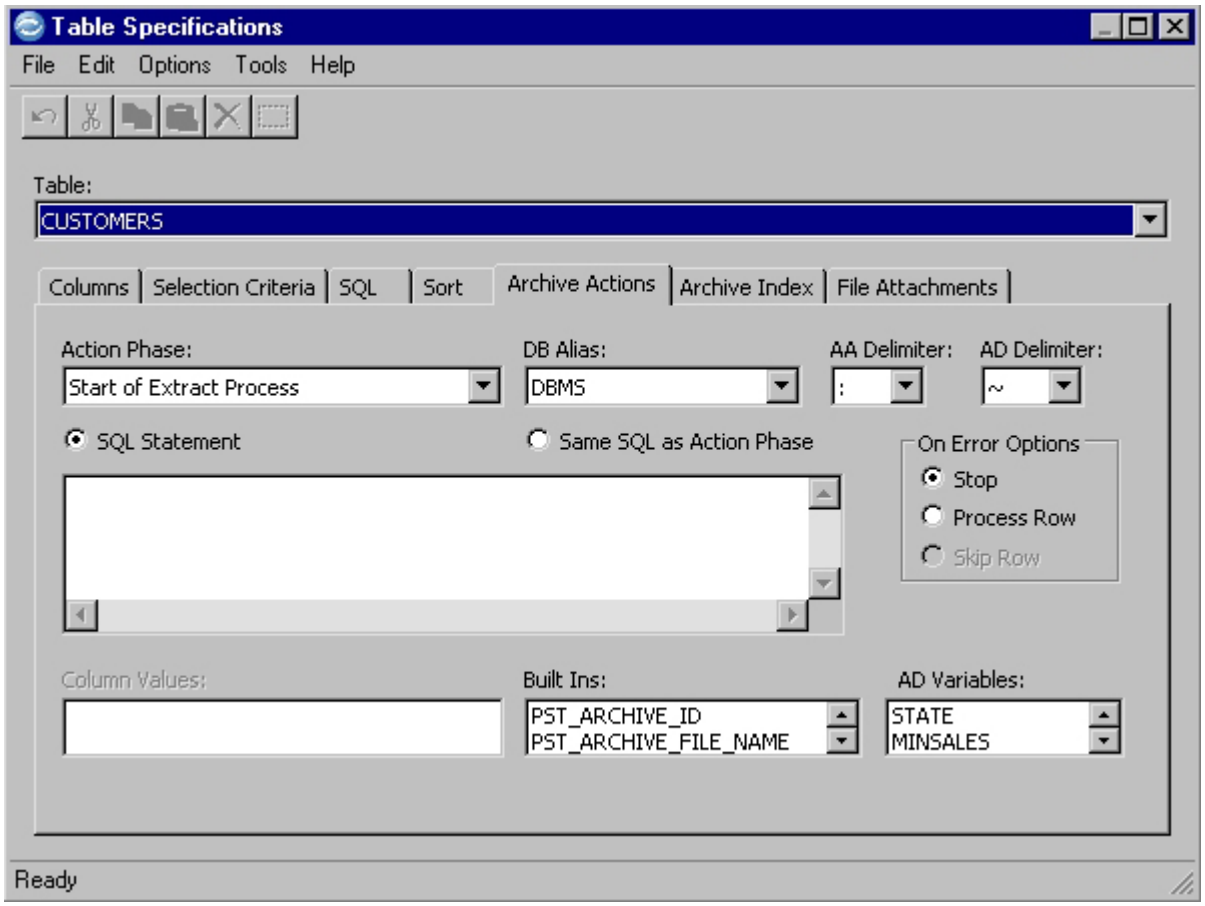
Sort Specifications Identified

A sort criteria icon in the Table Specifications column on the Access Definition Editor table List indicates that sort criteria are specified.

Archive Actions Tab

Use the **Archive Actions** tab to create supplemental SQL statements (Actions) to be executed at selected phases of an Archive, Delete, or Restore Process. For example, you might use this feature to audit deleted rows by executing an appropriate SQL statement before the start of the Delete Process, or after the deletion of each row. You can also create an SQL statement to call a stored procedure.

Note: You can define actions for all phases, including the Restore phases, in the Access Definition. The Access Definition is copied to the Archive File during the Archive Process. Any Restore Process actions in the Access Definition become the default actions when data is restored. These actions can be overridden in a Table Map used in the Restore Process. When you specify Load Processing in a Restore Request, Archive Actions are disregarded for most phases.



Action Phase

Click the arrow to select an Action Phase from the list. You can create an SQL statement for each selected Action Phase or use one SQL statement for multiple Action Phases. The Action Phase box lists phases in chronological order, from **Start of the Extract Process** through **End of Restore Process**.

Archive Actions are in two classes:

Global Actions

Apply to the process, and can be executed at the:

- Start of Extract Process
- End of Extract Process
- Start of Delete Process

- End of Delete Process
- Start of Restore Process
- End of Restore Process

Local Actions

Apply only to the selected table and can be executed:

- Before Extract of First Row from Table
- Before Extract of Row
- After Extract of Last Row from Table
- Before Delete of First Row from Table
- Before Delete of Row
- After Delete of Row
- After Delete of Last Row from Table
- Before Restore of First Row from Table
- Before Restore of Row
- After Restore of Row
- After Restore of Last Row from Table

Note: When you specify Insert Processing in a Restore Request, actions can be executed for all of the phases listed above. When you specify Load Processing in a Restore Request, actions can only be executed for the Start of Restore Process, End of Restore Process, Before Restore of First Row from Table, and After Restore of Last Row from Table phases; actions defined for all other phases are ignored.

DB Alias

Direct references to a DB Alias are invalid in an SQL statement. Thus, Optim generates supplementary SQL to identify the database. Enter a DB Alias needed to reference a table in a database other than the database referenced by the Default Qualifier.

AA Delimiter

Character required to identify a column value or built-in variable in SQL statements for Archive Actions. To change the delimiter, click the arrow and select from the list. The AA delimiter must differ from the AD delimiter.

AD Delimiter

Character required to identify an Access Definition variable (defined on the **Variables** tab in the Access Definition) in an SQL statement for the Archive Action.

Note: Access Definition variables are available for Extract and Restore Action Phases only.

To change the delimiter, click the arrow and select from the list. The AD delimiter must differ from the AA delimiter.

Note: The AD delimiter on the **Archive Actions** tab serves only to identify the variable in the SQL statement; it does not affect the Variable Delimiter used on the **SQL** and **Selection Criteria** tabs.

SQL Statement

The **SQL Statement** option enables the text box for the selected Action Phase. Type the desired SQL statement (Insert, Update, Delete, or Stored Procedure Call) in the box. You can select a column value, built⌂in variable, or Access Definition variable from the list boxes. (The selected variable and delimiter are inserted at the cursor location.)

Syntax for a stored procedure call is: *(qualifier.procedurename (arg1,arg2, ...argn)* . The parentheses around the argument list are required, and empty parentheses ( ) must be used if no arguments are required. Arguments can be literals or parameters.

To validate the syntax of an INSERT, UPDATE, or DELETE statement, right-click the text box and select **Validate SQL** from the shortcut menu. To remove an SQL statement, right-click the text box and select **Remove SQL**. You may also overtype with blanks or use the Delete or Backspace key.

You can use a raise exception statement in a stored procedure called by a 'Before' Archive Action statement to issue a message that includes the text **PST_STATE=SKIP_ROW** (anywhere in the message). If that text is in a message, Archive returns the value 10 for the Built-In Variable PST_AFTER_ROW_STATUS and the row is skipped.

Similarly, a stored procedure called by any Archive Action (Before, After, Start, or End), can issue a message that includes the text **PST_STATE=ABORT** (anywhere in the message). If that text is in a message, the process terminates.

See documentation for your DBMS to determine the proper syntax for a raise exception statement. The following is an example for an Oracle database:

```
CREATE PROCEDURE YourStoredProcedure () AS BEGIN
   /* Issue User-defined error message */
   raise_application_error(-20101, 'PST_STATE=SKIP_ROW'),
END YourStoredProcedure
```

Same SQL as Action Phase

Use the **Same SQL Statement as Action Phase** option to share an SQL statement specified for an Action Phase of the same class. Create the statement for one phase, select an Action Phase of the same class, and select **Same SQL as Action Phase**. Click the arrow to select the Action Phase associated with the SQL statement you want to apply.

Note: An SQL statement may include one or more Column Value or Built-In variables. Be certain that variables are available to all action phases that share an SQL statement.

On Error Options

If an error occurs when an SQL statement is executed, the Error Option you select determines how processing continues:

Stop The process stops when an error occurs as a result of the SQL statement.

Process Row
The SQL statement for the row is ignored and processing continues.

Skip Row
The row is not processed, but processing of other rows continues (available only for Action Phases: Before Extract of Row, Before Delete of Row, and Before Restore of Row). The process report notes the number of rows skipped. This option is not applicable when you specify Load Processing in a Restore Request.

Column Values

Names of columns that can be used in SQL statements associated with certain local Action Phases. When you select a column value (i.e., column name) for an action, the column name is inserted into the SQL statement as a variable, and the column value is used each time the statement is executed. Select a column name to insert it at the cursor position.

Built-Ins

Supported built-in functions that can be used in SQL statements to return information about an Action Phase. Select a built-in variable name to insert it at the cursor position.

AD Variables

Lists any Access Definition variables defined on the **Variables** tab of the Access Definition Editorfor the specified table. Select an Access Definition variable to insert the name at the cursor position in the SQL Statement box.

Note: AD Variables are available only when Extract and Restore Action Phases are specified.

Archive Actions Specifications Identified

An Archive Actions icon  in Table Specifications on the Access Definition Editor Table List indicates that one or more local Archive Actions are specified.

Action Phases and Associated Built-Ins

The following table lists the available Action Phases, the built-in functions associated with each, and the corresponding data type:

Note: DATE_TIME data types depend on the DBMS:

DBMS	Data Type
UDB	TIMESTAMP
Oracle	DATE
Sybase ASE	DATETIME
SQL Server	DATETIME
Informix	DATETIME

Action Phase	Available Built-in Variables	Data Type
Start of Extract Process	PST_ARCHIVE_ID	INTEGER
	PST_ARCHIVE_FILE_NAME	VARCHAR (250)
	PST_ACTION (returns “0” for this phase)	INTEGER
	PST_ACTION_TEXT (returns “BefExtProcess”)	VARCHAR (20)
	PST_INDEX_FILE_NAME	VARCHAR (250)
	PST_GROUP_NAME	VARCHAR (8)
	PST_USER_ID	VARCHAR (30)
	PST_SERVER_NAME	VARCHAR (15)
	PST_MACHINE_NAME	VARCHAR (15)
	PST_ARCHIVE_DESC	VARCHAR (128)
	PST_START_DATETIME	vary

Action Phase	Available Built-in Variables	Data Type
	PST_CURRENT_DATETIME	vary
	PST_CURRENT_DATETIME_CHAR	VARCHAR(24)
	PST_TOTAL_ROWS_EXTRACTED	INTEGER
	PST_ARCHIVE_GUID	CHAR (38)
	PST_PROCESS_STATUS	INTEGER

Action Phase	Available Built-in Variables	Data Type
Before Extract of First Row from Table	PST_ARCHIVE_ID	INTEGER
	PST_ARCHIVE_FILE_NAME	VARCHAR (250)
	PST_ACTION (returns "1" for this phase)	INTEGER
	PST_ACTION_TEXT (returns "BefExtFirstRow")	VARCHAR (20)
	PST_INDEX_FILE_NAME	VARCHAR (250)
	PST_GROUP_NAME	VARCHAR (8)
	PST_USER_ID	VARCHAR (30)
	PST_SERVER_NAME	VARCHAR (15)
	PST_MACHINE_NAME	VARCHAR (15)
	PST_ARCHIVE_DESC	VARCHAR (128)
	PST_START_DATETIME	*vary*
	PST_CURRENT_DATETIME	*vary*
	PST_CURRENT_DATETIME_CHAR	VARCHAR(24)
	PST_TOTAL_ROWS_EXTRACTED	INTEGER
	PST_ARCHIVE_GUID	CHAR(38)
	PST_PROCESS_STATUS	INTEGER
	PST_TBL_DBALIAS	VARCHAR (12)
	PST_TBL_SCHEMA_NAME	VARCHAR (64)
	PST_TBL_NAME	VARCHAR (64)
	PST_TOTAL_ROWS_EXTRACTED	INTEGER

Action Phase	Available Built-in Variables	Data Type
Before Extract of Row	PST_ARCHIVE_ID	INTEGER
	PST_ARCHIVE_FILE_NAME	VARCHAR (250)
	PST_ACTION (returns "2" for this phase)	INTEGER
	PST_ACTION_TEXT (returns "BefExtRow")	VARCHAR (20)
	PST_INDEX_FILE_NAME	VARCHAR (250)
	PST_GROUP_NAME	VARCHAR (8)
	PST_USER_ID	VARCHAR (30)
	PST_SERVER_NAME	VARCHAR (15)
	PST_MACHINE_NAME	VARCHAR (15)
	PST_ARCHIVE_DESC	VARCHAR (128)
	PST_START_DATETIME	*vary*

Action Phase	Available Built-in Variables	Data Type
	PST_CURRENT_DATETIME	*vary*
	PST_CURRENT_DATETIME_CHAR	VARCHAR(24)
	PST_TOTAL_ROWS_EXTRACTED	INTEGER
	PST_ARCHIVE_GUID	CHAR (38)
	PST_PROCESS_STATUS	INTEGER
	PST_TBL_DBALIAS	VARCHAR (12)
	PST_TBL_SCHEMA_NAME	VARCHAR (64)
	PST_TBL_NAME	VARCHAR (64)
	PST_TBL_ROWS_EXTRACTED	INTEGER

Action Phase	Available Built-in Variables	Data Type
After Extract of Last Row from Table	PST_ARCHIVE_ID	INTEGER
	PST_ARCHIVE_FILE_NAME	VARCHAR (250)
	PST_ACTION (returns "3" for this phase)	INTEGER
	PST_ACTION_TEXT (returns "AftExtLast Row")	VARCHAR (20)
	PST_INDEX_FILE_NAME	VARCHAR (250)
	PST_GROUP_NAME	VARCHAR (8)
	PST_USER_ID	VARCHAR (30)
	PST_SERVER_NAME	VARCHAR (15)
	PST_MACHINE_NAME	VARCHAR (15)
	PST_ARCHIVE_DESC	VARCHAR (128)
	PST_START_DATETIME	*vary*
	PST_CURRENT_DATETIME	*vary*
	PST_CURRENT_DATETIME_CHAR	VARCHAR(24)
	PST_TOTAL_ROWS_EXTRACTED	INTEGER
	PST_ARCHIVE_GUID	CHAR (38)
	PST_PROCESS_STATUS	INTEGER
	PST_TBL_DBALIAS	VARCHAR (12)
	PST_TBL_SCHEMA_NAME	VARCHAR (64)
	PST_TBL_NAME	VARCHAR (64)
	PST_TBL_ROWS_EXTRACTED	INTEGER

Action Phase	Available Built-in Variables	Data Type
End of Extract Process	PST_ARCHIVE_ID	INTEGER
	PST_ARCHIVE_FILE_NAME	VARCHAR (250)
	PST_ACTION (returns "4" for this phase)	INTEGER
	PST_ACTION_TEXT (returns "AftExtProcess")	VARCHAR (20)
	PST_INDEX_FILE_NAME	VARCHAR (250)
	PST_GROUP_NAME	VARCHAR (8)
	PST_USER_ID	VARCHAR (30)

Action Phase	Available Built-in Variables	Data Type
	PST_SERVER_NAME	VARCHAR (15)
	PST_MACHINE_NAME	VARCHAR (15)
	PST_ARCHIVE_DESC	VARCHAR (128)
	PST_START_DATETIME	*vary*
	PST_CURRENT_DATETIME	*vary*
	PST_CURRENT_DATETIME_CHAR	VARCHAR(24)
	PST_TOTAL_ROWS_EXTRACTED	INTEGER
	PST_ARCHIVE_GUID	CHAR (38)
	PST_PROCESS_STATUS	INTEGER

Action Phase	Available Built-in Variables	Data Type
Start of Delete Process	PST_ARCHIVE_ID	INTEGER
	PST_ARCHIVE_FILE_NAME	VARCHAR (250)
	PST_GROUP_NAME	VARCHAR (8)
	PST_USER_ID	VARCHAR (30)
	PST_SERVER_NAME	VARCHAR (15)
	PST_MACHINE_NAME	VARCHAR (15)
	PST_ARCHIVE_DESC	VARCHAR (128)
	PST_START_DATETIME	*vary*
	PST_CURRENT_DATETIME	*vary*
	PST_CURRENT_DATETIME_CHAR	VARCHAR(24)
	PST_ACTION (returns “5” for this phase)	INTEGER
	PST_ACTION_TEXT (returns “BefDelProcess”)	VARCHAR (20)
	PST_TOTAL_PROCESS_STATUS	INTEGER
	PST_TOTAL_ROWS_PROCESSED	INTEGER
	PST_TOTAL_ROWS_IN_ERROR	INTEGER
	PST_TOTAL_ROWS_REMAINING_TO_BE_PROCESSED	INTEGER
	PST_ARCHIVE_GUID	CHAR (38)

Action Phase	Available Built-in Variables	Data Type
Before Delete of First Row from Table	PST_ARCHIVE_ID	INTEGER
	PST_ARCHIVE_FILE_NAME	VARCHAR (250)
	PST_GROUP_NAME	VARCHAR (8)
	PST_USER_ID	VARCHAR (30)
	PST_SERVER_NAME	VARCHAR (15)
	PST_MACHINE_NAME	VARCHAR (15)
	PST_ARCHIVE_DESC	VARCHAR (128)
	PST_START_DATETIME	*vary*
	PST_CURRENT_DATETIME	*vary*

Action Phase	Available Built-in Variables	Data Type
	PST_CURRENT_DATETIME_CHAR	VARCHAR(24)
	PST_ACTION (returns “6” for this phase)	INTEGER
	PST_ACTION_TEXT (returns “BefDelFirstRow”)	VARCHAR (20)
	PST_TOTAL_PROCESS_STATUS	INTEGER
	PST_TOTAL_ROWS_PROCESSED	INTEGER
	PST_TOTAL_ROWS_IN_ERROR	INTEGER
	PST_TOTAL_ROWS_REMAINING_TO_BE_PROCESSED	INTEGER
	PST_ARCHIVE_GUID	CHAR (38)
	PST_SRCTBL_DBALIAS	VARCHAR (12)
	PST_SRCTBL_SCHEMA_NAME	VARCHAR (64)
	PST_SRCTBL_NAME	VARCHAR (64)
	PST_DSTTBL_DBALIAS	VARCHAR (12)
	PST_DSTTBL_SCHEMA_NAME	VARCHAR (64)
	PST_DSTTBL_NAME	VARCHAR (64)
	PST_TBL_ROWS_PROCESSED	INTEGER
	PST_TBL_ROWS_IN_ERROR	INTEGER
	PST_TBL_ROWS_REMAINING_TO _BE_PROCESSED	INTEGER
	PST_AFTER_ROW_STATUS	INTEGER
Action Phase	**Available Built-in Variables**	**Data Type**
Before Delete of Row	PST_ARCHIVE_ID	INTEGER
	PST_ARCHIVE_FILE_NAME	VARCHAR (250)
	PST_GROUP_NAME	VARCHAR (8)
	PST_USER_ID	VARCHAR (30)
	PST_SERVER_NAME	VARCHAR (15)
	PST_MACHINE_NAME	VARCHAR (15)
	PST_ARCHIVE_DESC	VARCHAR (128)
	PST_START_DATETIME	*vary*
	PST_CURRENT_DATETIME	*vary*
	PST_CURRENT_DATETIME_CHAR	VARCHAR(24)
	PST_ACTION (returns “7” for this phase)	INTEGER
	PST_ACTION_TEXT (returns “BefDelRow”)	VARCHAR (20)
	PST_TOTAL_PROCESS_STATUS	INTEGER
	PST_TOTAL_ROWS_PROCESSED	INTEGER
	PST_TOTAL_ROWS_IN_ERROR	INTEGER
	PST_TOTAL_ROWS_REMAINING_TO_BE_PROCESSED	INTEGER
	PST_ARCHIVE_GUID	CHAR (38)
	PST_SRCTBL_DBALIAS	VARCHAR (12)
	PST_SRCTBL_SCHEMA_NAME	VARCHAR (64)
	PST_SRCTBL_NAME	VARCHAR (64)

Action Phase	Available Built-in Variables	Data Type
	PST_DSTTBL_DBALIAS	VARCHAR (12)
	PST_DSTTBL_SCHEMA_NAME	VARCHAR (64)
	PST_DSTTBL_NAME	VARCHAR (64)
	PST_TBL_ROWS_PROCESSED	INTEGER
	PST_TBL_ROWS_IN_ERROR	INTEGER
	PST_TBL_ROWS_REMAINING_TO _BE_PROCESSED	INTEGER
	PST_AFTER_ROW_STATUS	INTEGER
Action Phase	**Available Built-in Variables**	**Data Type**
After Delete of Row	PST_ARCHIVE_ID	INTEGER
	PST_ARCHIVE_FILE_NAME	VARCHAR (250)
	PST_GROUP_NAME	VARCHAR (8)
	PST_USER_ID	VARCHAR (30)
	PST_SERVER_NAME	VARCHAR (15)
	PST_MACHINE_NAME	VARCHAR (15)
	PST_ARCHIVE_DESC	VARCHAR (128)
	PST_START_DATETIME	*vary*
	PST_CURRENT_DATETIME	*vary*
	PST_CURRENT_DATETIME_CHAR	VARCHAR(24)
	PST_ACTION (returns “8” for this phase)	INTEGER
	PST_ACTION_TEXT (returns “AftDelRow”)	VARCHAR (20)
	PST_TOTAL_PROCESS_STATUS	INTEGER
	PST_TOTAL_ROWS_PROCESSED	INTEGER
	PST_TOTAL_ROWS_IN_ERROR	INTEGER
	PST_TOTAL_ROWS_REMAINING_TO_BE_PROCESSED	INTEGER
	PST_ARCHIVE_GUID	CHAR (38)
	PST_SRCTBL_DBALIAS	VARCHAR (12)
	PST_SRCTBL_SCHEMA_NAME	VARCHAR (64)
	PST_SRCTBL_NAME	VARCHAR (64)
	PST_DSTTBL_DBALIAS	VARCHAR (12)
	PST_DSTTBL_SCHEMA_NAME	VARCHAR (64)
	PST_DSTTBL_NAME	VARCHAR (64)
	PST_TBL_ROWS_PROCESSED	INTEGER
	PST_TBL_ROWS_IN_ERROR	INTEGER
	PST_TBL_ROWS_REMAINING_TO _BE_PROCESSED	INTEGER
	PST_AFTER_ROW_STATUS	INTEGER
Action Phase	**Available Built-in Variables**	**Data Type**
After Delete of Last Row from Table	PST_ARCHIVE_ID	INTEGER

Action Phase	Available Built-in Variables	Data Type
	PST_ARCHIVE_FILE_NAME	VARCHAR (250)
	PST_GROUP_NAME	VARCHAR (8)
	PST_USER_ID	VARCHAR (30)
	PST_SERVER_NAME	VARCHAR (15)
	PST_MACHINE_NAME	VARCHAR (15)
	PST_ARCHIVE_DESC	VARCHAR (128)
	PST_START_DATETIME	*vary*
	PST_CURRENT_DATETIME	*vary*
	PST_CURRENT_DATETIME_CHAR	VARCHAR(24)
	PST_ACTION (returns "9" for this phase)	INTEGER
	PST_ACTION_TEXT (returns "AftDelLastRow")	VARCHAR (20)
	PST_TOTAL_PROCESS_STATUS	INTEGER
	PST_TOTAL_ROWS_PROCESSED	INTEGER
	PST_TOTAL_ROWS_IN_ERROR	INTEGER
	PST_TOTAL_ROWS_REMAINING_TO_BE_PROCESSED	INTEGER
	PST_ARCHIVE_GUID	CHAR (38)
	PST_SRCTBL_DBALIAS	VARCHAR (12)
	PST_SRCTBL_SCHEMA_NAME	VARCHAR (64)
	PST_SRCTBL_NAME	VARCHAR (64)
	PST_DSTTBL_DBALIAS	VARCHAR (12)
	PST_DSTTBL_SCHEMA_NAME	VARCHAR (64)
	PST_DSTTBL_NAME	VARCHAR (64)
	PST_TBL_ROWS_PROCESSED	INTEGER
	PST_TBL_ROWS_IN_ERROR	INTEGER
	PST_TBL_ROWS_REMAINING_TO _BE_PROCESSED	INTEGER
	PST_AFTER_ROW_STATUS	INTEGER
Action Phase	**Available Built-in Variables**	**Data Type**
End of Delete Process	PST_ARCHIVE_ID	INTEGER
	PST_ARCHIVE_FILE_NAME	VARCHAR (250)
	PST_GROUP_NAME	VARCHAR (8)
	PST_USER_ID	VARCHAR (30)
	PST_SERVER_NAME	VARCHAR (15)
	PST_MACHINE_NAME	VARCHAR (15)
	PST_ARCHIVE_DESC	VARCHAR (128)
	PST_START_DATETIME	*vary*
	PST_CURRENT_DATETIME	*vary*
	PST_CURRENT_DATETIME_CHAR	VARCHAR(24)
	PST_ACTION (returns "10" for this phase)	INTEGER
	PST_ACTION_TEXT (returns "AftDelProcess")	VARCHAR (20)

Action Phase	Available Built-in Variables	Data Type
	PST_TOTAL_PROCESS_STATUS	INTEGER
	PST_TOTAL_ROWS_PROCESSED	INTEGER
	PST_TOTAL_ROWS_IN_ERROR	INTEGER
	PST_TOTAL_ROWS_REMAINING_TO_BE_PROCESSED	INTEGER
	PST_ARCHIVE_GUID	CHAR (38)
Action Phase	**Available Built-in Variables**	**Data Type**
Start of Restore Process	PST_ARCHIVE_ID	INTEGER
	PST_ARCHIVE_FILE_NAME	VARCHAR (250)
	PST_GROUP_NAME	VARCHAR (8)
	PST_USER_ID	VARCHAR (30)
	PST_SERVER_NAME	VARCHAR (15)
	PST_MACHINE_NAME	VARCHAR (15)
	PST_ARCHIVE_DESC	VARCHAR (128)
	PST_START_DATETIME	*vary*
	PST_CURRENT_DATETIME	*vary*
	PST_CURRENT_DATETIME_CHAR	VARCHAR(24)
	PST_ACTION (returns “11” for this phase)	INTEGER
	PST_ACTION_TEXT (returns “BefRestProcess”)	VARCHAR (20)
	PST_TOTAL_PROCESS_STATUS	INTEGER
	PST_TOTAL_ROWS_PROCESSED	INTEGER
	PST_TOTAL_ROWS_IN_ERROR	INTEGER
	PST_TOTAL_ROWS_REMAINING_TO_BE_PROCESSED	INTEGER
	PST_ARCHIVE_GUID	CHAR (38)
Action Phase	**Available Built-in Variables**	**Data Type**
Before Restore of First Row to Table	PST_ARCHIVE_ID	INTEGER
	PST_ARCHIVE_FILE_NAME	VARCHAR (250)
	PST_GROUP_NAME	VARCHAR (8)
	PST_USER_ID	VARCHAR (30)
	PST_SERVER_NAME	VARCHAR (15)
	PST_MACHINE_NAME	VARCHAR (15)
	PST_ARCHIVE_DESC	VARCHAR (128)
	PST_START_DATETIME	*vary*
	PST_CURRENT_DATETIME	*vary*
	PST_CURRENT_DATETIME_CHAR	VARCHAR(24)
	PST_ACTION (returns “12” for this phase)	INTEGER
	PST_ACTION_TEXT (returns “BefRestFirstRow”)	VARCHAR (20)
	PST_TOTAL_PROCESS_STATUS	INTEGER

Action Phase	Available Built-in Variables	Data Type
	PST_TOTAL_ROWS_PROCESSED	INTEGER
	PST_TOTAL_ROWS_IN_ERROR	INTEGER
	PST_TOTAL_ROWS_REMAINING_TO_BE_PROCESSED	INTEGER
	PST_ARCHIVE_GUID	CHAR (38)
	PST_SRCTBL_DBALIAS	VARCHAR (12)
	PST_SRCTBL_SCHEMA_NAME	VARCHAR (64)
	PST_SRCTBL_NAME	VARCHAR (64)
	PST_DSTTBL_DBALIAS	VARCHAR (12)
	PST_DSTTBL_SCHEMA_NAME	VARCHAR (64)
	PST_DSTTBL_NAME	VARCHAR (64)
	PST_TBL_ROWS_PROCESSED	INTEGER
	PST_TBL_ROWS_IN_ERROR	INTEGER
	PST_TBL_ROWS_REMAINING_TO _BE_PROCESSED	INTEGER
	PST_AFTER_ROW_STATUS	INTEGER

Action Phase	Available Built-in Variables	Data Type
Before Restore of Row	PST_ARCHIVE_ID	INTEGER
	PST_ARCHIVE_FILE_NAME	VARCHAR (250)
	PST_GROUP_NAME	VARCHAR (8)
	PST_USER_ID	VARCHAR (30)
	PST_SERVER_NAME	VARCHAR (15)
	PST_MACHINE_NAME	VARCHAR (15)
	PST_ARCHIVE_DESC	VARCHAR (128)
	PST_START_DATETIME	*vary*
	PST_CURRENT_DATETIME	*vary*
	PST_CURRENT_DATETIME_CHAR	VARCHAR(24)
	PST_ACTION (returns “13” for this phase)	INTEGER
	PST_ACTION_TEXT (returns “BefRestRow”)	VARCHAR (20)
	PST_TOTAL_PROCESS_STATUS	INTEGER
	PST_TOTAL_ROWS_PROCESSED	INTEGER
	PST_TOTAL_ROWS_IN_ERROR	INTEGER
	PST_TOTAL_ROWS_REMAINING_TO_BE_PROCESSED	INTEGER
	PST_ARCHIVE_GUID	CHAR (38)
	PST_SRCTBL_DBALIAS	VARCHAR (12)
	PST_SRCTBL_SCHEMA_NAME	VARCHAR (64)
	PST_SRCTBL_NAME	VARCHAR (64)
	PST_DSTTBL_DBALIAS	VARCHAR (12)
	PST_DSTTBL_SCHEMA_NAME	VARCHAR (64)
	PST_DSTTBL_NAME	VARCHAR (64)
	PST_TBL_ROWS_PROCESSED	INTEGER

Action Phase	Available Built-in Variables	Data Type
	PST_TBL_ROWS_IN_ERROR	INTEGER
	PST_TBL_ROWS_REMAINING_TO _BE_PROCESSED	INTEGER
	PST_AFTER_ROW_STATUS	INTEGER

Action Phase	Available Built-in Variables	Data Type
After Restore of Row	PST_ARCHIVE_ID	INTEGER
	PST_ARCHIVE_FILE_NAME	VARCHAR (250)
	PST_GROUP_NAME	VARCHAR (8)
	PST_USER_ID	VARCHAR (30)
	PST_SERVER_NAME	VARCHAR (15)
	PST_MACHINE_NAME	VARCHAR (15)
	PST_ARCHIVE_DESC	VARCHAR (128)
	PST_START_DATETIME	*vary*
	PST_CURRENT_DATETIME	*vary*
	PST_CURRENT_DATETIME_CHAR	VARCHAR(24)
	PST_ACTION (returns “14” for this phase)	INTEGER
	PST_ACTION_TEXT (returns “AftRestRow”)	VARCHAR (20)
	PST_TOTAL_PROCESS_STATUS	INTEGER
	PST_TOTAL_ROWS_PROCESSED	INTEGER
	PST_TOTAL_ROWS_IN_ERROR	INTEGER
	PST_TOTAL_ROWS_REMAINING_TO_BE_PROCESSED	INTEGER
	PST_ARCHIVE_GUID	CHAR (38)
	PST_SRCTBL_DBALIAS	VARCHAR (12)
	PST_SRCTBL_SCHEMA_NAME	VARCHAR (64)
	PST_SRCTBL_NAME	VARCHAR (64)
	PST_DSTTBL_DBALIAS	VARCHAR (12)
	PST_DSTTBL_SCHEMA_NAME	VARCHAR (64)
	PST_DSTTBL_NAME	VARCHAR (64)
	PST_TBL_ROWS_PROCESSED	INTEGER
	PST_TBL_ROWS_IN_ERROR	INTEGER
	PST_TBL_ROWS_REMAINING_TO_BE_PROCESSED	INTEGER
	PST_AFTER_ROW_STATUS	INTEGER

Action Phase	Available Built-in Variables	Data Type
After Restore of Last Row to Table	PST_ARCHIVE_ID	INTEGER
	PST_ARCHIVE_FILE_NAME	VARCHAR (250)
	PST_GROUP_NAME	VARCHAR (8)
	PST_USER_ID	VARCHAR (30)
	PST_SERVER_NAME	VARCHAR (15)

Action Phase	Available Built-in Variables	Data Type
	PST_MACHINE_NAME	VARCHAR (15)
	PST_ARCHIVE_DESC	VARCHAR (128)
	PST_START_DATETIME	*vary*
	PST_CURRENT_DATETIME	*vary*
	PST_CURRENT_DATETIME_CHAR	VARCHAR(24)
	PST_ACTION (returns "15" for this phase)	INTEGER
	PST_ACTION_TEXT (returns "AftRestLastRow")	VARCHAR (20)
	PST_TOTAL_PROCESS_STATUS	INTEGER
	PST_TOTAL_ROWS_PROCESSED	INTEGER
	PST_TOTAL_ROWS_IN_ERROR	INTEGER
	PST_TOTAL_ROWS_REMAINING_TO_BE_PROCESSED	INTEGER
	PST_ARCHIVE_GUID	CHAR (38)
	PST_SRCTBL_DBALIAS	VARCHAR (12)
	PST_SRCTBL_SCHEMA_NAME	VARCHAR (64)
	PST_SRCTBL_NAME	VARCHAR (64)
	PST_DSTTBL_DBALIAS	VARCHAR (12)
	PST_DSTTBL_SCHEMA_NAME	VARCHAR (64)
	PST_DSTTBL_NAME	VARCHAR (64)
	PST_TBL_ROWS_PROCESSED	INTEGER
	PST_TBL_ROWS_IN_ERROR	INTEGER
	PST_TBL_ROWS_REMAINING_TO _BE_PROCESSED	INTEGER
	PST_AFTER_ROW_STATUS	INTEGER

Action Phase	Available Built-in Variables	Data Type
End of Restore Process	PST_ARCHIVE_ID	INTEGER
	PST_ARCHIVE_FILE_NAME	VARCHAR (250)
	PST_GROUP_NAME	VARCHAR (8)
	PST_USER_ID	VARCHAR (30)
	PST_SERVER_NAME	VARCHAR (15)
	PST_MACHINE_NAME	VARCHAR (15)
	PST_ARCHIVE_DESC	VARCHAR (128)
	PST_START_DATETIME	*vary*
	PST_CURRENT_DATETIME	*vary*
	PST_CURRENT_DATETIME_CHAR	VARCHAR(24)
	PST_ACTION (returns "16" for this phase)	INTEGER
	PST_ACTION_TEXT (returns "AftRestProcess")	VARCHAR (20)
	PST_TOTAL_PROCESS_STATUS	INTEGER
	PST_TOTAL_ROWS_PROCESSED	INTEGER
	PST_TOTAL_ROWS_IN_ERROR	INTEGER
	PST_TOTAL_ROWS_REMAINING_TO_BE_PROCESSED	INTEGER
	PST_ARCHIVE_GUID	CHAR (38)

Extract Process Status

Extract Action Phases enable a built-in variable labeled PST_PROCESS_STATUS. Possible numerical values returned are:

Value	Explanation
0	Success.
1	The process was cancelled.
2	A DBMS error occurred that aborted Archive.
3	An Archive Action aborted Archive.
4	An error occurred that aborted Archive.

After Row Status

Several Action Phases enable a built-in variable labeled PST_AFTER_ROW_STATUS. Possible numerical values returned are:

Value	Explanation
0	Status unavailable.
1	Inserted successfully.
2	Updated successfully.
3	Deleted successfully.
4	Row (primary key) not found for deletion.
5	Row (primary key) found, but non-primary key columns do not match (therefore, row not deleted).
6	Row not found for updating.
7	Duplicate index.
8	Constraint restriction.
9	View check error (when you define a view with a WITH CHECK clause, any inserted or updated row must conform to the specified condition).
10	The before row action indicated the row should be skipped.
99	All other status conditions.

Total Process Status

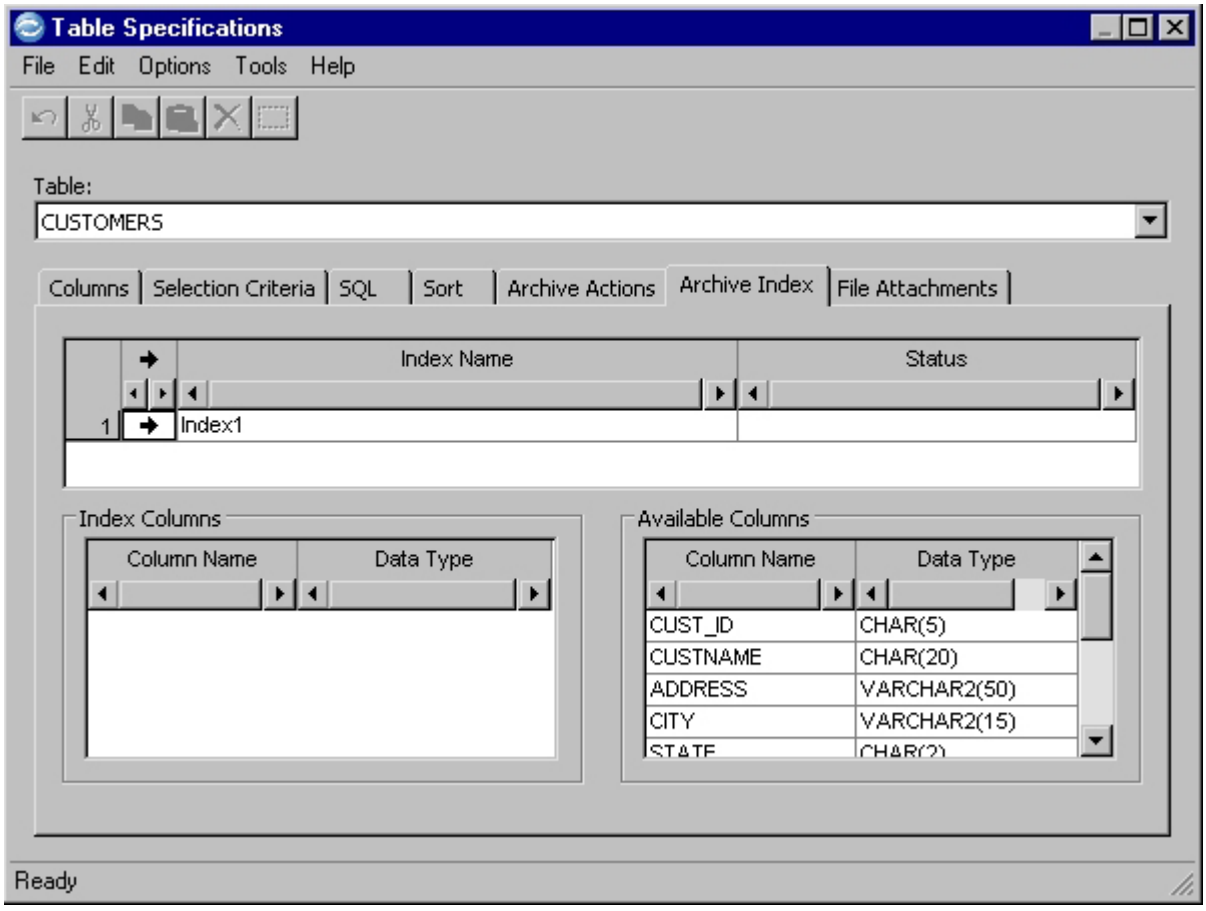
Several Action Phases enable a built-in variable labeled PST_TOTAL_PROCESS_STATUS. Possible numerical values returned are:

Value	Explanation
0	All rows successfully processed.
1	Some rows with errors.
2	Fatal DBMS error.
3	Action aborted.
4	Fatal error.

Archive Index Tab

Archive Index information expedites the identification of archived data when browsing an Archive File, or restoring data in an Archive File.

Archive can search an index more quickly than it can search an entire Archive File. Use the **Archive Index** tab to define one or more indexes of values in one or more columns to use when searching for an Archive File that contains specific data.



You can define indexes in three ways:

- Select **Add Relationship Indexes** from the Table Specifications dialog **Tools** menu to automatically generate indexes for all tables in the Access Definition, based on the relationships between the tables.
- For the current table, right-click and select **Add Database Indexes** to add all columns with database indexes to the current list, or select **Replace with Database Indexes** to replace the current list with a list of columns that have database indexes.
- For the current table, list columns manually by dragging one or more column names from Available Columns to Index Columns.

Index Name

You can create up to 16 indexes for each table in an Archive File. Archive automatically generates a name for each index. You can use this default name, or edit the name according to your conventions. The generated names are in the form "Index*n*" where *n* is a sequential number that provides a unique name for each index. After you create an index, you can right-click the index name and select ADD from the shortcut menu to create additional indexes. If you create more than one index, click the ➡ column to

display the list of Index Columns for the corresponding Archive Index name.

Status

Indicates the status of the Archive Index:

blank Index was defined prior to the current session.

New Database or relationship index was defined during the current session and is new to the list.

Index Columns

Double-click a column name to move it from the **Available Columns** list to the **Index Columns** list. (You can also drag column names between lists.)

Available Columns

The names of columns in the selected table.

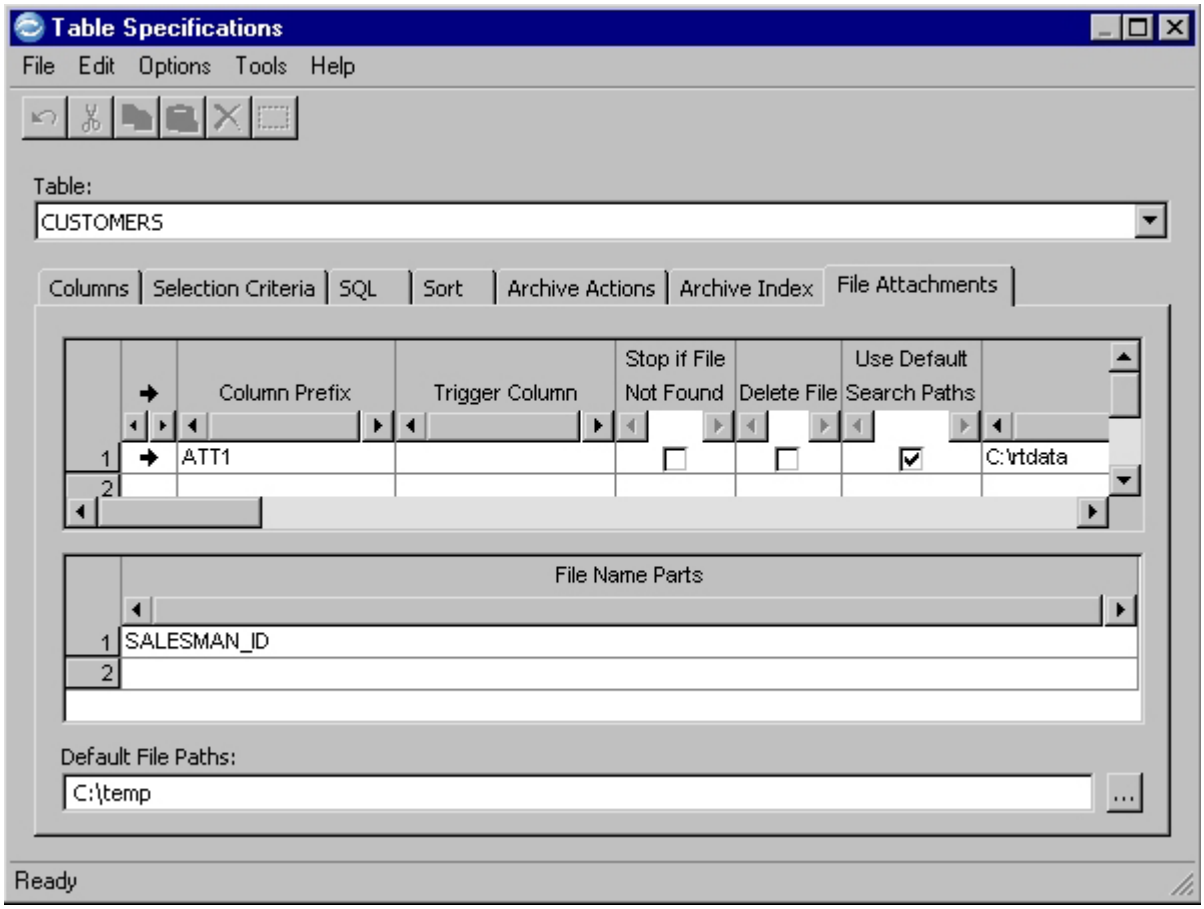
Note: Large Object (LOB) columns are not available for use as Archive Index columns.

Archive Index Specifications Identified

An Archive Index icon  in the Table Specifications column on the Access Definition Editor table List indicates that Archive Index specifications are defined for the table.

File Attachments Tab

Use the **File Attachments** tab to extract files referenced within or associated with a row of extracted data and include it in an Archive or Extract File. You can attach multiple files to each row in a table.



The extracted file is "attached" to an extracted database row by three pseudocolumns within the Archive or Extract File. The pseudocolumns are:

- *prefix*_FILE_NAME (a VARCHAR2 pseudocolumn that includes the file name from **File Name Parts** and the file search path indicated in the **File Attachment List**).
- *prefix*_BLOB (a BLOB pseudocolumn that includes the file data)
- *prefix*_ATTRIB (a BLOB pseudocolumn that includes file attribute information)

File Attachment List

Use the File Attachment List to provide details for each attached file. Each row in the list represents a file that is attached to the extracted data. The ➡ column indicates the selected row.

Column Prefix
The prefix used to name the pseudocolumns in the Archive or Extract File. Each prefix must be unique to the table.

Trigger Column
The name of a column that controls processing of the file attachment. Click the grid cell to obtain a list of eligible columns in the table. (Only a CHAR, VARCHAR, NCHAR, and NVARCHAR can serve as a trigger column.) Leave blank to indicate that no trigger applies. Optional.

Stop if File Not Found
Processing if a file is not found. Select the check box to stop processing or clear to continue.

Delete File
Deletion of attached file during Delete Processing. Select the check box to delete the file after the associated row is successfully deleted or clear to continue.

Use Default Search Paths
Select the check box to search the default paths for the attachment or clear to search the file Search paths. If selected, you must enter one or more paths in **Default File Paths**; if cleared, you must enter one or more paths in **File Search Paths**.

File Search Paths
One or more paths to search for the file, separated by a semicolon. Paths are searched in the order listed. Once the file is found, any remaining paths are not searched. Use the browse button to select paths.

File Name Parts

The file name for the attachment selected in File Attachment List. The name is generated by concatenating values from listed columns with any literals you enter.

You must enter each literal or column name in order on a separate row, enclosing literals in quotes and selecting column names from a drop down list of CHAR, VARCHAR, NCHAR, and NVARCHAR columns in the table.

The name resulting from concatenating the values and literals is combined with the search path(s) in order to locate the appropriate file. For example, if you provide the column name USERID and the literal ".txt" as file name parts, Optim searches for and attaches the first file named 123.txt that is located in a designated search path when processing a database row with the value '123' in the USERID column.

Default File Paths

One or more paths, separated by a semicolon. These paths are the default search paths for any attachments to any table referenced in the Access Definition and are searched if **Use Default Search Paths** is selected in the File Attachment List for any file. Paths are searched in the order listed. Once the file is found, any remaining paths are not searched. Use the browse button to select paths.

Short Cut Menu Commands

Use the following commands from the File Attachments List and the File Name Parts list.

Remove
Remove the selected row.

Remove All
Remove all rows from the list.

File Attachment Specifications Identified

A File Attachment icon  in the Table Specifications column on the **Access Definition Editor** table List indicates that File Attachment specifications are defined for the table.

Access Definition Tools Menu

Use commands on the **Tools** menu in the Access Definition Editor to analyze the combination of tables, relationship selection, table specifications and selection criteria or to select Start Table rows for processing.

Indented Table Display

Select **Indent** from the **Tools** menu to view the Indented Table Display dialog. The Indented Table Display dialog has three tabs:

- On the **Indent** tab, a list of tables referenced in the Access Definition is formatted to show relationships.
- On the **Reference** tab, any reference tables are listed.
- On the **Unconnected** tab, any unconnected tables (tables that are not related to other tables on the list) are listed.

Default Qualifier

Default qualifier specified for the Access Definition.

Indent Tab

View a list of table names in a format that shows the relationships selected in the Access Definition. The rows on the **Indent** tab are numbered for easy reference.

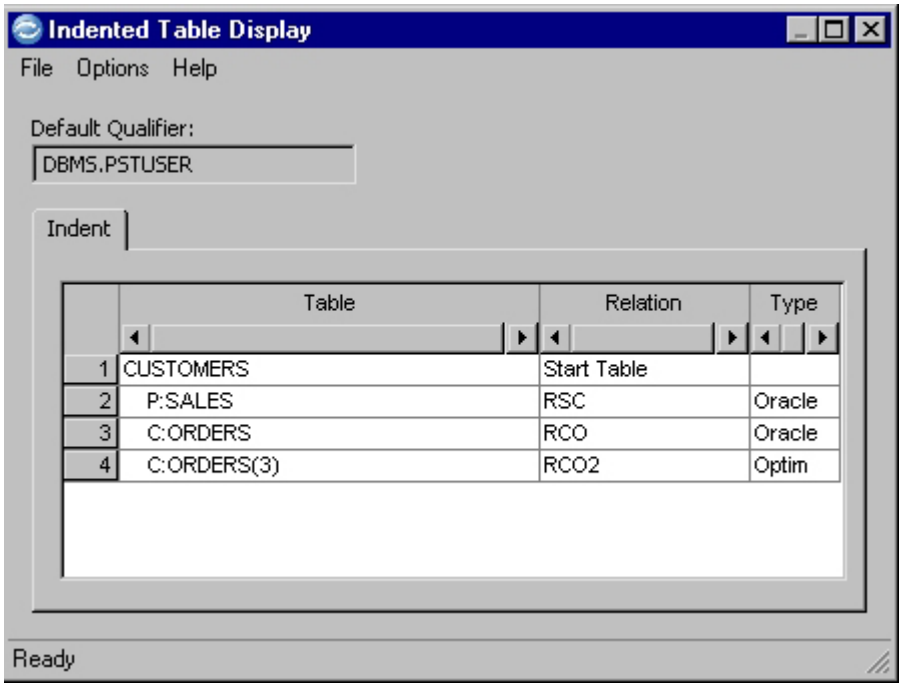


Table The names of the tables listed in the Access Definition.

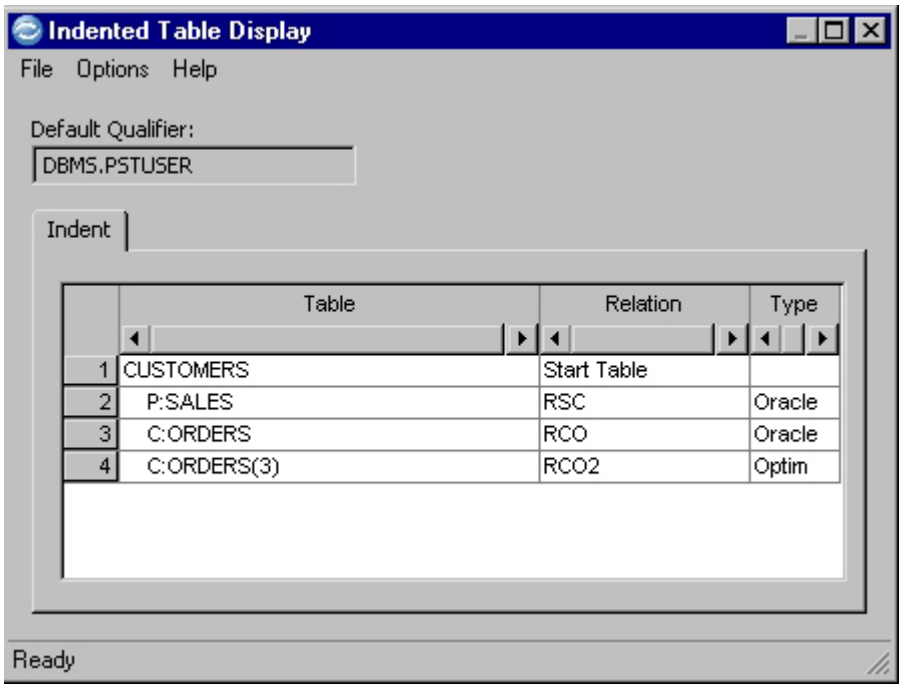
- The name of each table, except the Start Table, is prefixed with a **P** (parent table) or a **C** (child table) to show the relationship to the table under which it is listed.
- If a table is referenced several times (for example, a child table with three parents), the suffix (*n*) indicates the line number of the first occurrence.
- If a table is included in an RI cycle, the suffix (CYCLE:*n*) indicates the line number of other tables in the cycle.

Relation
Constraint name for the relationship with the table under which the table is listed. The constraint name for the Start Table is shown as **Start Table**.

Type The type of relationship as defined in the DBMS or the Optim Directory. This grid cell is blank for the Start Table.

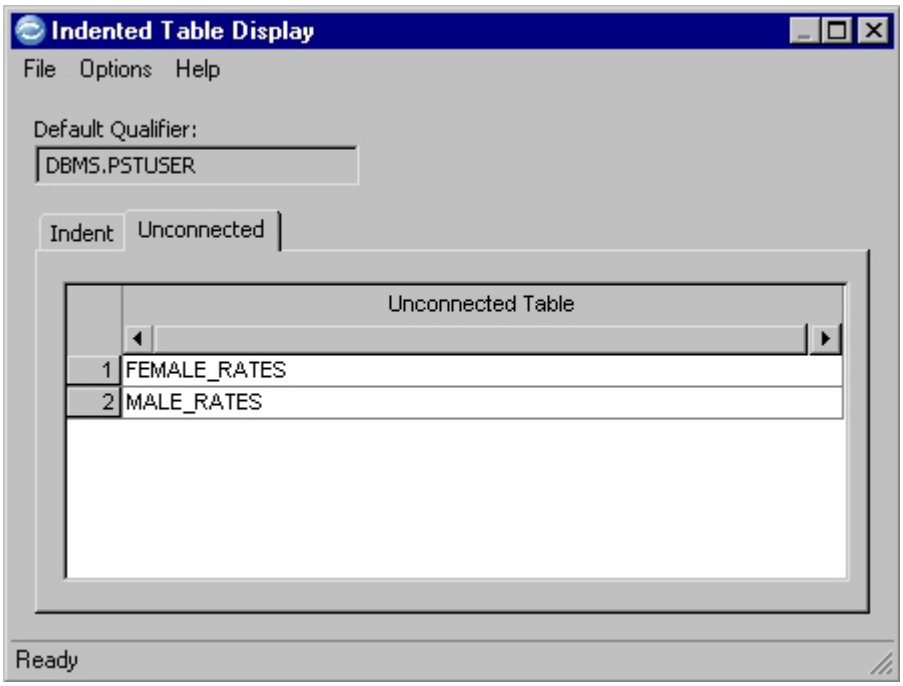
Reference Tables Tab

Tables designated as reference tables on the Access Definition Editor Table List. The **Reference Tables** tab is available only if you designated one or more reference tables.



Unconnected Tab

Tables on the Access Definition Editor table List that are not related to any other table in the Access Definition. The **Unconnected** tab is available only if one or more tables in the Access Definition cannot be traversed during a process.

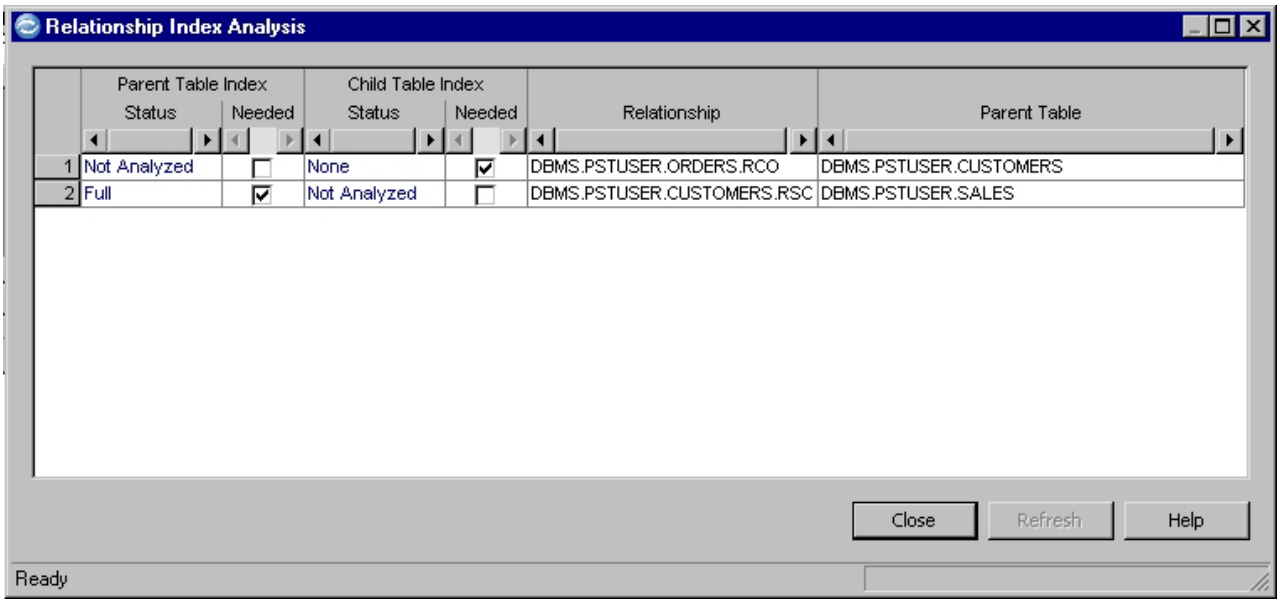


Relationship Index Analysis

One or more missing DBMS indexes is the most frequent cause of performance problems in an Archive or Extract Process. Use the Relationship Index Analysis dialog to analyze DBMS indexes for relationships used with the Access Definition and to create any needed indexes. The Relationship Index Analysis dialog lists each selected relationship in the Access Definition, with an analysis of DBMS indexes for the corresponding parent and child tables.

Note: Only indexes required for an Extract or Archive Process are analyzed.

Select **Relationship Index Analysis** from the **Tools** menu in the Access Definition Editor to display the Relationship Index Analysis dialog.



The relationship name and the name of the parent table are shown on the right side of the grid. The analysis is presented on the left.

Status

The status of DBMS indexes for the parent and child table columns in each relationship is displayed as follows:

None Necessary indexes do not exist.

Partial Necessary indexes for some relationship columns exist.

Full Necessary indexes for all relationship columns exist.

Indeterminate
Optim attempted to create DBMS indexes. Click **Refresh** to analyze the new index.

Not Analyzed
No indexes are needed.

If the analysis determines that a DBMS index for a parent or a child table is needed to increase the efficiency of processing, the **Needed** check box is selected. If the check box is cleared, the index was not analyzed, or is not needed.

Note: You cannot create an index for an expression in a relationship that is not a column name (e.g., a concatenation, literal, or substring). If a relationship includes an expression that is not a column name, only the column name expressions that precede the non-column name expression can be indexed.

Shortcut Menu

Using the shortcut menu, you can create needed DBMS indexes, and create, edit or browse relationships. Right-click the grid on the Relationship Index Analysis dialog to display shortcut menu.

If the status of an index is shown as **Partial**, or **None**, select from the following shortcut menu options to create necessary indexes:

Create All Indexes
From the submenu, select the DB Alias for the database to be indexed, displaying the Review Index SQL dialog. Generated SQL statements for creating the indexes are displayed.

Create Index
Select **Parent**, **Child**, or **Both** from the submenu to display the Review Index SQL dialog. (If parent and child tables have different DB Aliases, **Both** is disabled.) Generated SQL statements for creating the DBMS index are displayed.

Open Relationship
Open the Relationship Editor to create or modify an Optim relationship or browse a database relationship. See “Using the Editor” on page 217 for more information.

Note: If no expressions in a relationship can be indexed, a warning message is displayed.

In a generated SQL statement, the default name for a new index is in the form *identifier*.I_*tablenamnnnnnnnn*, where:

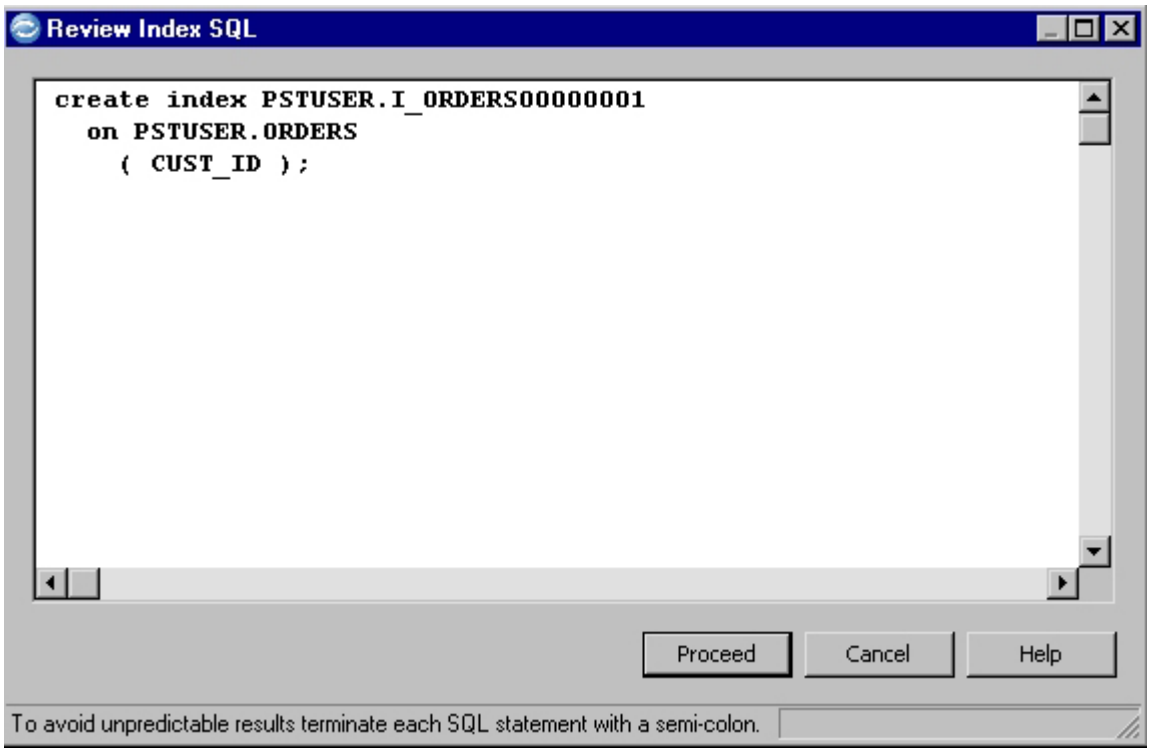
identifier
The identifier (Creator ID, Owner ID, or Schema Name) required by the DBMS to allow access to the database.

I_ Prefix for index name.

tablenam
First eight letters of the table name.

nnnnnnnn
Eight-digit, sequential number.

Review Index SQL



New indexes are named using the identifier (Creator ID, Owner ID, or Schema Name), followed by the letter "I", the first eight letters of the table name, and end with an eight-digit number, as follows:

identifier.I_*tablenamnnnnnnnn*

The Review Index SQL dialog also indicates if an expression in a relationship cannot be indexed.

Modify the name of the index, or other parts of the statements as necessary, then click **Proceed** to create the indexes.

The Browse Output dialog displays the results, after the SQL statements are executed.

Show Steps

Select **Show Steps** from the **Tools** menu to analyze the traversal path to be used in the process. Use **Show Steps** to evaluate the tables and relationships and display the steps performed to select the data. This evaluation can ensure that you retrieve the desired set of data.

Example

Assume that you want to select data from three tables: CUSTOMERS, ORDERS, and DETAILS. The CUSTOMERS table is the Start Table and rows are selected using Point and Shoot.

The ORDERS table is a child of CUSTOMERS, and DETAILS is a child of the ORDERS. The steps in the process are:

1. Extract Rows from Start Table DBMS.PSTUSER.CUSTOMERS. No Point and Shoot List, Selection Criteria or Statistical Controls used, therefore Start Table need not be revisited, even if part of a Cycle.
2. Extract Rows from DBMS.PSTUSER.ORDERS which are Children of Rows Previously Extracted from DBMS.PSTUSER.CUSTOMERS in Step 1 using Relationship RCO.
3. Extract Rows from DBMS.PSTUSER.DETAILS which are Children of Rows Previously Extracted from DBMS.PSTUSER.ORDERS in Step 2 using Relationship ROD.

Increasing Complexity

The steps in a process may be repeated any number of times according to relationships you select and criteria you specify.

Note: When a relationship is traversed to select parent rows for child rows already selected, any selection criteria for the parent table are ignored.

In a single step, rows may be selected from more than one table, or to satisfy more than one relationship. For example, if the DETAILS table is related to both the CUSTOMERS table and the ORDERS table, Step 3 is similar to the following:

- Extract Rows from DBMS.PSTUSER.DETAILS which are Children of Rows Previously Extracted from DBMS.PSTUSER.CUSTOMERS in Step 1 using Relationship RCD. Extract Rows from DBMS.PSTUSER.DETAILS which are Children of Rows Previously Extracted from DBMS.PSTUSER.ORDERS in Step 2 using Relationship ROD.

Any table may be revisited several times in successive steps. Cycles may also be involved. A cycle causes a set of tables to be traversed repeatedly until a complete pass through the cycle does not result in selecting additional rows.

Edit Point and Shoot List

Select **Edit Point and Shoot** from the **Tools** menu to display the Point and Shoot Editor. You can use the Point and Shoot Editor to create or edit a Point and Shoot list or to browse data to analyze the effects of criteria as you define it.

Note: You can use selection criteria to limit or rearrange the data displayed in the Point and Shoot Editor and column and sort specifications to arrange the display. For details, see “Table Specifications” on page 76.

Start Table rows that satisfy selection criteria, if any, are displayed. If the number of rows that satisfy the selection criteria exceeds the fetch limit, a message in the message bar alerts you that more rows exist than were retrieved. To retrieve more rows, you can increase the Maximum Fetch Rows value on the **Display** tab of the Personal Options dialog.

If you are editing a Point and Shoot list, and the list includes rows that do not satisfy any selection criteria, a prompt displays to allow you to retain these ‘unknown' rows or retain only rows that meet the current criteria.

If you create or modify a Local Point and Shoot list, you must select **Update** for the File menu to revise the list, and then save the Access Definition.

Primary Keys

The Start Table must have a primary key for Point and Shoot to function. The process uses primary key values to identify Start Table rows to be processed. The primary key values of the rows selected using Point and Shoot are saved in the Point and Shoot list.

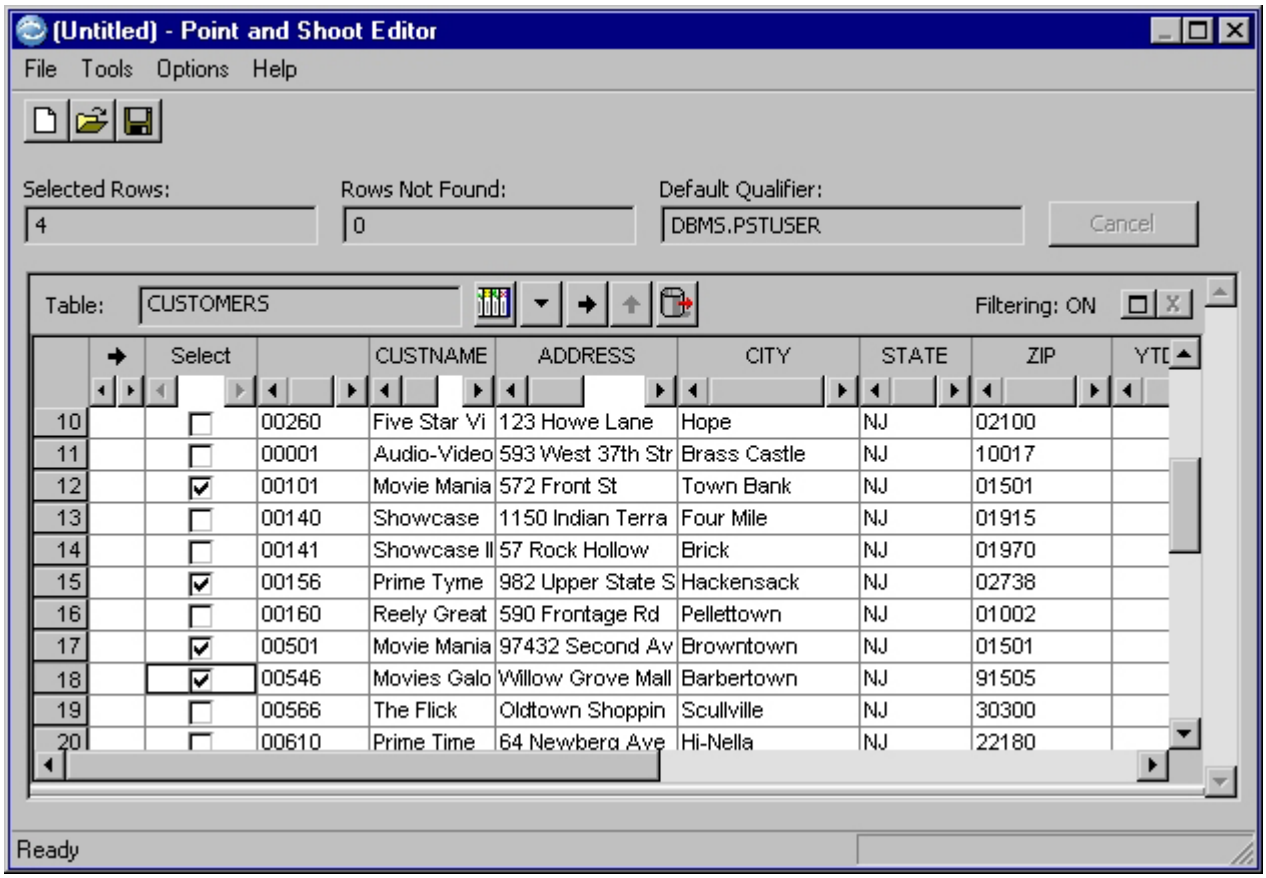
Processing Rules

During processing, the following rules pertain to Point and Shoot lists:

- If you specify selection criteria and use a Point and Shoot list, the process retrieves rows that match the selection criteria OR the Point and Shoot list.
- You can join other tables to the display for reference, but you can only select rows for the Point and Shoot list from the Start Table.
- If you use a Point and Shoot list, parameters defined for **Every Nth** and **Row Limit** on the **Tables** tab of the Access Definition Editor are ignored.

Point and Shoot Editor

The Point and Shoot Editor displays data from the Start Table in a browse window. A toolbar in the browse window allows you to select display options and menu choices that pertain to the display.



Selected Rows

Number of rows you select from the Start Table to include in the Point and Shoot list.

Rows Not Found

Number of rows in the Point and Shoot list that are not found among the rows retrieved for the current Point and Shoot session. A number greater than zero may indicate that the Default Qualifier has changed since the Point and Shoot list was created or that the Point and Shoot list includes rows that do not match current selection criteria.

Default Qualifier

The DB Alias and Creator ID of the Start Table.

Browse Window

The first browse window in the Point and Shoot Editor displays the Start Table rows. The Start Table browse window contains a grid column labeled **Select** that contains a check box for each row. Select the check box to include the row in the Point and Shoot list. Clear the check box to retract a selection.

You can join and view data from related tables in additional browse windows, using the **Join** button on the toolbar. Joining tables is useful for inspecting related data to ensure that appropriate sets of data are selected.

Data displayed in the table named in the browse window can be navigated and customized using the **Find**, **Exclude**, **Include**, **Hide**, and **Lock** options available on the grid heading shortcut menu. (See "Using Find and Replace" on page 23 for additional information.)

A browse window contains the following components:

Table Displays the name of the table shown in the browse window. The first browse window in the Point and Shoot Editor always contains the Start Table.

Format

or 

Click the **Format** button to switch the data display between columnar and side label format. The default format is set in Personal Options.

In *columnar format*, the column names are displayed across the top of the browse window and the data is displayed in columns beneath the headings. Note that the headings for primary key column(s) display in bold type.

Note: The **Select** grid column, with check boxes for selecting rows for the Point and Shoot list, is available in *columnar format* only.

In *side label format*, the column names are displayed down the left side of the browse window and a row of data is displayed to the right of the headings. Use the navigation buttons on the browse window toolbar to scroll to another row. Side label format focuses on a single row and can display more columns per row than columnar format.

Options

Click the **Options** button to display the browse window **Options** menu.

Display Attributes
Switch between displaying and hiding column attribute information in the column headings.

Sort You can sort rows according to values in one or more columns. Select **Sort** from the **Tools** menu in the Point and Shoot Editor to display a dialog with controls similar to those on the **Sort** tab in the Access Definition Editor. See "Sort Tab" on page 83 for information on Sort controls.

Select All Rows
Selects all rows displayed in the browse window.

Invert All Row Selections
Select this command to reverse selected and unselected rows.

Deselect All Rows
Clears all previously selected rows.

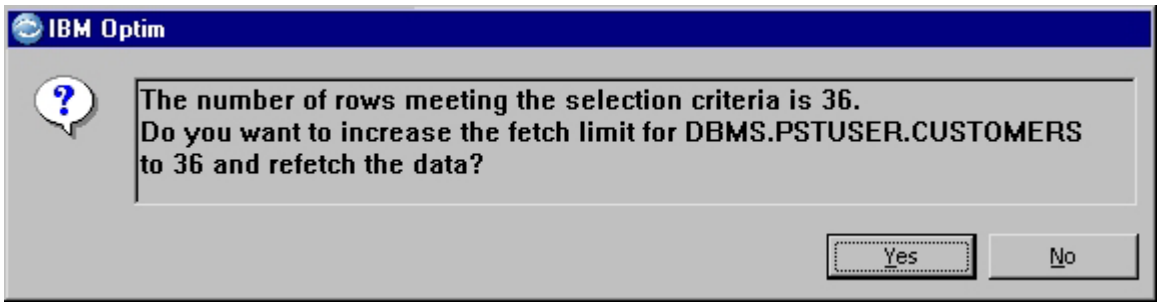
Show SQL
Displays a dialog containing the SQL used to obtain the fetch set. You can use the SQL dialog to review, print, and save SQL used to retrieve the current fetch set for the table in the active Edit Definition. To save or print SQL, select the appropriate command from the File menu on the SQL dialog.

List Constraints
Displays the Select a Constraint dialog. The Select a Constraint dialog provides information about each of the defined constraints for the specified table. Double-click a constraint name to view additional details in the Constraint Display dialog.

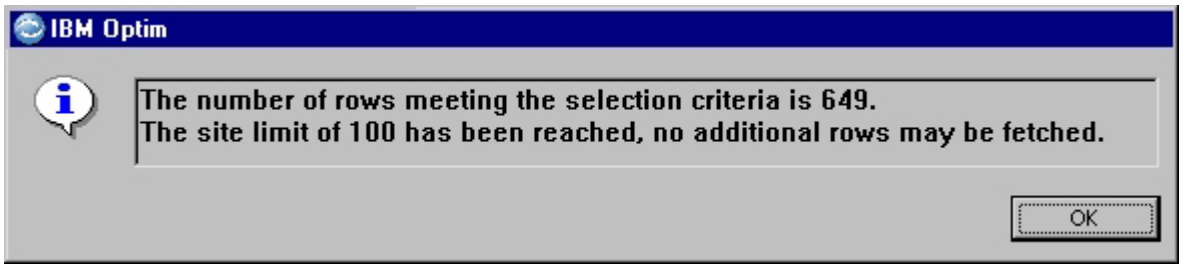
Show Excluded Rows
Displays all previously excluded rows (rows are excluded using the **Exclude** command on the shortcut menu). To display excluded rows individually, right-click a row and select **Show Next** from the shortcut menu.

Rows meeting Criteria
Displays a message to indicate the number of rows that meet the selection criteria. You can choose to increase the fetch limit for the table up to the maximum established in Product Options. (See the *Installation and Configuration Guide* .)

For example, when the number of rows that satisfy the criteria is 3523, but the fetch limit is 1000 and can be increased, the following is displayed:



However, if the fetch limit is at maximum, the following is displayed:



Unsupported Columns
Indicates there is at least one unsupported data type in the table. Select Unsupported Columns to display a dialog that contains a list of the unsupported columns and the corresponding data type for each.

Join

Click the **Join** button to join and view data from a related table in the Browse Table Data dialog. If more than one table is related to the table from which you join, or more than one relationship exists for a pair of tables, you must select from a selection list, as described under "Display Multiple Tables (Join Tables)."

Unjoin

Click the **Unjoin** button to remove a table and all subordinate joined tables from the Point and Shoot Editor.

Refetch Rows

Click the **Refetch Rows** button to retrieve a new fetch set of rows for the table. If other users are simultaneously accessing this data, refetch rows periodically to ensure you have current data.

Navigation

Click the **Navigation** buttons to scroll side label display to the first row, previous row, next row, or last row, respectively.

Filtering
Indicates whether Table Specifications, such as Selection Criteria, are defined for the table. Filtering OFF indicates no criteria are defined.

Current Row Indicator

When more than one table is displayed in the Point and Shoot Editor, position the Current Row Indicator to display related rows in joined table(s). To move to a different row, click the grid cell for the desired row or use the up/down arrows on your keyboard.

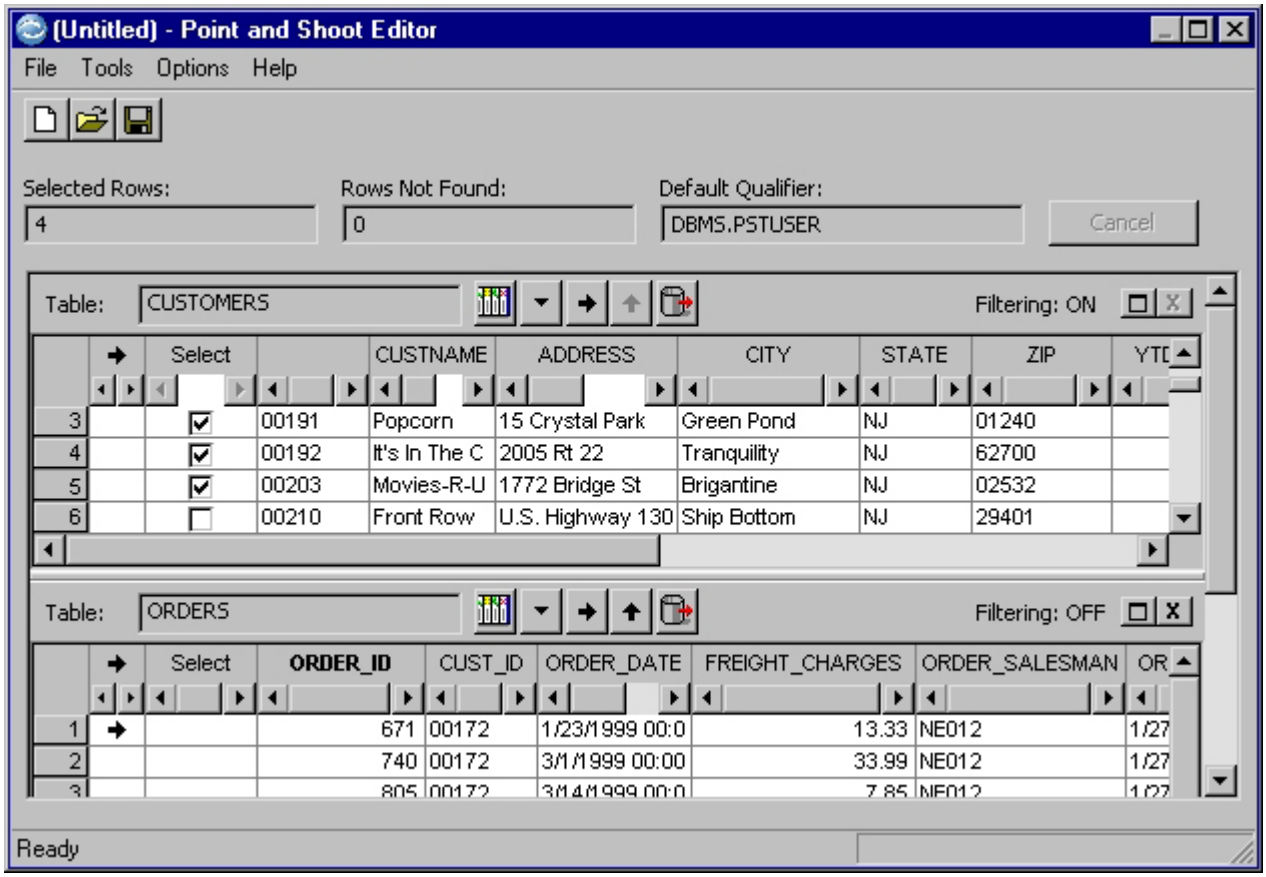
Select Select the check box to add the corresponding Start Table row to the Point and Shoot list.

Display Multiple Tables (Join Tables)

You can join multiple tables to any table in the Point and Shoot Editor. When you open the Point and Shoot Editor, the Start Table displays.

Use the Join function to display related data from other tables. When you join tables, the related data in the joined table displays in a new browse window in the Point and Shoot Editor. A relationship must exist between the tables to join them. You can join several tables to a single table, or join additional tables to each joined table. Each joined table displays in a new browse window.

Note: Rows can only be selected for a Point and Shoot list from the Start Table.



When you join tables, a relationship between the tables is required. Only one relationship can be used.

- If a relationship exists between the tables, the table joins automatically and related rows display in a new browse window.
- If a relationship does not exist, the Create a New Relationship dialog opens.
- If more than one relationship exists, the Select a Relationship dialog opens. You can select the relationship to use from a list.

You can also join more than one table to any table. When several tables are joined to a single table, the joined tables are "stacked" in a single edit window, in the order in which they were joined. The most recently joined table is displayed by default, and the other tables in the stack are hidden.

The name of the displayed table appears in a drop-down box in the browse window. Click the arrow to display the list of tables stacked in the browse window. Click a table name in the list to display that table in the browse window.

You can display any table in the stack and join other tables to any table in the stack. In many cases, a database table is related to two or more tables, creating different paths for joining and browsing the data.

When a stacked table is displayed, all subordinate joined tables are also displayed. When a stacked table is hidden, all subordinate joined tables are also hidden.

Use the Find Dialog

When editing a Point and Shoot list, use the Find dialog to find and select specific rows. Options allow you to select and unselect rows containing specific data. To display the Find dialog, right-click a grid column heading in the Point and Shoot Editor and select **Find**.

The Find dialog has three tabs, **Criteria**, **Columns**, and **Selection**. Use these tabs to specify search parameters and selection options. The **Find** parameters operate together to identify and select rows.

Command Buttons

Find Next

The function of the **Find Next** button varies according to the Find dialog tab:

- On the **Criteria** and **Columns** tabs, **Find Next** locates the next row that satisfies your search criteria.
- On the **Selection** tab, **Find Next** selects or unselects rows that satisfy the search criteria.

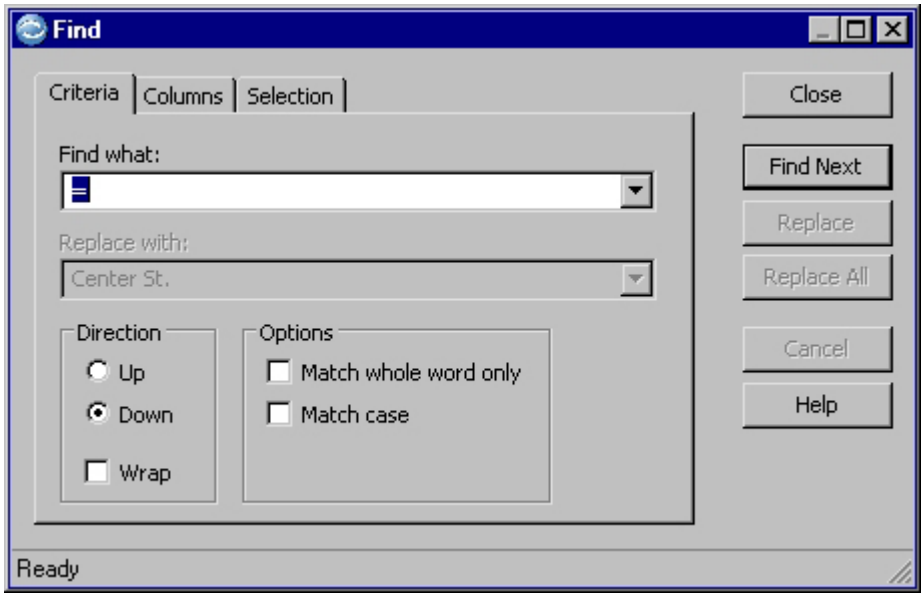
Replace

Replace All

The **Replace** and **Replace All** buttons are disabled when using **Find** with the Point and Shoot Editor. For information on these buttons, see Chapter 2, "Main Window, Menus, and Dialogs," on page 5.

Criteria Tab

Use the **Criteria** tab to define criteria for the data to find.



Find what

Enter a search value. To select from a list of recently specified search values, click the down arrow.

Direction

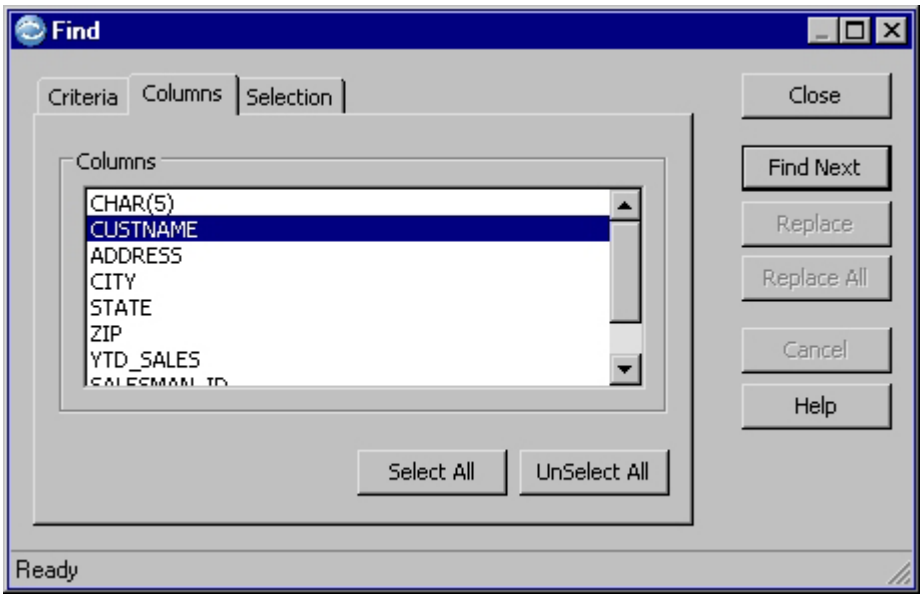
Choose a direction, from the cursor, for the search. Select **Wrap** to continue searching up (or down) after you reach the first row (or the last row) in a grid.

Options

Choose qualifiers for the search value, as required.

Columns Tab

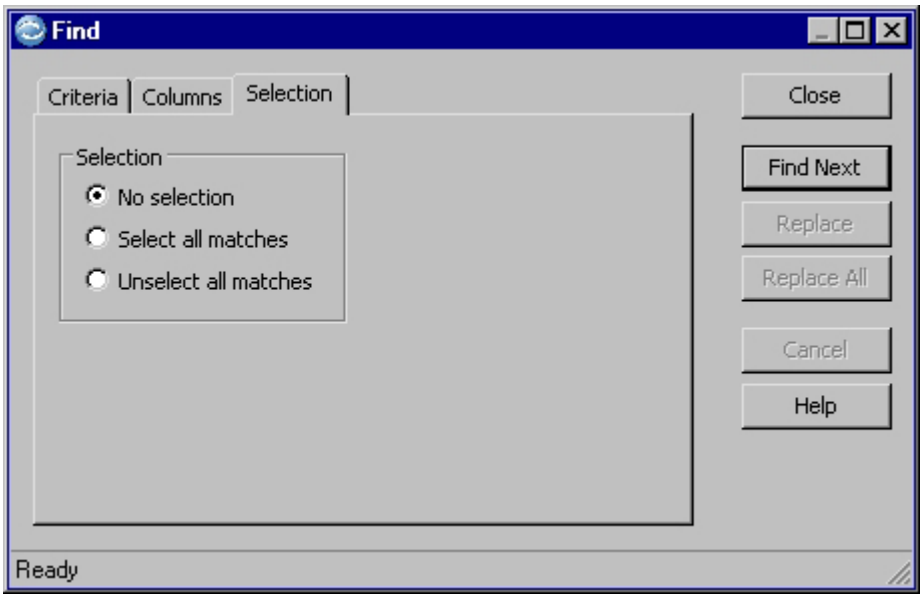
Use the **Columns** tab to select columns in the Start Table to limit the scope of the search.



- Select column name to search or click **Select All** to search all columns.
- Click **UnSelect All** to reset the list of selected columns.

Selection Tab

Choose options for selecting search values in the Point and Shoot list.



No selection
Select this option and use **Find Next** to find each occurrence of the search value. The Point and Shoot list is not affected.

Select all matches
Select this option and click **Find Next** to include rows that contain the search value in the Point and Shoot list.

Unselect all matches
Select this option and click **Find Next** to exclude rows that contain the search value from the Point and Shoot list.

Convert a Point and Shoot List

You can convert a Local Point and Shoot list to a named list, or a named list to a Local list. A Local Point and Shoot list is stored as part of the Access Definition and is available only to that Access Definition. A named Point and Shoot list is a file stored in the Optim Directory and can be used with other Access Definitions and shared with other users.

- To save a Local Point and Shoot list as a file, select **Save As** from the **File** menu in the Point and Shoot Editor to open a Save dialog. Specify a directory location and name for the Point and Shoot File and select **Save**.
- To convert a Point and Shoot File to a Local list, select **Convert to Local** from the **Tools** menu in the Point and Shoot Editor. After you close the Point and Shoot Editor, select **Save** from the **File** menu in the Access Definition Editor to save the Local Point and Shoot list with the Access Definition.

Using Point and Shoot

To facilitate row selection you can temporarily specify selection criteria to limit the list to specific rows and use it in combination with Point and Shoot Editor commands.

As an example, assume you are creating a test database of ORDERS and related DETAILS. To meet your testing requirements, you need orders posted more than 3 years and 1 month ago, but less than 4 years ago. Most orders that meet your requirements are for salesmen with an identifier that begins with the letters SE, but several are for two salesmen with the identifiers, NE005 and NC005. First, open **Table Specifications** to define selection criteria for ORDERS, the Start Table. Select the option on the **Selection Criteria** tab to combine all column criteria with AND. As criteria for the ORDER_POSTED column, specify:

BEFORE (3Y 1M)

and, as criteria for the ORDER_SALESMAN column specify:

LIKE ‘SE%'

Without closing Table Specifications, select the Access Definition Editor and choose **Edit Point and Shoot List** from the **Tools** menu to open the Point and Shoot Editor. ORDERS rows for orders posted before February 1998 for salesmen with an identifier beginning with SE are displayed.

Use the **Options** menu **Select All Rows** command to select all displayed rows. You can then use **Find** to locate and unselect rows for which the order was posted in 1997. Close the Point and Shoot Editor, saving the file. On the Table Specifications dialog, change the criteria for the ORDER_SALESMAN column to:

IN (‘NE005', ‘NC005')

and again choose **Edit Point and Shoot List** from the **Tools** menu on the Access Definition Editor. Click **Yes** at the prompt asking if you want to keep unknown entries. When the Point and Shoot Editor opens, ORDERS rows for orders posted before February 1998 for salesmen with the identifiers NE005 and NC005 are displayed. Select the rows appropriate for the test database and, again, close the Point and Shoot Editor, saving the file.

The Point and Shoot File now contains primary key values for all desired ORDERS rows. This file can be referenced in an Access Definition or Extract Process Request used to extract rows from the production database and the resulting Extract File used to create the test database.

Database Changes

The tables and relationships specified in an Access Definition can be affected when the database changes. The effect of these changes on an Access Definition depends on the change and whether a changed object is referenced in the Access Definition.

Table Added

Adding tables to the database has no effect on an Access Definition or any process request.

Table Deleted

If a table referenced in an Access Definition is dropped from the database, the status for that table in the Access Definition Editor is **Unknown** and a warning message is displayed.

Although you can use an Access Definition with a table in Unknown status, it is prudent to verify that the Access Definition describes the desired data. If the Start Table is dropped, you must select a different Start Table.

Column Added

If a column is added to a database table, the new column is included in the Access Definition automatically.

Column Modified

Changes to column attributes (data length or data type) in a database table affect an Access Definition. You can use the Access Definition; however, if you use it in a process, a warning is displayed.

Column Deleted

If a column in a database table referenced in an Access Definition is deleted, the effects vary:

- If the column is referenced explicitly (included in the selection criteria, a relationship, or the primary key), an error message is displayed. You must remove any explicit references to the deleted column before you can use the Access Definition.
- If the column is not referenced explicitly, a warning message is displayed, but you can use the Access Definition in a process. Verify that the Access Definition describes the desired data.

Relationship Added

New relationships are included in the Access Definition automatically and are assigned **New** status. By default, an Access Definition uses all new relationships, unless you specify otherwise.

Relationship Modified

Modified relationships are assigned **New** status in the Access Definition. An Access Definition uses all new relationships, unless you specify otherwise.

Relationship Deleted

Deleted relationships are assigned **Unknown** status in the Access Definition and a warning message indicates that the relationship is not found.

Note: Relationships that have **Unknown** status are ignored during a process.

Chapter 5. Column Maps

A Column Map provides specifications needed to match or exclude columns from processing. A Convert, Create, Insert, Load, Restore, or multi-table Compare Request must reference a Table Map, which may reference one or more Column Maps. A Compare Request for a single table comparison may reference a Column Map directly.

A Column Map must be used when column names or attributes are dissimilar, when data transformations (e.g., for data privacy) are needed, or when excluding one or more columns from processing. A Column Map referenced in a Convert, Insert, Load, or Restore Process can modify data, age dates, or convert currency. Advanced methods allow you to split data from one table into several tables or copy one source column to several destinations. A Column Map referenced in a Compare Process cannot transform data, however.

For data transformations that are beyond the scope of a Column Map, you can specify an exit routine or a Column Map Procedure for a source column. (Exit routines, programs that conform to the C programming language, are discussed in Appendix B, "Exit Routines for Column Maps," on page 473. Column Map Procedures are discussed in Chapter 6, "Column Map Procedures," on page 175.)

When you create a new Column Map, you must choose the source for the columns you want to map. The source can be either a table in an Extract or Archive File or a table in a database. Similarly, you must specify a destination database table.

Column Maps stored in the Optim Directory are available for reuse or sharing with other users. A local Column Map is stored as part of a Table Map or Compare Request and is otherwise not available. If the Table Map is Local to a process request, both the Table Map and Column Map are available only to the specific process request.

Table Maps

A Table Map provides a way to control or direct the placement of data in a Convert, Create, Insert, or Restore Process, and to exclude one or more tables from Compare, Convert, Create, Insert, or Restore Processing. (For details, see "Create a Table Map" on page 229.) Specify Column Maps in a Table Map to control processing, column by column.

Naming Conventions

The fully qualified name of a Column Map has two parts: *identifier.name*.

identifier
: Identifier assigned to the Column Map (1 to 8 characters).

name Name assigned to the Column Map (1 to 12 characters).

When you create Column Maps, it is helpful to use a logical set of naming conventions to identify and organize definitions for easy access.

Contents

This section provides detailed information on how to:

- Create a new Column Map.
- Open an existing Column Map.
- Match columns that have different names.

- Show mapped columns by status.
- Specify special values to be inserted in destination columns using column names, NULL, constants, literals, functions, expressions, special registers, or exit routines.
- Verify supported classes of data and supported data types.
- Verify compatibility of source data types to destination.
- Save a Column Map.
- Delete a Column Map.

Open the Column Map Editor

Use the Column Map Editor to create and maintain Column Maps. There are different ways to open the editor depending on whether you want to create a new Column Map or select a Column Map to edit.

Create a Column Map

This section explains how to create a Column Map from the main window or the Column Map Editor.

From the Main Window: Move or Archive

To create a Column Map for Move or Archive:

1. In the main window, select **New** from the **File** menu.
2. Select **Column Map** from the **Definitions** submenu to open the New Column Map dialog.
3. For Validation Rules, select **Move/Archive**.
4. Specify the name of the source table containing the columns you want to map:
   - If you select the **File** option, supply the file name and specify the name of a table in that file.
   - If you select the **Database** option, supply the name of the database table.
5. Specify the name of the destination table containing the columns you want to map.
6. Click **OK** to open the Column Map Editor.
7. Modify the Source Column names, as needed.
8. Select **Save** from the **File** menu to open the Save the Column Map dialog.
9. In the **Pattern** box, specify a unique Column Map name and then click **Save**.

Note: The pseudocolumns that represent attached files do not appear in a Move or Archive Column Map. However, the pseudocolumns can be mapped as source columns and will be used whether or not file attachments are processed.

From the Main Window: Compare

To create a Column Map for Compare:

1. In the main window, select **New** from the **File** menu.
2. Select **Column Map** from the **Definitions** submenu to open the New Column Map dialog.
3. For Validation Rules, select **Compare**.
4. Specify the name of the table containing the columns you want to map in each source:
   - If you select the **File** option, supply the file name and specify the name of a table in that file.
   - If you select the **Database** option, supply the name of the database table.
5. Click **OK** to open the Column Map Editor.
6. Choose Source 1 column names to map, as needed, by clicking on the column name and selecting from the list of columns available.
7. Select **Save** from the **File** menu to open the Save the Column Map dialog.
8. In the **Pattern** box, specify a unique Column Map name and then click **Save**.

From the Column Map Editor

To create a Column Map:

1. In the main window, select **Column Map** from the **Definitions** menu to open the Column Map Editorand last edited Column Map.
2. Your next step depends on your purpose:
   - To create a new Column Map, select **New** from the **File** menu in the Column Map Editor to open the New Column Map dialog. Specify the source/destination tables OR Source1/Source 2 tables, and click **OK** to open the Column Map Editor.
   - To create a new Column Map modeled on an existing one, open the desired Column Map and select **Save As** from the **File** menu in the Column Map Editor.
   - To create and store a copy of the active Column Map and continue editing, select **Save Copy As** from the **File** menu in the Column Map Editor.

Note: Pseudocolumns for attached files are automatically displayed in a Compare Column Map when Source 2 includes attached files. If Source 1 pseudocolumns are not displayed automatically, you can select them as Source 1 columns.

These steps are the minimum required to create a Column Map. The steps to edit a new Column Map or existing Column Map are similar. See "Using the Editor" on page 127 for details.

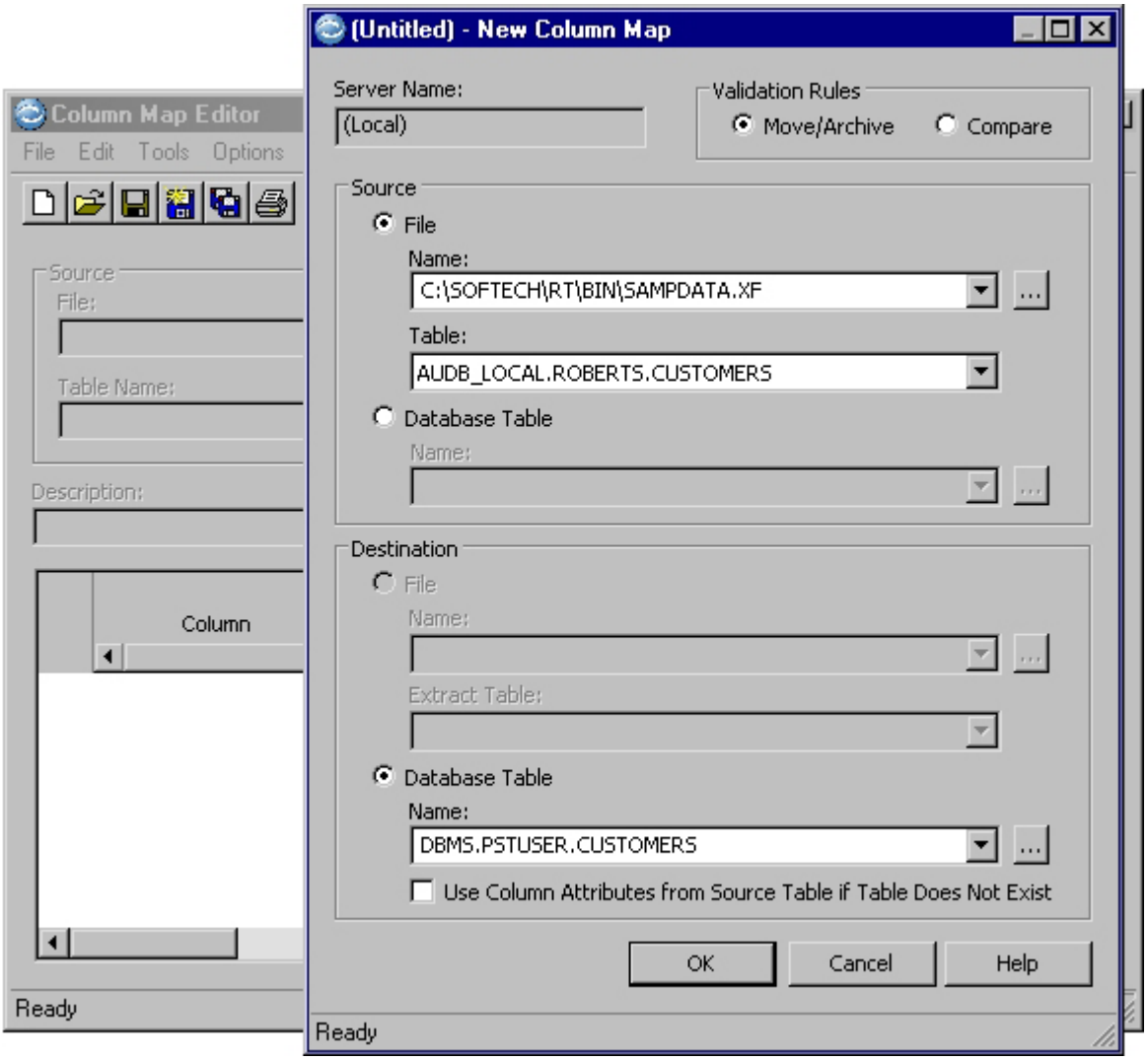
In addition, you can open the Column Map Editor from the **Tools** menu in the Convert Request Editor, Insert Request Editor, or Load Request Editor. For details, refer to the appropriate chapter in the corresponding user manual.

New Column Map Dialog

When you choose to create a new Column Map from the **Definitions** submenu, the Column Map Editor opens, along with the New Column Map dialog. You can also open the New Column Map dialog by selecting **New** from the **File** menu in the Column Map Editor.

(For a description of the Definitions submenu, see "From the Main Window: Move or Archive" on page 122.)

After you specify the appropriate parameters in the New Column Map dialog, the Column Map Editor is displayed.



Server Name

If the optional Optim Server is installed on your network, and the table containing the source column is in an Extract or Archive File, click the down arrow to select **Local** if the file is accessed from the client workstation, or the Server on which the file resides.

Validation Rules

If the Column Map is to be used in a Convert, Insert, or Restore Process, select **Move/Archive** validation rules. Select **Compare** for use in a Compare Process. The group box labels vary according to your selection. In the previous illustration, Move/Archive validation rules are selected and the group boxes are labeled Source and Destination. For Compare validation rules, the labels are Source 1 and Source 2.

Source or Source 1

When you create a new Column Map, you must indicate the source for the columns you want to map. You can specify the source as a table in an Extract or Archive File or the database:

File Select this option to map columns from a table in an Extract or Archive File.

Name Directory path and name of the file. If a directory path is not specified, the data directory path specified in Personal Options is used by default.

- To select from a list of recently used file names, click the down arrow.
- To select from another directory, click the browse button.

Table Name of the table in the file. To select from a list, click the down arrow.

Database Table
Select this option to map columns in a table that resides in the database.

Name Fully qualified name of the database table.

- To select from a list of recently used table names, click the down arrow.
- To select from a directory of existing tables, click the browse button.

Destination or Source 2

For Move/Archive validation rules, the destination can only be Database Table. For Compare validation rules, all Source 2 options apply.

File Select this option to map columns from a table in an Extract or Archive File. This option is enabled if Compare validation rules are selected.

Database Table
Fully qualified name of the destination table that is mapped to the source table.

Use Column Attributes from Source Table if Table does not exist
Select this check box to use column attributes from the source table to map columns for the destination table. Use this when defining a Column Map as part of a request and the destination table does not exist. This check box is enabled if Move/Archive validation rules are selected.

Enter the appropriate parameters, provide entries to define the source and destination tables OR Source 1 and Source 2 tables, and select **OK** to open the Column Map Editor.

Select a Column Map to Edit

You can select a Column Map for editing from the main window or from the Column Map Editor.

From the Main Window

To select a Column Map to edit:

1. In the main window, select **Open** from the **File** menu to display the Open (object selection) dialog.
2. Double-click **Column Map** to expand the Identifier list.
3. Double-click the desired identifier to display a list of Column Maps.
4. *Optional* – Specify a **Pattern** to limit the list based on your criteria and click **Refresh**.
5. Double-click the desired Column Map to open the Column Map Editor.

Note: To select the last Column Map you edited, select **Column Map** from the **Definitions** menu in the main window to open the Column Map Editor and last edited Column Map.

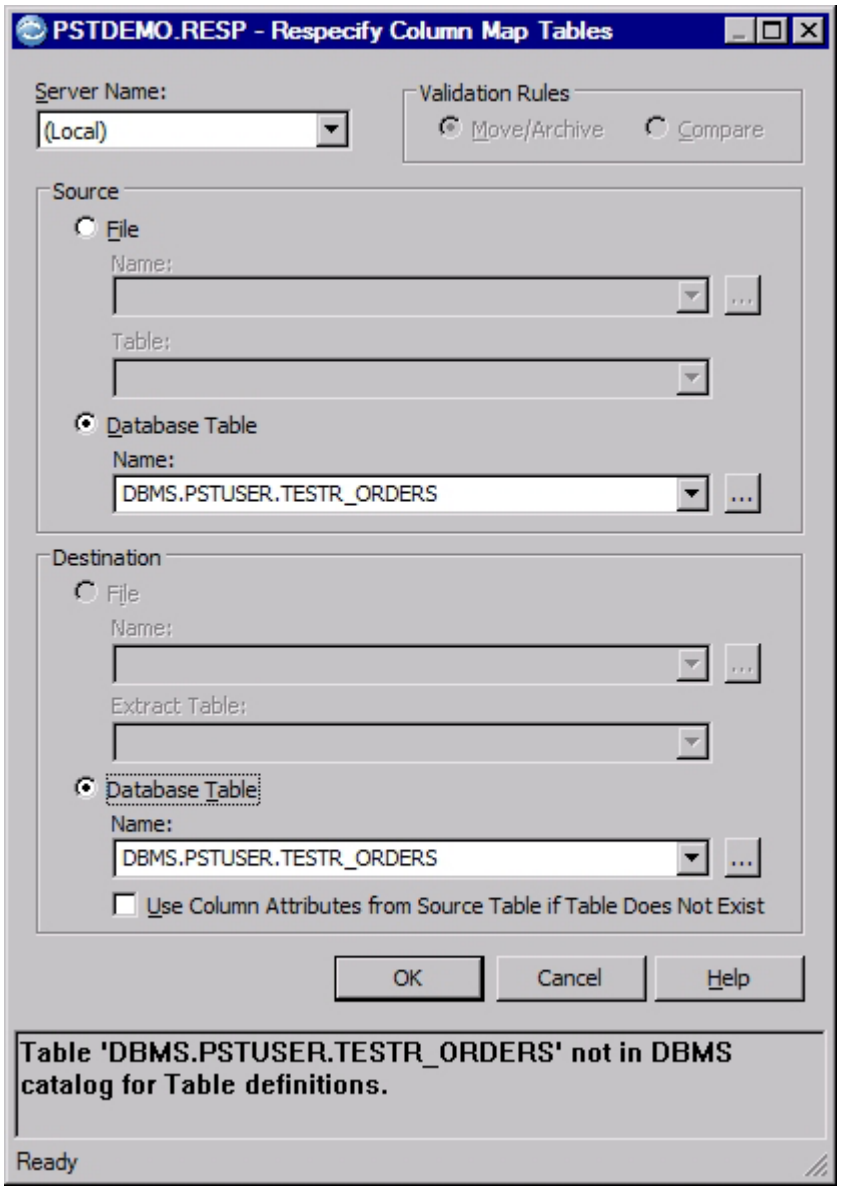
From the Column Map Editor

To select a different Column Map:

1. In the Column Map Editor, select **Open** from the **File** menu to display the Open a Column Map dialog. (See “Using the Open Dialog” on page 25 for more information on the Open dialog.)
2. Double-click the desired identifier to display a list of Column Maps.
3. *Optional* – Specify a **Pattern** to limit the list based on your criteria and click **Refresh**.
4. Double-click the desired Column Map to open the Column Map Editor.

Respecify Column Map Tables Dialog

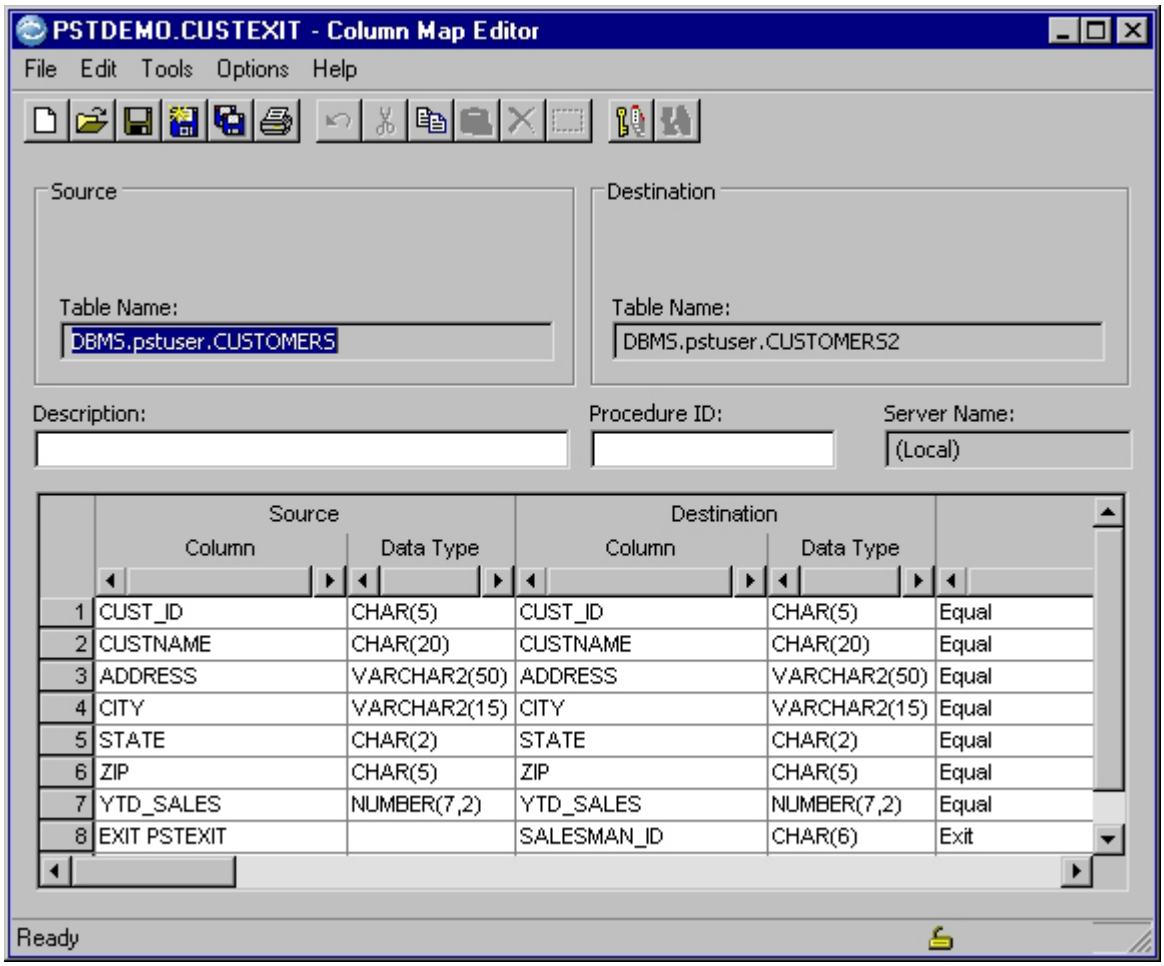
Before displaying a Column Map, Optim verifies that the source and destination tables exist. If any table (i.e., the Archive or Extract File or a database table used as a source or destination) does not exist, Optim displays the Respecify Column Map Tables dialog so you can provide the required information.



Edit the information as appropriate and click OK to display the Column Map in the Column Map Editor.

Using the Editor

In the Column Map Editor you can create, modify, or delete Column Maps stored in the Optim Directory or defined within a Table Map or Compare Request.



When you create a new Column Map, Optim populates the grid automatically on the basis of your selections in the New Column Map dialog. You edit the values in the first grid column to map columns. See "Create a Column Map" on page 122 for details. The names of the tables, for which columns are mapped, are displayed at the top of the Column Map Editor.

Source or Source 1
File name and fully qualified table name, or table name only.

Destination or Source 2
Fully qualified table name.

Description

Text to describe the purpose of the Column Map (up to 40 characters).

Procedure ID

The Procedure Identifier qualifies any Column Map Procedures specified by the base name only. If you specify a fully qualified Column Map Procedure name in the Column Map, the Procedure ID is ignored.

When you create a new Column Map, the Procedure ID is initially blank. Note that you cannot run a Column Map Procedure in a UNIX environment.

Server Name

The location of the Extract or Archive File containing the source table, if any.

Grid Details

The Column Map grid shows the mapped pairs of columns:

Source or Source 1 Column	Columns with the same name and compatible data types are automatically matched. If a match is not found for a Destination or Source 2 column, the status is NOT USED or REQUIRED. You can modify values in this grid column. A source column value in bold type can be viewed and edited from the Source Column dialog only. To open the dialog, double-click the value or use the Expand Source Column command. For more information, see "Expand Source Column" on page 134.		
Data Type	Data type for a named Source or Source 1 column.		
Destination or Source 2 Column	Names of the columns in the table. You cannot modify Destination or Source 2 column names.		
Data Type	Data type for each destination column.		
Status	Information about each pair of mapped columns. Status indicators are:		
	Equal	Columns have the same data type, and length.	
	Mapped	Columns do not have the same data type or length, but are compatible.	
	Not Used	No value for Source or Source 1 column. If you do not enter a value: • An Insert or Load Process inserts the default value defined to the database for the Destination column. • An Update Process does not update the value in the Destination column. • A Compare Process does not compare the value in the Source 2 column. • A Convert Process does not convert a value to the Destination column.	
	Unknown	*Unknown Source Column* *Unknown Destination Column*	
	Unknown status occurs when you apply the Column Map to a table that does not include the column.		
	Exit	An exit routine determines the value for the Destination column.	
	Expression or Function	Character Expression	The Source character expression determines the value placed in the Destination column.
		Numeric Expression	The Source numeric expression determines the value placed in the Destination column.
		Age Function	The Source age expression determines the value placed in the destination column.
		Currency	The Source currency expression determines the value placed in the destination column.
	Transformation	A data privacy transformation library function.	
	NULL	NULL is placed in the Destination column.	

	Special Register	Timestamp	The Timestamp determines the value for the Destination column.
		Workstation ID	The Workstation ID determines the value for the Destination column.
	Literal	String Literal	The Source string literal determines the value for the Destination column.
		Numeric Constant	The Source numeric constant determines the value placed in the Destination column.
		TRUE Bit Value	Boolean TRUE is the value for the Destination column.
		FALSE Bit Value	Boolean FALSE is the value for the Destination column.
	Not Inserted	Destination column cannot accept data. (**Move/Archive** validation rules only.)	
Error	Certain statuses alert you to errors in mapping columns. You must resolve errors in order to save a Column Map.		

Menu Commands

In addition to the standard commands on the **File**, **Edit**, and **Tools** menus, you can select the following commands from the **Tools** menu:

Show Open the Show Columns dialog and select options to list columns according to status. See "List Columns According to Status" for details.

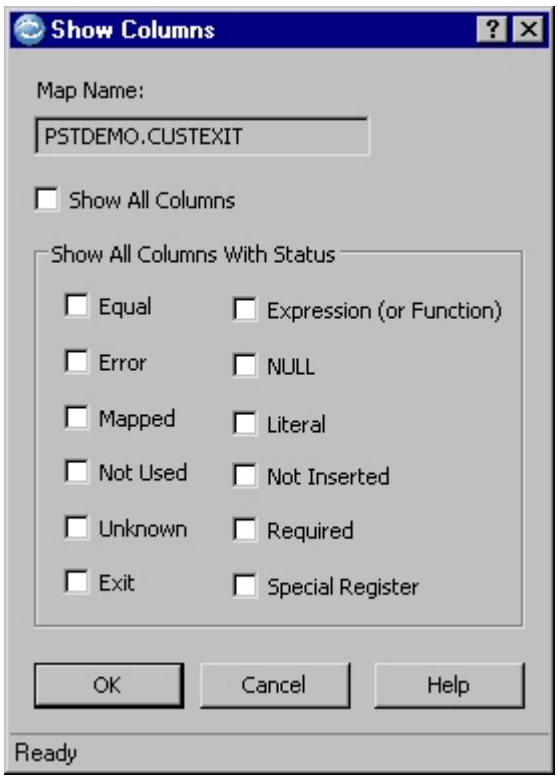
Delete Entry
Remove column names from the list. This command is enabled when a Destination or Source 2 column does not exist or is no longer valid.

Expand Source Column
Open the Source Column dialog for entering long Source Column values and functions. This command is available when **Move/Archive** validation rules are selected.

List Columns
Open the List Columns dialog and select from a list of Source column names. This command is available when **Move/Archive** validation rules are selected.

List Columns According to Status

As you edit the Column Map, the status of each pair of columns is updated automatically. To focus on a set of mapped columns according to status, select **Show** from the **Tools** menu to open the Show Columns dialog.



Map Name

Column Map name. A name is displayed after you name and save the Column Map.

Show All Columns

The Column Map Editor lists all Destination or Source 2 columns, by default. Clear the **Show All Columns** check box to enable status check boxes.

Show All Columns With Status

Select one or more check boxes to display columns with the indicated statuses. See the description for Status in "Using the Editor" on page 127 for a list of possible statuses and messages.

Specify Column Values in a Column Map

Rules for column maps differ between Compare and Move or Archive. These differences are discussed in the following section.

Compare Validation Rules

Compare automatically matches every column in the Source 1 table to the Source 2 column that has the same name and compatible data type. Click a Source 1 grid column to select a column name from the list of unmatched Source 1 columns. Select the blank space to exclude a column from the comparison. No other editing of Source 1 specifications is possible.

Move/Archive Validation Rules

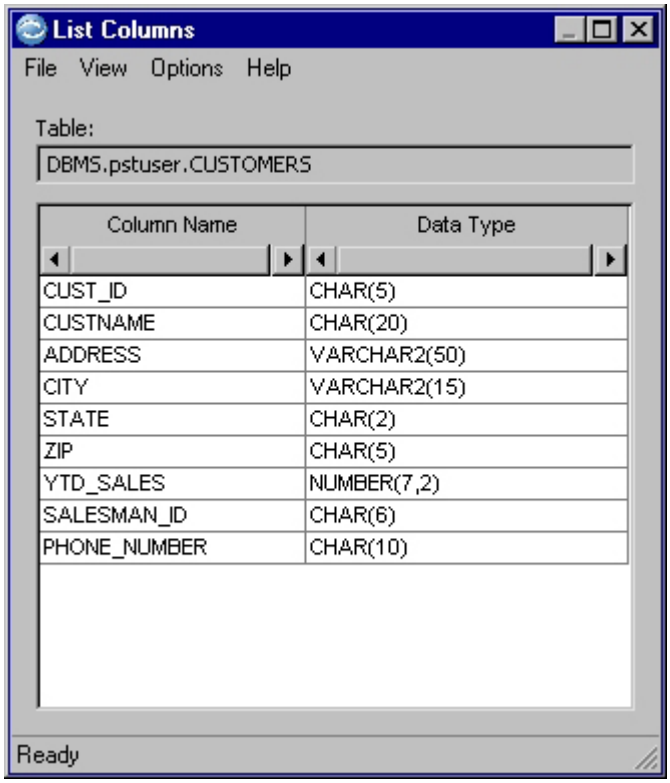
The Column Map Editor lists the pairs of columns that are mapped. You can edit Source specifications in the following ways:

- Add or replace column specifications by selecting column names from a list.
- Edit column specifications by typing a column name (up to 999 characters), a constant, NULL, a literal, a special register, a function, procedure, expression, or exit routine.

You can map a Source column to more than one Destination column as long as the data types are compatible.

Select Columns from a List

Right-click and select **List Columns** from the shortcut menu (or select **List Columns** from the **Tools** menu) to list columns from the source table. (The List Columns command is enabled when the Column Map uses Move/Archive validation rules.)



Table

Fully qualified name of the source table.

Column Name

Name of each column in the source table. To select a column, click and drag the column name from the List Columns dialog to the **Source Column** grid cell in the Column Map Editor. The status is updated automatically.

To change the column display, select commands from the **View** menu in the List Columns dialog, as follows:

- To view all columns in the Column Map, select **All**.
- To view all unused columns, select **Unused**. If the table contains many columns, it may be helpful to display only unused columns.

Data Type

Data type for each column in the source table.

Edit Source Column Names

Overtype source column values or right-click to select commands to **Clear**, **Find**, or **Replace** source column values. (If you clear a Source Column grid cell, the corresponding destination column value is not affected.)

A source column value in bold type can be viewed and edited from the Source Column dialog only. To open the dialog, double-click the value or use the Expand Source Column command. For more information, see "Expand Source Column" on page 134.

You can use any of several methods to map values to Destination Columns. Specify:

Column Name	An explicit column name (column names are case⌂insensitive).	
NULL	NULL. The destination column must be nullable.	
Numeric Constant	A numeric constant. The constant value must fit into the destination column as defined by its data type, precision, and scale.	
Boolean Constant	A Boolean constant (TRUE or FALSE).	
Special Register	A special register:	
	CURRENT DATE	CURRENT_DATE
	CURRENT TIME	CURRENT_TIME
	CURRENT TIMESTAMP	CURRENT_TIMESTAMP
	CURRENT SQLID	CURRENT_SQLID
	CURDATE( )	CURTIME( )
	GETDATE( )	GETTIME( )
	SYSDATE( )	NOW( )
	WORKSTATION_ID	USER
String Literal	A string literal, enclosed in single quotes. The destination column must contain character data. Example: 'CA' or '90210'.	
Hexadecimal Literal	A hexadecimal literal. Example: X'1234567890ABCDEF' or 0X1234567890ABCDEF	
Date/Time Literal	A date/time literal, enclosed in single quotes. Separate the date and time with a space. To format the date/time with a decimal fraction, place a period after the time, followed by the fraction. The date format is determined by the settings in **Regional Options** on the Control Panel of your computer. **Note:** For Oracle Timestamp with Time Zone columns, you must specify the time zone suffix last.	
"Substring Function" on page 135	A Substring Function to use a portion of a source column value.	
"Random Function" on page 135	A Random Function to generate a random value.	
"Sequential Function" on page 136	A Sequential Function to define an incremental sequential value.	

"Identity or Serial Function" on page 136	An Identity or Serial Function to direct the DBMS to supply a sequential value (integer) for a destination column.
"Oracle Sequence Function" on page 137	An Oracle Sequence Function to assign a value to a destination column using an Oracle Sequence.
"Propagate Primary or Foreign Key Value Function" on page 137	A Propagate Function to assign a value to a primary or foreign key column and propagate that value to all related tables.
"Concatenated Expressions" on page 139	A Concatenated Expression to obtain a derived value.
"Numeric Expressions" on page 139	A Numeric Expression to obtain a derived numeric value.
Column Map Procedure	A Column Map Procedure to obtain site-specific values. See Chapter 6, "Column Map Procedures," on page 175.
Exit Routine	An Exit Routine to obtain site-specific values. See Appendix B, "Exit Routines for Column Maps," on page 473.
Note: Information about functions and expressions is on the following pages. For details on using Exit Routines in a Column Map, see Appendix B, "Exit Routines for Column Maps," on page 473.	
Data Privacy Functions These functions require an Optim Data Privacy license. For detailed information, see "Data Privacy Functions" on page 140. To use these functions to map values to Destination Columns, specify:	
Lookup Functions	A Lookup Function to obtain the value for one or more destination columns from a lookup table.
Random Lookup Function	A Random Lookup Function to generate a random value for one or more destination columns from a lookup table.
Hash Lookup Function	A Hash Lookup Function to obtain the value for one or more destination columns from a lookup table, according to a hashed value derived from a source column.
Shuffle Function	A Shuffle Function to replace a source value with another value from the column.
Transformation Library Functions	The Transformation Library Functions to mask personal data such as a social security number, credit card number, or e-mail address.
Age Function	An Age Function to age the source column value.
Currency Function	A Currency Function to convert source column currency values.

Syntax Conventions

The syntax conventions used to describe the following functions are:

KEYWORD
Keywords are shown in uppercase for emphasis, but can be specified in lower or mixed case.

text Variable text is shown in lower-case italics.

[ ] Indicates an optional parameter.

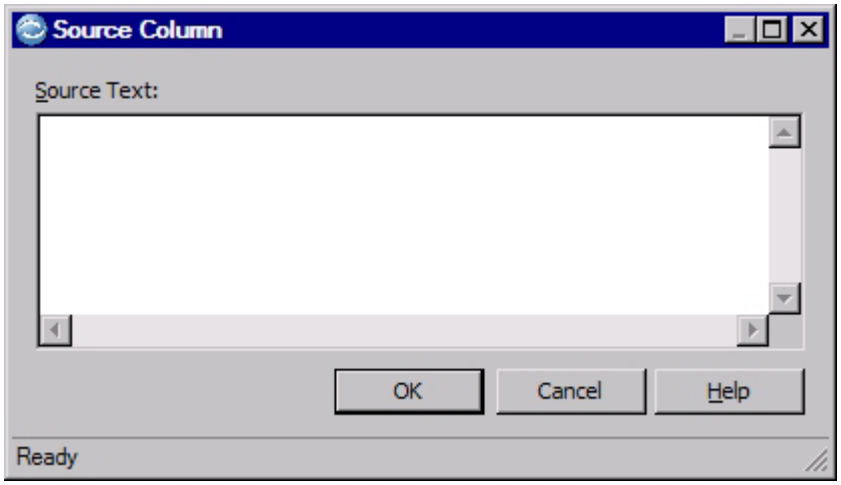
{ } Indicates a choice of two or more settings from which one (and only one) must be selected.

| Separates options.

Expand Source Column

Use the Source Column dialog to enter long source column values or functions up to 999 characters. The Source Column dialog is available only for Column Maps using Move/Archive validation and is not available for columns that contain a reference to a Column Map Procedure. You can enter a reference to Column Map Procedure in the Source Column dialog, but the dialog will not be available for the column after the value is defined.

To open the Source Column dialog, right-click the source column cell and select **Expand Source Column** from the shortcut menu (or place the cursor in the cell and select **Expand Source Column** from the **Tools** menu).



If the source column value includes carriage return/line feed pairs, the value is displayed in bold type in the Column Map and can be viewed and edited in the Source Columndialog only. Carriage return/line feed pairs are removed from a reference to a Column Map Procedure. Carriage return/line feed pairs are not displayed in the value and are created by using the Enter key to make a line break in the Source Columndialog (they may also be present in an imported Column Map). To open the Source Column dialog for a value in bold type, double-click the value or use the **Expand Source Column** command.

Enter text in the Source Text area. Click **OK** to exit the dialog and validate the entry (if the text is not valid, the dialog will remain open and display error messages). Click **Cancel** to exit the dialog and clear the entry, retaining the original source column value.

Shortcut Menu Commands

Right-click the **Source Text** area to display the following:

Restore Text to Original Value
Remove text entered in the dialog and display the original source column value. You will be prompted to confirm your selection.

Validate Text
Validate the value according to Move/Archive validation rules. The dialog will display error messages.

Substring Function

The Substring Function returns a substring of the contents of the named column. The syntax is:

SUBSTR(*columnname, start,* [*length*])

columnname
Name of a character or binary column.

start The position of the first character in the string.

length The number of characters to use.

- If the locale uses a comma as the decimal separator, you must leave a space after each comma that separates numeric parameters (for example, after the comma between *start* and *length*.
- *start* and *length* are integers greater than or equal to 1.
- *start* plus *length* cannot exceed the total data length plus 1.
- *column-name* and *start* value are required. If you specify only one integer, it is used as the *start* value. The substring begins at *start* and includes the remainder of the column value.

Example

If the PHONE_NUMBER column is defined as CHAR(10), you can use the Substring Function to map the area code. To obtain a substring of the first three positions of the phone number (area code) for the destination column, specify:

SUBSTR(PHONE_NUMBER, 1, 3)

Random Function

The Random Function returns a number selected at random within the range indicated by the low and high values. The syntax is:

RAND(*low, high*)

low Lowest possible random value.

high Highest possible random value.

- Use the Random Function with character or numeric data.
- If the locale uses a comma as the decimal separator, you must leave a space after the comma.
- *low* and *high* are integers within the range **-2,147,483,648** to **2,147,483,647**.
- *low* and *high* are further limited by the data type and length for the destination column.
- *low* must be less than *high*.
- When you use the Random Function in a concatenated expression, a variable length string is returned.

Example

You can use the Random Function to mask or change sales data for a test database. Assume the YTD_SALES column is defined as DECIMAL(7,2). The maximum number of digits to the left of the decimal is 5; the possible range for this column is -99999 to 99999. To create test data within a range from 1000 (low) to 89999 (high), specify:

RAND(1000, 89999)

In this example, the function returns random sales values within the range you specified from 1000.00 to 89999.99.

Sequential Function

The Sequential Function returns a number that is incremented sequentially. The syntax is:

SEQ(*start, step*)

start Start value.

step Incremental value.

- Use the Sequential Function with character and numeric data.
- If the locale uses a comma as the decimal separator, you must leave a space after the comma.
- *start* and *step* are integers within the range of **-2,147,483,648** and **2,147,483,647**.
- *start* and *step* are further limited by the data type and length of the destination column.
- If the calculated value exceeds the length of the destination column, the function automatically resets to the *start* value.
- When you use the Sequential Function in a concatenated expression, a variable length string is returned.

Example 1

You can use the Sequential Function to change customer data for a test database. Assume that the CUST_ID column is defined as CHAR(5). To increment by 50, starting at 1, specify:

SEQ(1, 50)

In this example, the function returns CUST_ID values starting at '00001' and increments by 50 to generate '00051', '00101', etc. When the result exceeds '99951', the function resets to the *start* value of 1.

Example 2

You can use the Sequential Function in a Column Map to mask sales data for a test database. Assume that the YTD_SALES column is defined as DECIMAL(7,2). To increment by 100 starting at 1000, specify:

SEQ(1000, 100)

In this example, the function returns YTD_SALES values starting at 1000 and increments by 100 to generate 1100, 1200, etc. When the result exceeds 99999, the function resets to the *start* value of 1000.

Example 3

Assume that the SALESMAN_ID column is defined as CHAR(6). To insert values beginning with 'NJ,' followed by a number starting at 50 and incremented by 10, use the function in a concatenated expression:

'NJ'||SEQ(50, 10)

In this example, the function returns SALESMAN_ID values starting at 'NJ50 ' and increments by 10 to generate 'NJ60 ', 'NJ70 ', etc. When the result exceeds 'NJ9990', the function resets to the *start* value.

Identity or Serial Function

The Identity and Serial Functions direct the DBMS to supply a sequential value (integer) for a destination column. The syntax for these functions is:

IDENTITY( )

SERIAL ( )

- Use the Identity Function for Identity columns in DB2, Sybase ASE, and SQL Server databases.
- Use the Serial Function for Serial columns in Informix databases.
- Both functions are valid for Insert (update/insert) and Load Processing, but are not valid for Convert Processing.
- If rows are updated in an Insert Process (update/insert), the destination column targeted by the Identity Function or Serial Function retains the original value. In addition, if the destination column is part of the primary key, the column value remains unchanged when the row is updated.
- You can use the Identity Function or Serial Function with the Propagate Function for Insert or Convert Processing; however, you cannot propagate Identity or Serial columns in a Load Process.

Oracle Sequence Function

The Oracle Sequence Function assigns a value to the destination column using an Oracle Sequence. The syntax is:

***schema.seqname*.NEXTVAL [INCL_UPD]**

schema Qualifier for the Oracle Sequence name.

seqname
Name of the Oracle Sequence that assigns sequential values.

NEXTVAL
Keyword that inserts the next Oracle value into the destination column.

INCL_UPD
Optional keyword that updates a sequence value assigned to a column when rows are updated during an Insert Process. If not specified (default), the column value remains unchanged when the row is updated.

- You can use the Oracle Sequence Function to assign unique sequential values for rows to be inserted into an Oracle database.
- The Oracle Sequence Function is valid when used in a Column Map for Insert or Load Processing, but is not valid for Convert Processing.
- If rows are updated in an Insert (update/insert) Process and the destination column is part of the primary key, the column value remains unchanged when the row in the destination table is updated. To use Oracle Sequence when performing an update/insert, include INCL_UPD with the function.
- During a Load Process, the process uses the Oracle Sequence Function to assign a new value to each destination row. The Load calls the DBMS to obtain these values. If you choose not to run the Oracle Loader, these sequence values are never used.

Example 1

To assign a sequential value to increment customer numbers, where the name of the Oracle Sequence is ***schema.numeven***, specify:

schema.numeven.NEXTVAL

Example 2

To expand the first example and update an existing sequence value, specify:

schema.numeven.NEXTVAL(INCL_UPD)

Propagate Primary or Foreign Key Value Function

The Propagate Function assigns a value to a primary key or foreign key column and propagates that value to all related tables. The syntax is:

PROP({ *value* [, *columnname* |] EXIT *exitname* |
PROC { LOCAL | *identifier.name* } })

value Value to assign to the column. Specify any valid Column Map source value (for example, a column name, string literal, expression, or function). The value must be appropriate for the column.

columnname
Name of the source column containing the value that is the subject of the function. The resulting value is inserted into the destination column of the mapped table and the appropriate destination column in the participating related tables.

The column name is required *only* if no source column matches the destination column in both name and data type. If not specified, the name of the destination column is used.

exitname
See"Exit in a Column Map" on page 473

identifier.name
See"Procedure in a Column Map" on page 175

- If the locale uses a comma as the decimal separator, you must leave a space after each comma that separates numerical parameters.
- The Propagate Function is valid in a Column Map for Insert (but not Update or Update/Insert), Load, or Convert Processing.
- When you use the Propagate Function, at least one related table must be included in the process. You can use Propagate multiple times for the same process.
- You can use the Propagate Function for either a primary key column or its corresponding foreign key column, but not both.
- If multiple columns define a relationship, you can use the Propagate Function for one or more of those columns. However, in an Optim extended relationship, you can specify the Propagate Function only on column-to-column relations.
- You can use the Identity Function or the Serial Function within the Propagate Function for Insert or Convert Processing; however, you cannot propagate the Identity Function in a Load Process.
- The parameters specified in the Propagate Function are not validated until run time. If there are conflicts, the process does not run.
- Insert can have propagate cycles. However, Load and Convert Processing may not result in propagate cycles. Cycles are detected when the process is validated at run time. If a Load or Convert Request generates a propagate cycle, the process does not run.
- Optim remembers the source values and the values assigned to corresponding destination columns. Therefore, you can propagate to destination columns where the source is an expression. When the evaluated expression matches a source value, Optim assigns the corresponding destination value. When the evaluated expression does not match any source values, a conversion error occurs.

Before executing an Insert, Load or Convert Process, you can review the Column Map to verify how the Propagate Function is used in the process.

Example 1

You can generate a random number, assign it to the default destination column, and propagate the number in the destination columns of related tables. To generate a value between 10000 and 99999, insert it into the mapped destination column and propagate it to the destination columns of related tables, specify:

PROP(RAND(10000, 99999))

Example 2

You can perform the same function as in Example 1 when the source and destination column names do not match. To include the name of the source column (CUST_NUMBER) in the Propagate Function, specify:

PROP(RAND(10000, 99999), CUST_NUMBER)

Example 3

You can use Oracle Sequence to generate the value for the destination column and propagate that value in destination columns of the related tables. To propagate the Oracle Sequence named, ***schema.numeven***, specify:

PROP(schema.numeven.NEXTVAL)

Concatenated Expressions

Concatenation allows you to combine column values or combine a column value with another value, using a concatenation operator (CONCAT, ||, or +). A concatenated expression can include character values or binary values, but not both:

Character Values
Concatenated character values can be character columns, string literals, substrings of values in character columns, the sequential function, or the random function.

Binary Values
Concatenated character values can be binary columns, hexadecimal literals, substrings of binary columns, the sequential function, or the random function.

- A concatenated expression cannot include a zero-length string literal (' '), a special register, or the Age Function.

Example

Assume that the CUSTOMERS table stores an address in two columns: ADDRESS1 and ADDRESS2. The SHIP_TO table stores an address in one column: ADDRESS. You can use a concatenated expression to combine address information from two columns in one table to one column in another.

To combine the address, specify one of the following:

CUSTOMERS Table	SHIP_TO Table
ADDRESS1 || ADDRESS2	ADDRESS
ADDRESS1 CONCAT ADDRESS2	ADDRESS
ADDRESS1 + ADDRESS2	ADDRESS

Numeric Expressions

You can use a numeric expression to specify a value in the source column whenever the data types for the corresponding source and destination columns are compatible. An arithmetic expression consists of:

operand1 operator operand2

Each operand must be a numeric column or a numeric constant. The operator specifies whether to add (+), subtract (-), divide (/), or multiply (*).

Example 1

To increase the value in a column UNIT_PRICE defined as DECIMAL(5,2) by 10 percent, specify:

1.1 * UNIT_PRICE

Example 2

To divide the value in an INTEGER column ON_HAND_INVENTORY in half, specify:

ON_HAND_INVENTORY / 2.

Data Privacy Functions

Data Privacy functions provide various methods to transform or mask sensitive data. They require an Optim Data Privacy License.

Lookup Functions

The Column Map lookup functions allow you to select values from a lookup table that are used to populate the destination table. Use the Lookup and Hash Lookup functions to select values based on the source value. Alternatively, the Random Lookup function allows you to select values from a lookup table without regard to the source value.

Sample Lookup Tables

Optim supplies an Extract file, PRIVACY.XF, that you can use to create lookup tables. PRIVACY.XF includes names for Australia, Canada, Germany, Spain, France, Italy, and the US:

- A table of mixed gender first names
- A table of feminine first names
- A table of masculine first names
- A table of last names

PRIVACY.XF also includes:

- A table of company names that are not country-specific.
- A table of names with personal information, including Social Security and Credit Card numbers
- A table of United States addresses
- A table of valid credit card numbers for:
  - American Express Company
  - Diners Club
  - Discover Card
  - JCB (Japan)
  - Visa

Each table has an integer key column that can be used with the Hash Lookup function.

See the *Installation and Configuration Guide* for more information.

Lookup Function

The Lookup Function obtains the value for a destination column from a lookup table, according to the value in a source column. There are two forms of the Lookup Function, single column and multiple column.

The single column form inserts a value into a single destination column. The multiple column form inserts values from multiple lookup table columns into corresponding destination columns.

You can enter the multiple column Lookup Function for any source column that will be replaced by a lookup table value, but you must edit the Column Map to remove the names of remaining source columns that will also be replaced.

The IGNORE parameter allows you to ignore the lookup table and use a source value when a row in a specified source column contains a specified value (NULL, SPACES (for CHAR columns), or zero-length VARCHAR).

You can use the PRESERVE parameter to ignore the lookup table and use a source value when a row in a specified source column contains a specified value (NULL, SPACES (for CHAR columns), or zero-length VARCHAR). PRESERVE can also be used to ignore the lookup table if a source column does not contain a value.

If a match is not found in the lookup table, a conversion error is reported.

The syntax is:

LOOKUP ([*sourcecol,***] [dest=(***col1, coln***) ,]**
***lktablename* (*search*, {value | values=(*col1, coln*) }**
[,cache | ,nocache])
[,ignore=(*colname* **(spaces, null, zero_len),) | PRESERVE=([NOT_FOUND,]** ***colname*** **(spaces, null, zero_len),)])**

sourcecol
Name of the source table column that contains the search value (optional). If not specified, the name of the destination column is used.

dest= Names of the destination table columns in which values from the lookup table are inserted. (Required for multiple column lookup.)

col1, coln
Destination table column names. The order of the column names must correspond to the lookup table columns in the **values=** parameter.

lktablename
Name of the lookup table. You may specify the lookup table name as ***dbalias.creatorid.tablename***, ***creatorid.tablename***, or ***tablename***. If you do not fully qualify the table name, the qualifiers for the destination table are used.

search Name of the column in the lookup table that contains a value to match against the search value from the source column.

value Name of the column in the lookup table that contains the translated search value to be inserted at the destination. (Required for single column lookup.)

values=
Names of the lookup table columns that contain values to be inserted at the destination. (Required for multiple column lookup.)

col1, coln
Lookup table column names. The order of the column names must correspond to the destination table columns in the **dest=** parameter.

cache | nocache
Specify cache (default) to maintain a table of found lookup values in memory or nocache to discard found values. Using cache is faster when retrieving a value many times, but requires extra memory.

ignore=

List of source columns with values that are inserted at the destination instead of the lookup value when the column has a row with a stated value (null, spaces, zero or zero-length varchar).

col The source column name.

For single column lookup, enter one column name only.

For multiple column lookup, the order of the column names must correspond to the destination table columns in the **dest=** parameter. The number of columns must equal the columns in the **dest=** parameter, and at least one column must include values. To not specify values for a column, do not enter a value. For example, coln().

null Ignore the lookup table if the source column row has a null value.

SPACES
Ignore the lookup table if the source column row has a SPACES value. For CHAR columns only.

ZERO_LEN
Ignore the lookup table if the source column row has a zero-length VARCHAR value.

PRESERVE=

List of source columns with values that are inserted at the destination instead of the lookup value when the source column contains a stated value (NOT_FOUND, null, spaces, or zero-length varchar).

NOT_FOUND
Ignore the lookup table if no match is found for the source column row.

Note:

Preserve= and ignore= are mutually exclusive. Ignore= will be deprecated in a future release.

The col, null, spaces, and zero-length varchar operands have the same effect when used with either preserve= or ignore=. Details on these operands are shown above with ignore=.

Single Column Example

Use the Lookup Function to translate the source value in a lookup table to a corresponding value in another table.

For example, assume the source column, STATE, contains state abbreviations (for example, **NJ**) and the destination column is to contain the complete state name (in this example, **New Jersey**). A lookup table named STATE_LOOKUP contains a column (CODE) for state abbreviations or codes and a column (NAME) for the corresponding names.

To obtain the value for the destination column using the STATE_LOOKUP table, specify:

LOOKUP(STATE,STATE_LOOKUP(CODE,NAME))

The Lookup Function searches for a value in the CODE column of the STATE_LOOKUP table that matches the value (**NJ**) in the source table STATE column. When a match is found, the function inserts the corresponding value from the NAME column (**New Jersey**) in the destination column.

Multiple Column Example

Use the Lookup Function to insert values from columns in a lookup table row into columns in a destination table row, based on a value in a source column.

For example, based on a source column (SOC_SEC) that contains social security numbers, you can replace values in destination columns (FIRST_NAME and LAST_NAME) with first and last names from a lookup table. A table named NAME_LOOKUP contains a column (SSN) with the social security numbers from the source table as well as columns (FIRST_MASK and LAST_MASK) to mask corresponding names in the destination.

To replace names in the destination table based on a social security number, specify:

LOOKUP(SOC_SEC,DEST=(FIRST_NAME,LAST_NAME),
NAME_LOOKUP(SSN,VALUES=(FIRST_MASK, LAST_MASK)))

The Lookup Function searches for a value in the SSN column of the NAME_LOOKUP table that matches the value in the source table SOC_SEC column. When a match is found, the function inserts the corresponding values from the lookup table FIRST_MASK and LAST_MASK columns into the corresponding destination columns.

Ignore Example

Use the following statement to extend the Single Column Example, where you want to use the source NULL and SPACES values instead of lookup table values:

LOOKUP(STATE,STATE_LOOKUP(CODE,NAME),
IGNORE=(STATE(NULL,SPACES)))

NoCache Example

Use the following statement to extend the Single Column Example, where you do not want to maintain a table of found lookup values in memory:

LOOKUP(STATE,STATE_LOOKUP(CODE,NAME),NOCACHE)

Hash Lookup Function

The Hash Lookup Function obtains the value for a destination column from a lookup table, according to a hashed value derived from a source column. The Hash Lookup Function allows you to consistently mask data when you use the same source and lookup tables in any environment.

The source column that is hashed does not need to be a column that will be replaced by lookup table values.

The Hash Lookup Function is case-sensitive. For example, the source values John and JOHN will be hashed to different values. You can use the TRIM parameter to convert the source value to uppercase before it is hashed.

There are two forms of the Hash Lookup Function, single column and multiple column. The single column form inserts a value into a single destination column. The multiple column form inserts values from multiple lookup table columns into corresponding destination columns, based on a single hash value from a source column.

You can enter the multiple column Hash Lookup Function for any source column that will be replaced by lookup table values, but you must edit the Column Map to remove the names of remaining source columns that will also be replaced.

The lookup table must include a key column that contains sequential number values without any gaps, and the remaining columns contain replacement values. The key column must be a numeric data type. The lookup table is typically indexed. The function hashes a source column to derive sequential numbers

from 1 to the maximum value in the key column of the lookup table. The hashed value from the source table is matched with the sequential numbers in the lookup table, and values from the corresponding lookup table row are inserted at the destination.

If the source column used to derive the hashed value contains certain values (NULL, spaces (for CHAR columns), zero-length VARCHAR), the value is not hashed and the following reserved values are used as keys to the lookup table:

Source Value	Lookup Table Key
NULL	-1
spaces (CHAR or VARCHAR)	-2
zero-length VARCHAR	-3

The lookup table should include a row for each of these numbers, allowing you to insert a lookup value for each of these source values. If one of these source values is found and a corresponding number is not in the lookup table, a conversion error is reported.

The IGNORE parameter allows you to ignore the lookup table and use a source value when a row in a specified source column contains a specified value (NULL, SPACES (for CHAR columns), or zero-length VARCHAR).

You can use the PRESERVE parameter to ignore the lookup table and use a source value when a row in a specified source column contains a specified value (NULL, SPACES (for CHAR columns), or zero-length VARCHAR). PRESERVE can also be used to ignore the lookup table if a source column does not contain a value.

The TRIM parameter allows you to specify characters that will be trimmed from the source value before it is hashed. For example, if you choose to trim commas from a source value, the values Smith, John, and Smith John will each be hashed to the same value. You can also use this parameter to convert the source value to uppercase before it is hashed.

If the source value is converted to uppercase, the trim characters are also converted to uppercase.

You can use the SEED parameter to vary the calculation performed by the hashing algorithm. The hashed value from the source column and the SEED value are matched with a sequential number from the lookup table to obtain the replacement value for the destination column.

The syntax is:

HASH_LOOKUP([*sourcecol***,] [trim=([***char1char2***] [\u]),]**
dest=(*col1, coln*), *lktablename* (*search*,
{ *value* | values=(*col1, coln*) }) [,cache | ,nocache]
[,ignore=(*col* (spaces, null, zero_len),) | PRESERVE=([NOT_FOUND,] *colname* (spaces, null, zero_len),)])[,seed=*n*])

sourcecol
Name of the source table column from which hashed values are derived (optional). If not specified, the name of the destination column is used.

trim= List of characters to be trimmed from the source value before it is hashed as well as an option to convert the source value to uppercase before it is hashed. If the resulting source value is NULL or all spaces after characters have been trimmed, the source value will not be hashed and will be assigned the appropriate reserved value (-1 or -2).

*char1char2***...**
Characters to be trimmed from the source value before it is hashed. The list is

case-sensitive. You can specify a space or comma as a character. After the initial occurrence of a character, any additional occurrences in the list are ignored.

To specify a backslash "\" or a right parentheses ")", you must precede the character with a backslash escape character. For example, to specify a right parentheses, enter: trim=(\)).

You can only use the escape character with a backslash, a right parentheses, or as part of the uppercase indicator.

\u Indicates the source value is to be converted to uppercase before it is hashed. The characters to be trimmed are also converted to uppercase.

dest= Names of the destination table columns in which values from the lookup table are inserted. (Required for multiple column lookup.)

col1,coln
Destination table column names. The order of the column names must correspond to the lookup table columns in the values= parameter.

lktablename
Name of the lookup table. You may specify the lookup table name as **dbalias.creatorid.tablename**, **creatorid.tablename**, or **tablename**. If you do not fully qualify the table name, the qualifiers for the destination table are used.

search Name of the column in the lookup table that contains sequential values to match against the hash values from the source column.

value Name of the column in the lookup table that contains the translated search value to be inserted at the destination. (Required for single column lookup.)

values=
Names of the columns in the lookup table that contain values to be inserted at the destination. (Required for multiple column lookup.)

col1,coln
Lookup table column names. The order of the column names must correspond to the destination table columns in the dest= parameter.

cache | nocache
Specify CACHE (default) to maintain a table of found lookup values in memory or NOCACHE to discard found values. Using CACHE is faster when retrieving a value many times, but requires extra memory.

ignore=
List of source columns with values that are inserted at the destination instead of the lookup value when the column has a row with a stated value (NULL, SPACES, ZERO, or zero-length VARCHAR).

col The source column name.

For single column lookup, enter one column name only.

For multiple column lookup, the order of the column names must correspond to the destination table columns in the DEST= parameter. The number of columns must equal the columns in the DEST= parameter, and at least one column must include values. To not specify values for a column, do not enter a value. For example, coln().

NULL Ignore the lookup table if the source column row has a NULL value.

SPACES
Ignore the lookup table if the source column row has a SPACES value. For CHAR columns only.

ZERO_LEN
Ignore the lookup table if the source column row has a zero-length VARCHAR value.

PRESERVE=
List of source columns with values that are inserted at the destination instead of the lookup value when the column contains a stated value (NOT_FOUND, null, spaces, or zero-length varchar).

NOT_FOUND
Ignore the lookup table if no match is found for the source column row.

Note:

Preserve= and ignore= are mutually exclusive. Ignore= will be deprecated in a future release.

The col, null, spaces, and zero-length varchar operands have the same effect when used with either preserve= or ignore=. Details on these operands are shown above with ignore=.

seed= Use SEED= to vary the hashing algorithm calculation. Values from 1 to 2,000,000,000 can be used. If you use a value of 0, the SEED parameter is ignored.

Single Column Example

Use the Hash Lookup Function to insert values from a column in a lookup table into a destination table column, based on a value hashed from a source column.

For example, assume the source column, FIRST_NAME, contains first names and the destination column will include replacement first names from the lookup table. A lookup table, NAME_LOOKUP, contains a column (FIRST) with first names and a column (SEQ) containing sequential values.

To obtain values for the destination column using the NAME_LOOKUP table, specify:

HASH_LOOKUP(FIRST_NAME,NAME_LOOKUP(SEQ, FIRST))

The Hash Lookup Function matches the hash values from the source column with values in the SEQ column of the NAME_LOOKUP table. When a match is found, the function inserts the corresponding value from the FIRST column into the destination column.

Multiple Column Example

Use the Hash Lookup Function to insert values from columns in a lookup table row into columns in a destination table row, based on a value hashed from a source column.

For example, based on values hashed from a source column (FIRST_NAME) that contains first names, you can replace values in destination columns (FIRST and LAST) with first and last names from a lookup table. A lookup table named NAME_LOOKUP contains a column (SEQ) with sequential values as well as columns (FIRST_MASK and LAST_MASK) to mask values in the destination.

To replace names in the destination table based on values hashed from a source column, specify:

HASH_LOOKUP(FIRST_NAME,DEST=(FIRST,LAST),
NAME_LOOKUP(SEQ,VALUES=(FIRST_MASK, LAST_MASK)))

The Hash Lookup Function matches the hash values from the source FIRST_NAME column with values in the SEQ column of the NAME_LOOKUP table. When a match is found, the function inserts the corresponding values from the lookup table FIRST_MASK and LAST_MASK columns into the corresponding destination columns.

Ignore Example

Use the following statement to extend the Single Column Example, where you want to use the source NULL and SPACES values instead of lookup table values:

**HASH_LOOKUP(FIRST_NAME,NAME_LOOKUP(SEQ,
FIRST),IGNORE=(FIRST_NAME(NULL,SPACES)))**

NoCache Example

Use the following statement to extend the Single Column Example, where you do not want to maintain a table of found lookup values in memory:

HASH_LOOKUP(FIRST_NAME,NAME_LOOKUP(SEQ, FIRST),NOCACHE)

Trim Example

Use the following statement to extend the Single Column Example, where you want to trim spaces and commas from the source value as well as convert the source value to uppercase before it is hashed:

HASH_LOOKUP(FIRST_NAME, TRIM=(,\u),NAME_LOOKUP(SEQ,FIRST))

Random Lookup Function

The Random Lookup Function selects a value at random from a specified lookup table to insert in a destination column. The function generates a random number between 1 and the limit or number of rows in the lookup table to use as a subscript into the table. The column value or values from the row that correspond to the subscript are inserted in the destination column.

There are two forms of the Random Lookup Function, single column and multiple column. The single column form inserts a value into a single destination column. The multiple column form inserts values from multiple lookup table columns into corresponding destination columns.

You can enter the multiple column Random Lookup Function for any source column that will be replaced by a lookup table value, but you must edit the Column Map to remove the names of remaining source columns that will also be replaced.

The IGNORE parameter allows you to ignore the lookup table and use a source value when a row in a specified source column contains a specified value (NULL, SPACES (for CHAR columns), or zero-length VARCHAR).

You can use the PRESERVE parameter to ignore the lookup table and use a source value when a row in a specified source column contains a specified value (NULL, SPACES (for CHAR columns), or zero-length VARCHAR).

The syntax is:

RAND_LOOKUP(*lktablename***, { *columnname* | dest=(*col1*,*coln*) ,values=(*col1*,*coln*) }
[,*limit*] [,ignore=(*col*(spaces, null, zero_len),) | PRESERVE=(*colname* (spaces, null, zero_len),)])**

lktablename
: Name of the lookup table. You may specify the lookup table name as ***dbalias.creatorid.tablename***, ***creatorid.tablename***, or ***tablename***. If the table name is not fully qualified, destination table qualifiers are used.

columnname
: Name of the column in the lookup table that contains the values to be randomly selected for insertion at the destination. (Required for single column lookup.)

dest= Names of the destination table columns in which values from the lookup table are inserted. (Required for multiple column lookup.)

col1,coln
Destination table column names. The order of the column names must correspond to the lookup table columns in the **values=** parameter.

values=
Names of the columns in the lookup table that contain values to be inserted at the destination. (Required for multiple column lookup.)

col1,coln
Lookup table column names. The order of the column names must correspond to the destination table columns in the **dest=** parameter.

limit Optional limit on number of rows from the lookup table used to select column values. Specify an integer, up to a maximum value of 2,000,000,000. If no limit is specified, all rows are used.

Note: A table of column values is generated in memory. The size of this table may be limited by system resources.

ignore=
List of source columns with values that are inserted at the destination instead of the lookup value when the column has a row with a stated value (NULL, SPACES, or zero-length VARCHAR).

col The source column name.

For single column lookup, enter one column name only.

For multiple column lookup, the order of the column names must correspond to the destination table columns in the **dest=** parameter. The number of columns must equal the columns in the **dest=** parameter, and at least one column must include values. To not specify values for a column, do not enter a value. For example, coln().

null Ignore the lookup table if the source column row has a NULL value.

spaces Ignore the lookup table if the source column row has a SPACES value. For CHAR columns only.

zero_len
Ignore the lookup table if the source column row has a zero-length VARCHAR value.

PRESERVE=
List of source columns with values that are inserted at the destination instead of the lookup value when the column contains a stated value (NOT_FOUND, null, spaces, or zero-length varchar).

NOT_FOUND
Ignore the lookup table if no match is found for the source column row.

Note:

Preserve= and ignore= are mutually exclusive. Ignore= will be deprecated in a future release.

The col, null, spaces, and zero-length varchar operands have the same effect when used with either preserve= or ignore=. Details on these operands are shown above with ignore=.

Single Column Example

To select a value at random from the STATE column in the first 50 rows of a table named STATE_LOOKUP and insert it in the destination column, specify:

RAND_LOOKUP(STATE_LOOKUP,STATE,50)

Multiple Column Example

To select values from the CITY, STATE, and ZIPCODE columns in a random row of a table named STATE_LOOKUP and insert them in the corresponding destination columns, specify:

RAND_LOOKUP(STATE_LOOKUP,
DEST=(CITY,STATE,ZIPCODE),
VALUES=(CITY,STATE,ZIP))

Ignore Example

Use the following statement to extend the Single Column Example, where the source column is named STATES and you want to use the source NULL and SPACES values instead of lookup table values:

RAND_LOOKUP(STATE_LOOKUP,STATE,50, IGNORE=(STATES(NULL,SPACES)))

Shuffle Function

The Shuffle Function replaces a source value with another value from the column that is then inserted in a destination column. The source row and the row that contains the replacement value will never be the same, but depending on your data, source and replacement values can be identical.

You can indicate the number of times the function will refetch a replacement value until a value that does not match the source value is found (a "retry"), or you can allow a replacement value to match the source. Each Shuffle Function operates independently of other Shuffle Functions used in a Column Map.

Use the **Actions** tab in Personal Options to define a default that determines either the number of retries allowed per row or if the replacement value can match the source (no retry). You can use the **retry** parameter in the function to override this default. If the maximum number of retries is reached before a value that does not match the source is found, a conversion error will occur.

There are two forms of the Shuffle Function, single column and multiple column. The single column form inserts a replacement value into a single destination column. The multiple column form inserts replacement values from multiple columns in a row into corresponding destination columns. A column cannot be included in more than one Shuffle Function in a Column Map. If the retry feature is used with a multiple column shuffle, the function will refetch another replacement row if any value in the source row column matches the value in a corresponding replacement row column. (The multiple column form cannot be used in a Propagate Function.)

To create a multiple column Shuffle Function, enter the function for a source column that will be replaced with shuffled values, and edit the Column Map to remove the names of any other source columns with values that will also be replaced.

The **ignore** parameter prevents the function from replacing a source row or using a replacement row if either contains a specified value (NULL, SPACES (for CHAR columns), or zero-length VARCHAR). If no retries are allowed, the **ignore** parameter will not apply to the replacement row.

The syntax is:

SHUFFLE [(dest=(col1,coln))] |

[(dest=(col1,coln) , retry[=number])] |

[(dest=(col1,coln) [, retry[=number]] , ignore=(col1 ([spaces] | [spaces,null] | [spaces,null,zero_len] | [null] | [null,zero_len] | [zero_len]) , coln (....)))] |

[(retry[=number])] |

[(retry[=number] , ignore=(col ([spaces] | [spaces,null] | [spaces,null,zero_len] | [null] | [null,zero_len] | [zero_len])))] |

[(ignore=(col ([spaces] | [spaces,null] | [spaces,null,zero_len] | [null] | [null,zero_len] | [zero_len])))]

where:

dest= Names of the destination table columns in which replacement values are inserted. (Required for multiple column shuffle.)

col1, *coln*, **...**
Destination table column names.

retry Number of times to refetch a replacement value to find a value that does not match the source row. Enter zero to allow a replacement value to match the source. This parameter overrides the default set on the **Actions** tab in Personal Options.

Note: Using a high retry value with columns that contain many duplicate values will increase the processing time. For these columns, it may be best to use a retry value of zero.

=number
Enter a value in the range 0-1000. Enter 0 to allow a replacement value to match the source. (If you enter "retry" alone, Optim uses the default set in Personal Options.)

ignore=
List of columns for which the function will not replace a source value or not use a replacement value if either is a specified value (NULL, SPACES (for CHAR columns), or zero-length VARCHAR). If a replacement value is ignored, the function will refetch another replacement value. If no retries are allowed, the **ignore** parameter will not apply to replacement values.

col The source column name.

For single column shuffle, enter one column name only.

For multiple column shuffle, the order of the column names must correspond to the destination table columns in the **dest=** parameter. The number of columns must equal the columns in the **dest=** parameter, and at least one column must include values. To not specify values for a column, do not enter a value. For example, coln().

null Do not replace the source value or use a replacement value if either is a NULL value.

spaces Do not replace the source value or use a replacement value if either is a SPACES value. For CHAR columns only.

zero_len
Do not replace the source value or use a replacement value if either is a zero-length VARCHAR value.

Single Column Default Example

The following example inserts shuffled values in a single column using the default retry setting in Personal Options.

SHUFFLE

Single Column Retry Example

The following example inserts shuffled values in a single column and refetches a replacement value that does not match the source up to 12 times.

SHUFFLE(RETRY=12)

Multiple Column Example

The following example inserts shuffled values in the STATE and ZIP columns and refetches a replacement value that does not match the source up to 12 times.

SHUFFLE(DEST=(STATE,ZIP),RETRY=12)

Ignore Example

The following example inserts shuffled values in the STATE and ZIP columns and refetches a replacement value that does not match the source up to 12 times. The example also does not replace a source value or use a replacement value for the STATE column if a source or replacement row contains a NULL or SPACES value, but does not ignore any source or replacement rows for the ZIP column.

SHUFFLE(DEST=(STATE,ZIP),RETRY=12,
IGNORE=(STATE(NULL,SPACES),ZIP()))

Data Privacy Transformation Library Functions

The data privacy transformation library functions allow you to mask personal data such as social security numbers, credit card numbers, and email addresses. You can generate transformed data that is valid and unique.

When a transformation library function is called, all destination and source columns are available except those destination columns that are yet to be assigned a value by an exit or transformation library function. The parameters in a transformation library function are validated at run time.

TRANS SSN

Use the TRANS SSN function to generate a valid and unique U.S. Social Security Number (SSN). By default, TRANS SSN algorithmically generates a consistently altered destination SSN based on the source SSN. TRANS SSN can also generate a random SSN when the source data does not have an SSN value or when there is no need for transforming the source SSN in a consistent manner.

An SSN is made of 3 subfields. The first 3 digits (area) represent an area generally determined by the state in which the SSN is issued. The next 2 digits (group) define a group number corresponding to the area number. The last 4 digits (serial) are a sequential serial number. Regardless of the type of processing, default or random, TRANS SSN will generate an SSN with a group number appropriate to the area number.

The default processing method generates an SSN that includes the source area number as well as altered group and serial numbers based on the source SSN.

The random processing method generates an SSN that can include the source area number and uses a group number most recently issued by the Social Security Administration for the destination area number. Serial numbers begin with 0001 and are incremented by 1 for each additional SSN generated for the area number. When the serial number exceeds 9999, the serial number will be reset to 0001 and the group number preceding the number most recently issued for the area number will be used.

The syntax of TRANS SSN is:

TRANS SSN [('[=*flags*] [*sourcecol* **[preserve=invalid]')]**

flags You can specify one or more case-insensitive processing option flags.

n Generate a random SSN that is not based on a source value.

r Generate a random SSN that includes the source area number.

v Validate the source group number by comparing it with numbers used by the Social Security Administration.

- The destination SSN should include dashes separating the fields (for example, 123-45-6789). Requires a character-type destination column at least 11 characters long.

sourcecol
The source column name. If a source column name is not specified, the destination column name will be used. If a source column name is not specified and the destination column name does not match a column name in the source table, an error will occur during processing.

preserve=invalid
If the source column contains an invalid SSN, do not replace it with a generated value. The source column value will be used in the destination column.

Data Types Allowed

The following source and destination data types are permitted:

CHAR
The length of data in the column must be from 9 to 256 characters.

DECIMAL
The precision of the column must be 9 - 20 and the scale 0.

INTEGER
No restrictions.

VARCHAR
The length of data in the column must be from 9 to 254 characters.

If a source or destination column does not adhere to these restrictions, an error will occur during processing.

Destination Processing Rules

The following rules apply to the destination SSN value, according to the destination data type or value:

CHAR
If the source value is 0, spaces, or a zero-length VARCHAR, the destination value will be set to spaces.

If a source value is 11 characters or more and includes embedded dashes (-), or if the '-' flag is specified, the destination value will include dashes if the destination column length is 11 characters or more.

DECIMAL,
INTEGER
If the source value is 0, spaces, or a zero-length VARCHAR, the destination value will be 0.

VARCHAR
If the source value is 0, spaces, or a zero-length VARCHAR, the destination length will be 0.

If a source value is 11 characters or more and includes embedded dashes (-), or if the '-' flag is specified, the destination value will include dashes if the destination column length is 11 characters or more.

NULL If the source value is NULL, the destination value will be NULL.

Skipped Rows

The following conditions may cause a source row to be skipped and not written to the destination:

- The source value is NULL, and the destination column does not allow a NULL value.
- The source column is CHAR or VARCHAR, and the source value is less than 9 characters, contains a non-numeric character (other than dashes between the 3 subfields), or is too large.
- The source area number has not been used by the Social Security Administration.
- The source group number has not been used with the area number by the Social Security Administration (only if the 'v' flag has been specified).
- The source serial number is 0000, or the SSN is a reserved value not issued by the Social Security Administration (for example, 078-05-1120).
- The source value cannot be converted to a format TRANS SSN supports.

Error Messages

The following error messages may be issued:

SSN01

Parm on Col ccccc ("ppp") is invalid

Explanation

The column contains a TRANS function with a processing option flag that is not valid.

User Action

Ensure that the TRANS function on the column specified uses a valid processing option flag (n, r, v, -).

SSN02

Col ccccc not on source

Explanation

The column that was entered as a sourcecol parameter or the destination column name (if the sourcecol parameter was omitted) was not found on the source table.

User Action

Check the source table and resolve any discrepancies or missing columns.

SSN03

Source Col ccccc-aaa invalid

Explanation

The format of the source column is not supported because the attribute indicated is not valid.

User action

Check the source column and ensure the values for type, length, precision, and scale are appropriate.

SSN04

Dest Col ccccc-aaa invalid

Explanation

The format of the destination column is not supported because the indicated attribute is not valid.

User Action

Check the destination column to ensure the values for type, length, precision, and scale are appropriate.

SSN05

Get col ccccc data-rc=nnn

Explanation

An unexpected internal error has occurred while getting the value from the source column.

User Action

Check the values of the source and destination columns and ensure the values for type, length, precision, and scale are appropriate. If the problem persists, contact IBM Software Support.

SSN08

Put col ccccc data-rc=nnn

Explanation

An unexpected internal error has occurred while setting the value on the destination column.

User Action

Check the values of the source and destination columns and ensure the values for type, length, precision, and scale are appropriate. If the problem persists, contact IBM Software Support.

If any other errors occur, contact Technical Support.

Example 1

The following example uses a source column name that matches the destination column and generates a random SSN that is not based on the source value:

TRANS SSN ('=n')

Example 2

The following example uses a source column name (NATIONAL_ID) that differs from the destination column and generates an SSN using the default processing method and including dashes:

TRANS SSN ('=- NATIONAL_ID')

TRANS CCN

Use the TRANS CCN function to generate a valid and unique credit card number (CCN). By default, TRANS CCN algorithmically generates a consistently altered CCN based on the source CCN. TRANS CCN can also generate a random value when the source data does not have a CCN value or when there is no need for transforming the source CCN in a consistent manner.

A CCN, as defined by ISO 7812, consists of a 6-digit issuer identifier followed by a variable length account number and a single check digit as the final number. The check digit verifies the accuracy of the CCN and is generated by passing the issuer identifier and account numbers through the Luhn algorithm. The maximum length of a CCN is 19 digits.

The default processing method generates a CCN by including the first 4 digits of the issuer identifier from the source CCN and altering the remaining 2 digits of the issuer identifier number and the account number based on the source CCN. A valid check digit is also assigned.

The random processing method generates a CCN that can include the first 4 digits of the source issuer identifier number or an issuer identifier number assigned to American Express, Discover, MasterCard, or VISA. A valid check digit is also assigned. If the first four digits of a source issuer identifier number are included, the first account number based on those digits will begin with 1, and for each additional CCN that uses those digits, the account number will be incremented by 1.

The syntax of TRANS CCN is:

TRANS CCN [('[=*flag*] [*sourcecol*] [preserve=invalid]')]

flag Specify an option flag to generate a random CCN.

n Generate a random CCN that is not based on a source value and includes an issuer identifier number assigned to American Express, Discover, MasterCard, or VISA.

r Generate a random CCN that includes the first 4 digits of the source issuer identifier number.

6 Generate a random CCN that includes the first 6 digits of the source issuer identifier number.

sourcecol

The source column name. If a source column name is not specified, the destination column name is used.

If a source column name is not specified and the destination column name does not match a column name in the source table, an error will occur during processing.

preserve=invalid

If the source column contains an invalid CCN, do not replace it with a generated value. The source column value will be used in the destination column.

Data Types Allowed

The following source and destination data types are permitted:

CHAR

The column length must be from 13 to 256 characters.

VARCHAR

The column length must be from 13 to 254 characters.

DECIMAL

The precision of the column must be from 13 to 254 and the scale 0.

If a source or destination column does not adhere to these restrictions, an error message occurs.

Destination Processing Rules

The following rules apply to the destination CCN value, according to the destination data type or value:

CHAR

If the source value is spaces or a zero-length VARCHAR, the destination value will be set to spaces.

VARCHAR

If the source value is spaces or a zero-length VARCHAR, the destination length will be 0.

DECIMAL

If the source value is 0, the destination value will be 0.

NULL If the source value is NULL, the destination value will be NULL.

Skipped Rows

The following conditions may cause a source row to be skipped and not written to the destination:

- The source value is NULL, and the destination column does not allow a NULL value.

- The source value is less than 13 characters, contains a non-numeric character, is too large, or has an incorrect check digit.
- The source value length is not valid for the credit card issuer.
- The source value cannot be converted to a format TRANS CCN supports.

Error Messages

The following error messages may be issued:

CCN01

Parm on Col ccccc ("ppp") is invalid

Explanation

The indicated column contains a TRANS function with a processing option flag that is not valid.

User Action

Ensure that the TRANS function on the column specified uses a valid processing option flag (n, r, 6).

CCN02

Col ccccc not on source

Explanation

The column that was entered as a sourcecol parameter or the destination column name (if the sourcecol parameter was omitted) was not found on the source table.

User Action

Check the source table and resolve any discrepancies or missing columns.

CCN03

Source Col ccccc-aaa invalid

Explanation

The format of the source column is not supported because the attribute indicated is not valid.

User Action

Check the source column and ensure the values for type, length, precision, and scale are appropriate.

CCN04

Dest Col ccccc-aaa invalid

Explanation

The format of the destination column is not supported because the indicated attribute is not valid.

User Action

Check the destination column to ensure the values for type, length, precision, and scale are appropriate.

CCN05

Get col ccccc data-rc=nnn

Explanation

An unexpected internal error has occurred while getting the value from the source column.

User Action

Check the values of the source and destination columns and ensure the values for type, length, precision, and scale are appropriate. If the problem persists, contact IBM Software Support.

CCN08

Put col ccccc data-rc=nnn

Explanation

An unexpected internal error has occurred while setting the value on the destination column.

User Action

Check the values of the source and destination columns and ensure the values for type, length, precision, and scale are appropriate. If the problem persists, contact IBM Software Support.

If any other errors occur, contact Technical Support.

Example 1

The following example uses a source column name (CREDITCARD) that differs from the destination column and generates a random CCN not based on the source value:

TRANS CCN ('=n CREDITCARD')

Example 2

The following example uses a source column name (CREDITCARD) that differs from the destination column and generates a CCN using the default processing method:

TRANS CCN ('CREDITCARD')

TRANS EML

Use the TRANS EML function to generate an email address. An email address consists of two parts, a user name followed by a domain name, separated by '@'. For example, user@domain.com.

TRANS EML generates an email address with a user name based on either destination data or a literal concatenated with a sequential number. The domain name can be based on an email address in the source data, a literal, or randomly selected from a list of large email service providers. The email address can also be converted to upper or lower case.

TRANS EML can generate a user name based on the values in one or two destination table columns (usually containing the name of a user). Processing options allow you to use only the first character of the value in the first column (for example, the initial letter of a first name) and separate the values from both columns using either a period or an underscore.

If the user name is based on a single destination column value or a literal, the name will be concatenated with a sequential number. If a user name is based on values in two destination table columns and a separating period or underscore is not used, the values are concatenated. If a parameter is not provided for the user name, the name will be formed by the literal "email" concatenated with a sequential number. Sequential numbers for user names are suffixes that begin with 1 and are incremented by 1.

The syntax of TRANS EML is:

TRANS EML [('[=*flags*] , [{*sourcecol* | "*domain*" | , }
[{*name1col*[*name2col*] | "*userpfx*"}]] [preserve=invalid]')]

flags You can specify one or more case-insensitive processing option flags.

n Generate a random domain name from a list of large email service providers.

. Separate the *name1col* and *name2col* values with a period.

_ Separate the *name1col* and *name2col* values with an underscore.

i Use only the first character of the *name1col* value.

l Convert the email address to lower case.

u Convert the email address to uppercase.

sourcecol
The source column name with email addresses used to provide the domain name.

If neither the 'n' flag nor the *domain* parameter are defined, the domain name in the source column is used. (If *sourcecol* is not defined, the source column name is based on the destination column name.)

If a source column name is not specified and the destination column name does not match a column name in the source table, an error will occur during processing.

domain A literal, up to 31 characters, that forms the domain name.

, A comma is required if neither a *sourcecol* nor a *domain* parameter is defined and you define either a literal or column name(s) for the domain name.

name1col
A destination table column name with values used to form the first (or only) part of the user name.

name2col
A destination table column name with values used to form the second part of the user name.

userpfx A literal, up to 31 characters, that is concatenated with a sequential number to form the user name.

preserve=invalid
If the source column contains an invalid email address, do not replace it with a generated value. The source column value will be used in the destination column.

Data Types Allowed

The following source and destination data types are permitted:

CHAR
The column length must be from 3 to 256 characters.

VARCHAR
The column length must be from 3 to 254 characters.

If a source or destination column does not adhere to these restrictions, an error message will be issued.

Destination Processing Rules

The following rules apply to the destination email value, according to the destination data type or value:

CHAR
If the source value is spaces or a zero-length VARCHAR, the destination value will be set to spaces.

VARCHAR
If the source value is spaces or a zero-length VARCHAR, the destination length will be 0.

NULL If the source value is NULL, the destination value will be NULL.

Skipped Rows

The following conditions may cause a source row to be skipped and not written to the destination:

- The source value is NULL, and the destination column does not allow a NULL value.
- The source value is a VARCHAR less than 3 characters long.
- The source email value does not contain a '@'.
- The source value cannot be converted to a format TRANS EML supports.

Error Messages

The following error messages may be issued:

EML01

Parm on Col ccccc ("ppp") is invalid

Explanation

The indicated column contains a TRANS function with a processing option flag that is not valid.

User Action

Ensure that the TRANS function on the column specified uses a valid processing option flag (n, . , _ , i, l, u).

EML02

Col ccccc not on source

Explanation

The column that was entered as a sourcecol parameter or the destination column name (if the sourcecol parameter was omitted) was not found on the source table.

User Action

Check the source table and resolve any discrepancies or missing columns.

EML03

Source Col ccccc-aaa invalid

Explanation

The format of the source column is not supported because the attribute indicated is not valid.

User Action

Check the source column and ensure the values for type, length, precision, and scale are appropriate.

EML04

Dest Col ccccc-aaa invalid

Explanation

The format of the destination column is not supported because the indicated attribute is not valid.

User Action

Check the destination column to ensure the values for type, length, precision, and scale are appropriate.

EML05

Get col ccccc data-rc=nnn

Explanation

An unexpected internal error has occurred while getting the value from the source column.

User Action

Check the values of the source and destination columns and ensure the values for type, length, precision, and scale are appropriate. If the problem persists, contact IBM Software Support.

EML08

Put col ccccc data-rc=nnn

Explanation

An unexpected internal error has occurred while setting the value on the destination column.

User Action

Check the values of the source and destination columns and ensure the values for type, length, precision, and scale are appropriate. If the problem persists, contact IBM Software Support.

EML09

Domain literal sssss too long

Explanation

The string specified as the domain name literal (domain) exceeds the maximum limit of 31 characters.

User Action

Specify a domain name consisting of 31 characters or fewer.

EML10

User literal sssss too long

Explanation

The string specified as the user name literal (userpfx) exceeds the maximum limit of 31 characters.

User Action

Specify a user name consisting of 31 characters or fewer.

EML11

Name1 Col ccccc not on dest

Explanation

To perform the indicated TRANS function, a name1 column must be specified on the destination table.

User Action

Verify that the specified name1 column matches the name1 column indicated in the TRANS function.

The *name1col* column name was not found on the destination table.

EML12

Name1 Col ccccc-aaa invalid

Explanation

To perform the indicated TRANS function, the name1 column specified must be in a valid format.

User Action

Verify that the name1 column availability, type, and length are appropriate.

EML13

Name2 Col ccccc not on dest

Explanation
To perform the indicated TRANS function, a name2 column must be specified on the destination table.

User Action
Verify that the specified name2 column matches the name2 column indicated in the TRANS function.

The *name1col* column name was not found on the destination table.

EML14
Name2 Col ccccc-aaa invalid

Explanation
To perform the indicated TRANS function, the name2 column specified must be in a valid format.

User Action
Verify that the name2 column availability, type, and length are appropriate.

If any other errors occur, contact Technical Support.

Example 1

The following example uses a literal (optim.com) to form the domain name and two destination table columns (NAME_FIRST and NAME_LAST) to form a user name that includes an underscore:

TRANS EML ('=_ "optim.com" NAME_FIRST NAME_LAST')

Example 2

The following example uses a domain name from the source column and a literal (OptimUser) to form a user name that will be suffixed with a sequential number:

TRANS EML (', "OptimUser" ')

TRANS COL

The TRANS COL function can mask data that has no inherent format or a format that is not widely known. TRANS COL maintains the format and character type of the source data at the destination.

Data types CHAR, VARCHAR, and non-float numeric can be masked with TRANS COL. Alphabetic and numeric characters are masked, but any other characters in the source data are copied to the destination without being changed. You can generate unique values for unique source data. For non-unique source data, you can generate a different value for each occurrence of the same source and you can generate values with a length different from the source. The source data format and character type determines the format and character type of the destination data. If the source data is uppercase alphabetic characters, the output generated at the destination is uppercase alphabetic.

The syntax of TRANS COL is:

TRANS COL ('{ unique | hash }[source=*colname* **]**
[copy=((*start,len* **[, *"lit"*])...)] [seed= *"lit"* | var (***variable* **)]**
[length=*n* **| max] [preserve=([null] [spaces] [zero_len])]**
[ignore=(spaces)] [num]')

unique
Generate a unique destination value. Use this keyword to ensure that the same source value always produces the same destination value.

hash Generate a destination value by hashing the source value. When hash is used, the same source value can produce different destination values each time the process is run.

source=*colname*
Use this keyword to generate a name for the destination column that is different from the source column name. The value you specify will be converted to uppercase; to prevent the value from conversion to uppercase, enclose the value in double quotation marks.

copy= One or more pairs of substrings to be copied to the destination without being masked. If you supply a literal string, the source characters in the specified positions are replaced. This is valid only for a non-numeric column.

seed= Literal string or reference to an environment variable used to alter the behavior of the masking algorithms. If you specify a literal string, enclose the string in double quotation marks. For an environment variable, the variable name and its value cannot include double quotation marks.

length=*n*
Generate a destination value with a length different from the source value length. Use length=max to generate a destination value that will fill the column completely. The value you select for *n* cannot exceed the defined length of the destination column.

preserve=
List one or more source values that should not be replaced at the destination. Allowable values are null, spaces, or zero_len.

null If the source column has a null value do not replace the value at the destination.

spaces If the source column has a value of spaces do not replace the value at the destination. For CHAR columns only.

zero_len
If the source column has a zero-length VARCHAR value do not replace the value at the destination.

ignore=(spaces)
Omit any spaces in the source column before generating the output.

num For a character data type column, generate the same output value as for a numeric data type column. If you use this operand, do not specify copy= or length=.

Examples:

Source Data Value	TRANS COL Function	Destination Data Value
CDE-7834	TRANS COL UNIQUE	ZWQ-4598
CDE-7834-2008	TRANS COL UNIQUE	SWX-3162-8451
Smith, John	TRANS COL UNIQUE	Fnxwq, Lrzp
SMITH JOHN	TRANS COL UNIQUE	FNXWQ LRZP
CDE-7834-2008	TRANS COL UNIQUE (COPY=((1,3)(10,4))	CDE-4032-2008
CDE-7834	TRANS COL HASH (LENGTH=13)	ZWQ-4598RN7A0
Smith, John	TRANS COL HASH (LENGTH=13)	Fnxwq, LrzpKX
SMITH JOHN	TRANS COL HASH (LENGTH=13)	FNXWQ LRZPKXH
Smith, John	TRANS COL HASH (LENGTH=4)	Fnxw

Age Function

Use the Age Function to age values in a source column. The source column can contain character, numeric, date, or timestamp data. A CHAR or VARCHAR column has a maximum length of 256 bytes.

The Age Function is formatted as:

AGE(*parameters*)

- Define the Age Function to include one or more aging parameters.
- Use commas or spaces to separate parameters in the Age Function.
- Parameters can be specified in any order.

The following is a list of the parameters with valid format and values:

Parameter	Format	Valid Values
Column Name – Specify the name of the source column if it differs from the destination column.	SC=*column-name* SRCCOL=*column-name*	Column Name
Default – Age dates based on the date adjustment value specified in a process request.	DEF	Uses date adjustment value specified in the process request.
None – Do not age value.	NONE	Value should not be aged regardless of specifications in the process request.
Incremental – Incremental Aging is based on a known time unit. Optim supports date aging in single units (for example 20 years) or multiple units (for example, 2 years, 3 months, 2 days).	[+ or -] *n*Y	*n*Y -2500 to +1581 *n*M -30000 to +30000 *n*W -30000 to +30000 *n*D -99999 to +99999
	[+ or -] *n*M	
	[+ or -] *n*W	
	[+ or -] *n*D	
	(The plus [+] sign is optional.)	
Specific Year – Age dates based on a specific four⌂digit year in the desired format.	*nnnn*Y	1582 - 3999
Multiple/Rule – Age dates based on the number of times to apply a business rule. If you define the Age Function using the Multiple/Rule, you must also include the RULE parameter.	*nnnnn*R	1 - 30000

Semantic Aging

Semantic Aging is based on a set of rules that you define to manage dates that occur on holidays, weekends, and so on. You can use Semantic Aging to adjust dates so that they occur on valid business days.

Calendar –

Name of the calendar that defines the special dates to which the rules apply. If you use CALENDAR, you must also specify a RULE.
CA=*calendar-name*
CALENDAR=*calendar-name*

Rule –

Name of the rule that defines the adjustment for special dates. If DEF is specified, the default rule specified in the process request is used.
RU=*rule-name*

RULE=*rule-name*
RU=DEF
RULE=DEF

Century Pivot –
Determines the century for two-digit years. Enter a value 00 to 99.
PI=*nn*
PIVOT=*nn*

- You define calendars and rules by selecting **Calendar** from the **Options** menu in the main window. See "Open the Calendar Editor" on page 259 for details.
- If you specify **AGE(RU=DEF)**, the RULE specified in a process request is used. You must specify values for any other age function parameters.
- If you use RULE and do not specify a CALENDAR, then the Age Function uses the default calendar you specify in a process request.
- If you do not include CALENDAR, RULE, and PIVOT where needed in the Age Function, the default values you specify in a process request apply.
- To specify the correct century for a two-digit year, you must include the PIVOT in the Age Function.
- If you specify a PIVOT value, all two-digit years equal to or greater than the PIVOT value are placed in the 20th century (19xx). All two⌂digit years less than the PIVOT value are placed in the 21st century (20xx). The default PIVOT is **65**.

Date Formats

The source date format and the destination date format must contain a single valid date format and must be less than or equal to the length of the destination column. The format string must be delimited by single quotation marks.

Source Date Format –
Applies the source column format string to age character and numeric columns.
SF=*'format-string'*
SRCFMT=*'format-string'*

If the source column is character or numeric, you must use SRCFMT or a Source Exit Routine (SRCEXIT) to describe the contents of the column. These parameters are mutually exclusive. See Appendix B, "Exit Routines for Column Maps," on page 473 for details.

Destination Date Format –
Applies the destination column format string to age character and numeric columns.
DF='format-string'
DSTFMT=*'format-string'*

If the destination column is character or numeric, you can specify DSTFMT or a Destination Exit Routine (DSTEXIT). If you do not specify a format for the destination, the date aging function uses SRCFMT by default. The destination column for an AGE function cannot be binary.

Use the following character strings to specify components of the date format:

Year	Month	Day	Time	Parts/Second
YYYY	MONTH	DDD	HH	FFFFFF
CCYY	MMM	DD	MI	FFFFF
YY	MM	D	SS	FFFF
	M			FFF
				FF
				F

- If you specify a question mark (**?**) in a format string, the Age Function maps the character value as it is. (Use the question mark to include slashes, dashes, periods, and so on, in the date format.)
- If you specify an asterisk (*) in a format string, the Age Function maps any remaining characters in the source column to the destination column. (Use the asterisk when the column value is a date concatenated to additional characters.)

Note: You can use the Calendar Utility to define a default separator and a default output year. These defaults apply when the source and destination formats require separators or a specific year.

Example 1

To age a date column by 2 years, 6 months, 40 weeks, and 15 days, and then apply a rule, format the Age Function as:

AGE(+2Y,+6M,+40W,+15D,RU=NEXTPAYDAY)

Example 2

To age only the year portion in a date column to the year 2020, and apply a rule, format the Age Function as:

AGE(2020Y,RU=NEXTWORKDAY)

Example 3

To age a date column using MULTIPLE/RULE to increment by five occurrences of a rule called NEXTSTRTQTR, using a calendar called PSAPRULE, format the Age Function as:

AGE(CA=PSAPRULE,RU=NEXTSTRTQTR,5R)

Example 4

To age data in a character or numeric column by the following parameters:

- A named source column.
- The source format, using the first two characters for the last two digits of the year and the remaining 3 digits as the day in the Julian calendar.
- A century pivot to determine the correct century because the source is formatted with a two-digit year. The century pivot in this example is 42. All two-digit years greater than or equal to 42 are placed in the 20th century (19xx). All two-digit years less than 42 are placed in the 21st century (20xx).
- Age date by 5 years.

Format the Age Function as:

AGE(5Y,SC=ORDER_DATE,SF='YYDDD',PI=42)

Currency Function

Use the Currency Function to convert a currency value in a source column from one currency to another. The source column must be defined as numeric, but not floating point. Two conversion methods are available:

Direct Conversion
Provide conversion parameters based on values defined in a Currency Definition. Use the Currency Function to convert a monetary value in a column (replacing the prior value) or, by

defining different source and destination columns, retain the original value and the converted value. You can explicitly define the source and destination currency types or you can identify a reference column to indicate the currency type.

The first calculation preference is to use the conversion rate for the source currency to the destination currency. The second calculation conversion preference is to use the conversion rate for the destination currency to the source.

Triangulation
Convert the value in a column from the source currency to the euro dollar and then convert the euro dollar to the destination currency. Both rates must be provided in the rate table: one between the euro dollar and the source currency, and one between the euro dollar and the destination currency. The specification expression is TRIANG or TR.

The Currency Function is formatted as follows:

CURRENCY({ST=code | SS=(column-name,Types Table number)}
{DT=code | DS=(column-name,Types Table number)}
[SC=column-name] [TR] [CU=Currency Definition name]
[TD=transaction-date-column-name] [DF='format']
[NS=scale])

The Currency Function must include at least a combination of the source currency type (ST) or source specification (SS) and the destination currency type (DT) or destination specification (DS). All other parameters are optional.

The source and destination currency types can be specified in one of two ways:

1. Use the ST/DT keywords to allow explicit specification of the currency using the three-character ISO 4217 Currency Code. The code is searched for on the **Rates** tab of the Currency Definition (see "Rates Tab" on page 278).
2. Use the SS/DS keywords to allow indirect specification of the currency type where a value in a named column in the row is used as a key. The key is correlated with a currency type as defined in the specified Currency Definition Type Table.

If you specify a transaction date (TD) and the transaction date column does not use the DATE format, you must also specify a date format (DF). If a specified transaction date is outside the date ranges specified in the Currency Definition Rates Table, the nearest date range is used for conversion calculations.

Note: If any required data (for example, currency types, rates) are missing, conversion errors will result at run time.

The following table describes the valid format and allowed values for the Currency Function parameters. Parameters can be specified in any order. Use commas or spaces to separate parameters in the Currency Function.

Parameter	Format
Source Column	SC=*column-name* *SRCCOL=column name*
Source Currency Type	ST=*code* *SRCTYP=code* *where code = ISO 4217 Currency Code*
Destination Currency Type	DT=*code* DSTTYP=*code* *where code = ISO 4217 Currency Code*

Parameter	Format
Source Specification	SS=*column name, Types Table number* *SRCSPC=column name, Types Table number* *where column name, Types Table number = Types Table column and number (defined in the Currency Definition) to be used to specify the source currency type*
Destination Specification	DS=*column name, Types Table number* *DSTSPC=column name, Types Table number* *where column name, Types Table number = Types Table column and number (defined in the Currency Definition) to be used to specify the destination currency type*
Triangulation (Forces conversion via the Euro dollar)	TR TRIANG
Currency Definition	CU=*Currency Definition name* *CURTBL=Currency Definition name* *where Currency Definition name = Currency Definition that contains the appropriate conversion parameters. (Overrides the Currency Definition specified in the Insert request. See "Using the Currency Editor" on page 276.)*
Transaction Date	TD=*column name* TRNDAT=*column name* *where column name = Transaction Date column name to identify the conversion date*
Date Format	DF=*'format'* *DATFMT='format' where format = format of transaction date column, if not Date type.*
Numeric Scale	NS=*scale* *NUMSCL=scale* *where scale = scale to be applied to Oracle numeric destination columns with an undefined scale.*

Example 1

To convert from Finnish Markkas to Euro Dollars, format the Currency Function as:

CURRENCY(ST=FIM DT=EUR)

If the original value must be preserved, use the Currency Function to provide a value for a different column in the destination table.

Example 2

To convert from Finnish Markkas to Euro Dollars, and create a new column to retain the original source value (in Finnish Markkas) in a column labeled ITEM_COST, format the Currency Function as:

CURRENCY(ST=FIM DT=EUR SC=ITEM_COST)

Example 3

The examples above assume a simple conversion from a known and fixed source currency to a target currency. Consider a more complex scenario where a column in a table has a numeric monetary value, and a separate column in the table has a key to indicate the type of currency. The monetary value in one row could be Euro Dollars and in another row it could be Finnish Markkas.

To support this scenario, the **Types** tab of the Currency Definition is used to create a Types Table. (See "Types Tab" on page 280.) The Types Table defines the key and the corresponding currency type for the key. The Currency Function must include a source specification that identifies the column that contains the key, and the number of the Types Table that defines the key, as shown:

CURRENCY(SC=ITEM_COST SS=(ITEM_BASIS,1) DT=EUR)

SC=ITEM_COST
The source value is in the column ITEM_COST.

SS=(ITEM_BASIS,1)
The key for the type of currency is in the column ITEM_BASIS, and Types Table 1 in the Currency Definition is used to correlate the key to the currency type.

DT=EUR
The result of the calculation displays as euro dollars.

Compatibility Rules for Column Maps

The following classes of data and associated data types are supported. These data classes are important for data compatibility when you specify column values in relationships and Column Maps.

Class	DBMS	Data Types
Character	DB2	CHAR, VARCHAR, CLOB
	Oracle	CHAR, VARCHAR2, LONG, CLOB, NCLOB, NCHAR, NVARCHAR
	Sybase ASE	CHAR, VARCHAR, TXT
	SQL Server	CHAR, VARCHAR, TXT
	Informix	CHAR, VARCHAR, TXT
	Note: Single-byte character columns are not compatible with multi-byte or Unicode character columns.	
Numeric	DB2	INTEGER, SMALLINT, DECIMAL, FLOAT, DOUBLE
	Oracle	NUMBER, FLOAT
	Sybase ASE	TINYINT, INT, SMALLINT, DECIMAL, FLOAT, REAL, MONEY, SMALL MONEY
	SQL Server	TINYINT, INT, SMALLINT, DECIMAL, FLOAT, REAL, MONEY, SMALL MONEY
	Informix	INTEGER, SMALLINT, DECIMAL, FLOAT, REAL, DOUBLE PRECISION, SMALLFLOAT, SERIAL, MONEY, NUMERIC
Binary	DB2	CHAR (for Bit Data), VARCHAR (for Bit Data), BLOB
	Oracle	RAW, LONG RAW
	Sybase ASE	BINARY, VARBINARY, IMAGE
	SQL Server	BINARY, VARBINARY, IMAGE
	Informix	BYTE
Boolean	Sybase ASE	BOOLEAN (TRUE or FALSE)
Datetime	DB2	TIMESTAMP
	Oracle	DATE, TIMESTAMP, TIMESTAMP WITH LOCAL TIME ZONE, TIMESTAMP WITH TIME ZONE
	Sybase ASE	DATETIME, SMALL DATE TIME
	SQL Server	DATETIME, SMALL DATE TIME
	Informix	DATE, DATETIME

Class	DBMS	Data Types
Date	DB2	DATE
	Oracle	DATE
	Informix	DATE
Time	DB2	TIME
Interval	Oracle	YEAR/MONTH INTERVAL, DAY/SECOND INTERVAL
	Informix	YEAR/MONTH INTERVAL, DAY/TIME INTERVAL

Comparison Compatibility Rules

The following table defines the compatibility of data types for comparisons. In some cases, columns can be mapped, but cannot be used as Match Key columns. For example, if you compare source data type Char to source data type Numeric, the data types are compatible, but the pairing cannot be specified as a Match Key.

Source Data Type	Char (1)	Numeric	Boolean	Date/Time (2)	Interval
Char (1)	Yes (9)	Yes (4)	Yes (4, 8)	No	No
Numeric	Yes (4)	Yes	Yes (4, 7)	No	No
Boolean	Yes (4, 8)	Yes (4, 7)	Yes	No	No
Date/Time (2)	No	No	No	Yes (3, 5)	No
Interval	No	No	No	No	Yes (6)

Note:

1. Includes Char, VarChar, NChar, NVarChar, Binary, VarBinary. You cannot compare multi-byte or Unicode character data to binary or single-byte character data.
2. Includes Date, Time, DateTime, Timestamp, Timestamp with Time Zone, Timestamp with Local Time Zone.
3. It is recommended that you compare Oracle Timestamp with Time Zone columns to Timestamp with Time Zone columns only, and Timestamp with Local Time Zone columns to Timestamp with Local Time Zone columns only. Comparison to columns of different data types may produce unexpected results due to Oracle assumptions for storing and retrieving this data.
4. Pairing cannot be specified as a Match Key column.
5. Comparison is valid as long as both columns have the same high-order component. For example, a DB2 Timestamp may be compared with an Informix DateTime column only when the Informix column includes a ‘year' component.
6. YearMonth interval columns can be compared to YearMonth interval columns only. DayTime or DaySecond interval columns can be compared to DayTime or DaySecond interval columns only.
7. In Numeric/Boolean comparisons, when Numeric value is 0, then Boolean value is False. When Numeric value is anything other than 0, then Boolean value is True.
8. In Char/Boolean comparisons, when Char value starts with ‘T', ‘t', ‘Y', 'y', or ‘1', Boolean value is True. When Char value starts with ‘F', ‘f', ‘N', 'n', ‘0', Boolean value is False.
9. Comparisons of binary with multi-byte or Unicode data are not valid.

Mapping Column Data

In the following discussion, the mapping of data types from source to destination is grouped according to the destination data type:

Character Destination

Source	Destination	Mapping
Character	Character	Data is left-justified and truncated or padded with spaces on the right, as needed.
Numeric	Character	Data is right-justified with leading zeros, as needed. Leading spaces are used, as needed, for floats. Destination column must be large enough to hold significant portion of the source numeric value; otherwise error results.
Binary	Character	Data is left-justified and truncated or padded with spaces on the right, as needed. There is no conversion. **Note:** You cannot map a binary source column to a multi-byte or Unicode destination.
Date/Time	Character	Fixed date/time format applies, based on source column. (See fixed formats, below.)

Note:

- A character data type must have a minimum length of five, CHAR(5), to map floating point data types.
- If you map a source date/time column to a destination character column, you can specify a different date format using the Age Function.
- The following fixed formats apply when you map a date/time column to a character column:

Date/Time	Sybase ASE	YYYY/MM/DD-HH:MM:SS.FFF
	SQL Server	YYYY/MM/DD-HH:MM:SS.FFF
	Informix	YYYY-MM-DD HH:MM:SS.FFFFFF (based on high/low qualifier)
Small Date/Time	Sybase ASE	YYYY/MM/DD-HH:MM:SS
	SQL Server	YYYY/MM/DD-HH:MM:SS
Date	DB2	YYYY/MM/DD
	Oracle	YYYY/MM/DD-HH:MM:SS
Time	DB2	HH:MM:SS
Timestamp	DB2	YYYY/MM/DD-HH:MM:SS.FFFFFF
	Oracle	YYYY/MM/DD-HH:MM:SS.FFFFFFFFF
Timestamp w/Time Zone	Oracle	YYYY/MM/DD-HH:MM:SS.FFFFFFFFF +/-HH:MM
Timestamp w/Local Time Zone	Oracle	YYYY/MM/DD-HH:MM:SS.FFFFFFFFF
Day/Time Interval	Informix	[-]DDDDDDDDD HH:MM:SS.FFFFF
Day/Second Interval	Oracle	[-]DDDDDDDDD HH:MM:SS.FFFFFFFFF
Year/Month Interval	Informix	[-]YYYYYYYYY-MM
	Oracle	[-]YYYYYYYYY-MM

Note:

- FFF equals hundredths of a second, FFFFFF equals millionths of a second, and FFFFFFFFF equals billionths of a second. Note that there will only be as many fractional seconds (F) decimal places as the specified scale.
- For Timestamp with Time Zone columns, +/-HH:MM represents a time zone value, where HH is between -12 and +13.

- For interval columns, [-] represents an optional negative sign. Also, note that the maximum digits are shown; however, there will only be as many day (D) digits or year (Y) digits as the specified precision.
- Valid date values range from 1 to 9999.

Numeric Destination

Source	Destination	Mapping
Character	Numeric	Character data must be valid as numeric data. Destination column must be large enough to hold source value; otherwise, error results. Decimal places can be dropped.
Numeric	Numeric	Destination column must be large enough to hold significant portion of the source value; otherwise, error results. Truncate decimals, if needed.
Binary	Numeric	Binary data is transformed to numeric. Destination column must be large enough to hold source value; otherwise, error results. Decimal places can be dropped.

Note:

- If a destination column is defined as numeric, you cannot map a source column defined as date/time. Use the Age Function.
- If the process of mapping significant positions of numeric data results in truncating the data, the data is not mapped and an error is reported.

Binary Destination

Source	Destination	Mapping
Character	Binary	Data is left-justified and truncated or padded with NULL on the right, as needed. There is no conversion. **Note:** You cannot map a multi-byte or Unicode source column to a binary destination.
Numeric	Binary	Destination column must be large enough to hold significant portion of source value; otherwise, error results. Data is right⌂justified with leading zeros for integers and decimals. Leading spaces are used as needed.
Binary	Binary	Data is left-justified and truncated or padded with NULL on the right, as needed.

Note: If a destination column is defined as binary, you cannot map a source column defined as date/time, nor use the AGE function.

Boolean Destination

Source	Destination	Mapping

Character	Boolean	Data is mapped from the first character. If the first character is 'T' , 'Y' , or '1', translate to TRUE. If the first character is 'F' , 'N', or '0', translate to FALSE.
Numeric	Boolean	A zero numeric value translates to FALSE. A non-zero numeric value translates to TRUE.
Binary	Boolean	Data is mapped from the first character. If the first character is 'T' , 'Y' , or '1', translate to TRUE. If the first character is 'F' , 'N', or '0', translate to FALSE.
Boolean	Boolean	Equal

Note: If a destination column is defined as Boolean, you cannot map a source column defined as date/time nor use the AGE function.

Date/Time Destination

Source	Destination	Mapping
Date	Date	Equal
Time	Date	Not Supported
Timestamp	Date	Use date portion only.
Timestamp w/Time Zone	Date	Use date portion only; drop time zone.
Timestamp w/Local Time Zone	Date	Use date portion only.
Date	Time	Not Supported
Time	Time	Equal
Timestamp	Time	Use time portion only.
Timestamp w/Time Zone	Time	Use time portion only; drop time zone.
Timestamp w/Local Time Zone	Time	Use time portion only.
Date	Timestamp	Use source date and midnight.
Time	Timestamp	Use source time and current date.
Timestamp	Timestamp	Equal
Timestamp w/Time Zone	Timestamp	Use timestamp portion; drop time zone.
Timestamp w/Local Time Zone	Timestamp	Equal
Date	Timestamp w/Time Zone	Use source date, midnight, and “+0”.
Time	Timestamp w/Time Zone	Use source time, current date, and “+0”.
Timestamp	Timestamp w/Time Zone	Use timestamp and “+0”.
Timestamp w/Time Zone	Timestamp w/Time Zone	Equal
Timestamp w/Local Time Zone	Timestamp w/Time Zone	Use timestamp with “+0”.
Date	Timestamp w/Local Time Zone	Use source date and midnight.
Time	Timestamp w/Local Time Zone	Use source time and current date.

Timestamp	Timestamp w/Local Time Zone	Equal
Timestamp w/Time Zone	Timestamp w/Local Time Zone	Use timestamp portion; drop time zone portion.
Timestamp w/Local Time Zone	Timestamp w/Local Time Zone	Equal

Note:

1. If a destination column is defined as date, time, or timestamp, you can specify a date/time special register as the source.
2. If a destination column is defined as date/time, you cannot map a source column defined as character or numeric. Use the Age Function.
3. If a destination column is defined as date/time, you cannot map a source column defined as binary or Boolean. Use the Age Function.
4. If the destination column is a Sybase ASE/SQLServer DateTime or SmallDateTime column, AND the source column is NOT a Sybase ASE/SQLServer DateTime or SmallDateTime column, AND the hour value is greater than or equal to 24, AND the **SybTimeOverflow** option is TRUE, the destination time value is set to 00:00:00.000000.
5. If the source or destination column is an Oracle Timestamp with Time Zone or Timestamp with Local Time Zone column, mapping to a column of a different data type may produce unexpected results due to Oracle assumptions for storing and retrieving this data.

   For example, for a Timestamp with Local Time Zone column, Oracle normalizes the time stored in the database to the session time of the Oracle server. When Oracle retrieves this data, the stored value is adjusted by the difference between the server session time and the client session time. However, Optim returns the value stored in the database without making adjustments.

   Therefore, it is recommended that you map Timestamp with Time Zone columns and Timestamp with Local Time Zone columns to columns of the same date type only.

Special Registers

The following are various special registers or functions for putting the current date and/or time into a Destination column. They may be used interchangeably.

CURRENT DATE	CURRENT TIMESTAMP	GETDATE
CURRENT_DATE	CURRENT_TIMESTAMP	GETTIME( )
CURRENT TIME	CURDATE	NOW( )
CURRENT_TIME	CURTIME( )	SYSDATE

The following are special registers for putting a User ID into a Destination column. Note that they do not provide the same value.

CURRENT_SQLID
User ID as known to database

USER User ID as known to database

WORKSTATION_ID
User ID as known to Windows 95/NT

Note: Mapping special registers is based on the length of the destination column.

Chapter 6. Column Map Procedures

A Table Map used in a process may reference one or more Column Maps used to derive the appropriate values for destination columns. The Table Map identifies and matches two sets of tables for processing, and a Column Map provides a way to control processing, column by column. With a Column Map, you can match or exclude columns from processing or, using native functions, derive values for one or more destination columns.

(For details on defining Column Maps, see Chapter 5, "Column Maps," on page 121.)

For data transformations in a Convert, Insert, Load, or Restore Process that are beyond the scope of native Column Map functions, you can specify an exit routine or, for processes that run in a Windows environment, a Column Map Procedure for a source column. (Exit routines, which are programs that conform to the C programming language, are discussed in Appendix B, "Exit Routines for Column Maps," on page 473.) If the process is delegated to an Optim Server on a Windows machine, the Column Map Procedure is also executed on the Server.

Note: You cannot run a Column Map Procedure in a UNIX or Linux environment.

The function of a Column Map Procedure is generally the same as that of an exit routine. An exit routine must exist in a DLL external to Optim, and must conform to calling conventions used in the C programming language. A Column Map Procedure, however, is stored in the Optim Directory and is written in Optim Basic, which is distributed with Optim.

An exit routine may execute more efficiently than a Column Map Procedure but is written externally, and must be externally compiled and linked. A Column Map Procedure, on the other hand, is written within Optim and can be embedded in a Column Map. An external compile and link, and complicated calling conventions, are not required.

One function of a Column Map Procedure is to generate values that could not otherwise be defined for the destination column. This function is useful for handling special processing and data manipulation according to site⌂defined rules. During processing, Optim runs the Column Map Procedure, which is a program that derives values for the corresponding destination column. Column Map Procedures are not limited to data transformation, however. You can also use a Column Map Procedure to reject rows on the basis of custom processing, to create a report tailored to the needs of your site, or to implement conditional data transformations.

Procedure in a Column Map

To reference a named procedure in a Column Map, use the following syntax for the appropriate source column:

PROC ***identifier.name*** **[(["*parm1*" [,"*parmn*"]])]**

To reference a local procedure in a Column Map, use the following syntax for the appropriate source column:

PROC LOCAL [(["*parm1*" [,"*parmn*"]])]

As many as eight optional string or numeric constants, separated by commas, are passed to the procedure when it is called. String constants must be enclosed in single quotes.

Naming Conventions

The fully qualified name of a Column Map Procedure has two parts: *identifier.name*.

identifier
: Identifier assigned to the Column Map Procedure (1 to 8 characters).

name
: Name assigned to the Column Map Procedure (1 to 12 characters).

When you create Column Map Procedures, use a logical set of naming conventions to identify and organize definitions for easy access.

Contents

This section provides detailed information on how to:

- Create a new Column Map Procedure.
- Open an existing Column Map Procedure.
- Specify a Column Map Procedure in a Column Map.
- Convert a named Column Map Procedure to a Local Column Map Procedure.
- Save a Column Map Procedure.
- Delete a Column Map Procedure.

Open the Column Map Procedure Editor

Use the Column Map Procedure Editor to create and maintain Column Map Procedures. There are different ways to open the editor depending on whether you want to create a new Column Map Procedure or select a Column Map Procedure to edit.

Create a Column Map Procedure

This section explains how to create a Column Map Procedure from the main window or the Column Map Procedure Editor.

From the Main Window

To create a Column Map Procedure:

1. In the main window, select **New** from the **File** menu.
2. Select **Column Map Proc** from the **Definitions** submenu to open Column Map Procedure Editor.
3. In the **Description** box, optionally type a description that indicates the purpose of the Column Map Procedure.
4. In the code window, edit code in the sample Column Map Procedure provided with Optim to carry out your purpose.
5. In the **File** menu, select **Compile** to validate the code. If the code generates an error, an arrow is placed on a line to indicate an error. Correct the code, as necessary.
6. Select **Save** from the **File** menu to open the Save a Column Map Proc dialog.
7. In the **Pattern** box, specify a unique Column Map Procedure name and click **Save**.

From the Column Map Procedure Editor

To create a Column Map Procedure:

1. In the main window, select **Column Map Proc** from the **Definitions** menu to open the Column Map Procedure Editor and most recently edited Column Map Procedure.
2. Your next step depends on your purpose:

- To create a new Column Map Procedure, select **New** from the **File** menu to display the sample Column Map Procedure provided with Optim.
- To create a new Column Map Procedure modeled on an existing one, open the desired Column Map Procedure and select **Save As** from the **File** menu.
- To create and store a copy of the active Column Map Procedure and continue editing, select **Save Copy As** from the **File** menu.

These steps are the minimum required to create a Column Map Procedure. The steps to edit a new Column Map Procedure or existing Column Map Procedure are similar. See "Using the Editor" for details.

Note: You can also open the Column Map Proc Editor while editing a Column Map by right-clicking a Source Column grid cell and selecting **List Procedures** or **Open Procedure** from the short-cut menu.

Select a Column Map Procedure to Edit

You can select a Column Map Procedure for editing from the main window or from the Column Map Procedure Editor.

From the Main Window

To select a Column Map Procedure to edit:

1. In the main window, select **Open** from the **File** menu to display the Open (object selection) dialog. (See "Using the Open Dialog" on page 25 for more information on the Open dialog.)
2. Double-click **Column Map Proc** to expand the Identifier list.
3. Double-click the desired identifier to display a list of Column Map Procedures.
4. **Optional** – Specify a **Pattern** to limit the list based on your criteria and click **Refresh**.
5. Double-click the desired Column Map Procedure to open the Column Map Proc Editor.

Note: To select the last Column Map Procedure you edited, select **Column Map Proc** from the **Definitions** menu in the main window to open the Column Map Proc Editor and most recently edited Column Map Procedure.

From the Column Map Proc Editor

To select a different Column Map Procedure:

1. In the Column Map Proc Editor, select **Open** from the **File** menu to display the Select a Column Map Proc dialog.
2. Double-click the desired identifier to display a list of Column Maps Procedures.
3. **Optional** – Specify a **Pattern** to limit the list based on your criteria and click **Refresh**.
4. Double-click the desired Column Map Procedure to open the Column Map Proc Editor.

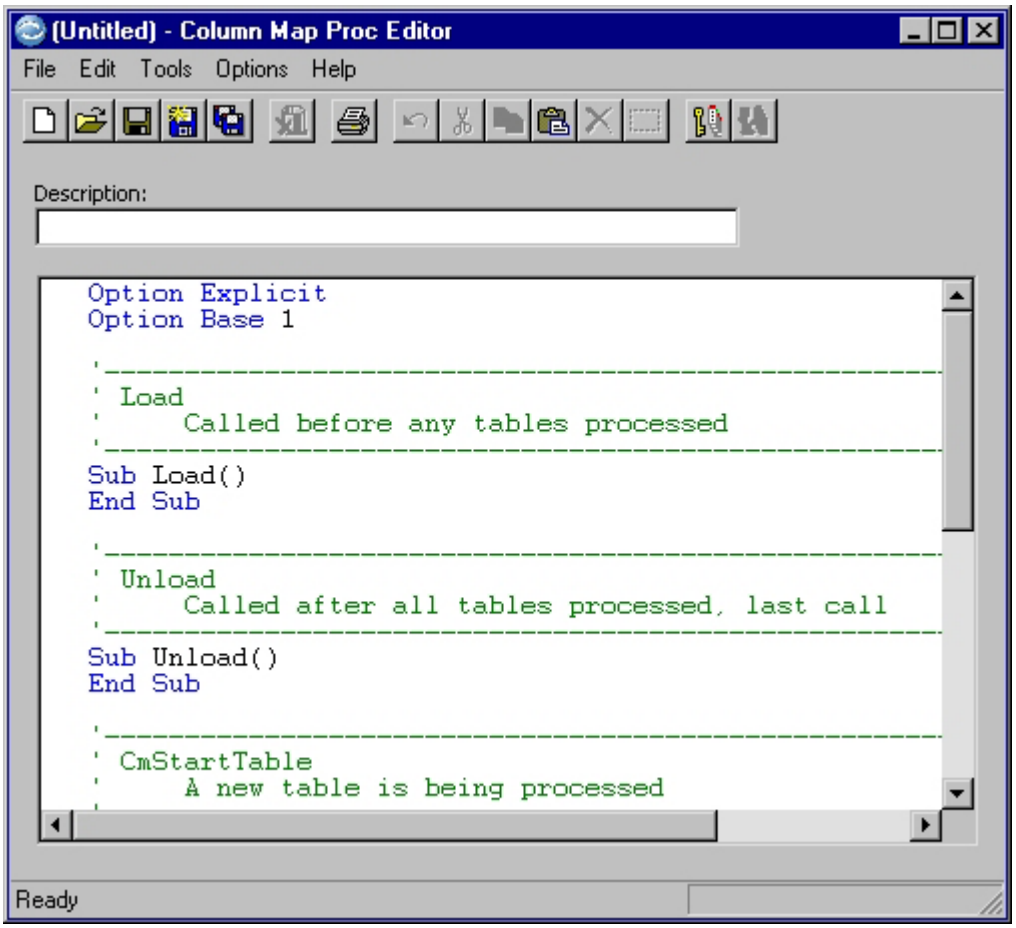
Sample Column Map Procedures

Optim application files on the installation CD include sample Column Map Procedures. The samples show how to use a Column Map Procedure and provide examples of the type of processing that can be performed using the sample database.

Refer to the *Installation and Configuration Guide* for a description of each sample.

Using the Editor

In the Column Map Proc Editor you can create, modify, or delete Column Map Procedures as objects stored in the Optim Directory or defined within a Column Map. You can also export a file that contains information needed to create a Column Map Procedure or import a similar file in order to create a Column Map Procedure.



Description

Text to describe or explain the purpose of the Column Map Procedure (up to 40 characters).

Edit Window

When you open the editor, the window displays either:

- The last edited Column Map Procedure (if any exists). You can edit an existing procedure, or save it as a new procedure and edit as necessary.
- The DEFAULT.BAS Column Map Procedure stored in the SOFTECH\RT\BIN directory. This code template is displayed each time you select **New** from the **File** menu within the editor, whether from the main menu or from within the editor.

Note: A copy of DEFAULT.BAS is distributed with Optim and stored in the SOFTECH\RT\BIN directory. If desired, you can modify DEFAULT.BAS to meet site requirements or standards and export it to the SOFTECH\RT\BIN directory. You can also import a Column Map Procedure before exporting it to SOFTECH\RT\BIN\DEFAULT.BAS.

Template Column Map Procedure

Intended for use as a guideline, the template Column Map Procedure includes the following native functions:

Sub Load ... ... End Sub	The Load function can be included in a Column Map Procedure. This optional function is called before any tables are processed. You can use this function for initialization or other tasks that apply generally in the procedure.
Function CmStartTable () As Integer ... ... End Function	The CmStartTable function can be included in a Column Map Procedure. This optional function is called before each table is processed. You can use this function for initialization or other tasks that affect or apply to each table processed in the procedure.
Function CmTransform () As Integer ... Target.Value = Source.Column _ (Target.Column).Value ... End Function	The CmTransform function must be included in a Column Map Procedure. This required function is called for each row that is processed and provides instructions for the data transformation to be performed.
Function CmEndTable ... ... End Function	Only one instance of the CmEndTable function can be included in a Column Map Procedure. This optional function is called after each table is processed. You can use this function for tasks required after each table is processed.
Sub Unload() ... ... End Sub	Only one instance of the Unload function can be included in a Column Map Procedure. This optional function is called after all tables are processed. You can use this function for tasks required before termination of the procedure.

Of the native functions, only the CmTransform function is required; others are optional. The CmTransform function in the template simply copies each source value to a target value. You can add your own functions to be called by one or more native functions or otherwise edit the template to generate destination values or reports that meet your requirements.

Return Codes

The CmStartTable and CmTransform functions must return one of the following codes:

Return Code	Mnemonic	Explanation
0	PST_CM_EXIT_SUCCESS	Procedure executed successfully
1	PST_CM_EXIT_REJECT_ROW	Procedure rejected row
2	PST_CM_EXIT_ABORT_PROCESS	Procedure detected abort condition

Available functions and statements and additional reserved words are listed in the following tables. See the *Optim Basic Language Reference* for syntax and other information about the statements and functions.

General Statements and Functions

Abs	Access	Alias	And	Any
App	AppActivate	Asc	Atn	As
Base	Begin	Binary	ByVal	Call
Case	CBool	CDate	ChDir	ChDrive
Choose	Chr	Const	Cos	CurDir
CDbl	CInt	CLng	CSng	CStr
CVar	Close	CreateObject	Date	DateSerial
DateValue	Day	Declare	Dim	Dir
Do...Loop	DDEInitiate	DDEExecute	Double	Else
ElseIf	End	EndIf	EOF	Eqv
Erase	Err	Error	Exit	Exp
Explicit	False	FileCopy	FileLen	Fix
For	For Each...Next	For...Next	Format	FreeFile
Function	Get	GetAttr	Get Object	Global
GoTo	Hex	Hour	If...Then...Else...[End If]	Imp
Input	Input #	InputBox	InStr	Int
Integer	Is	IsArray	IsEmpty	IsNull
IsNumeric	IsDate	IsObject	Kill	LBound
LCase	Left	Len	Let	LOF
Log	Long	Loop	LTrim	Line Input #
Mid	Minute	MkDir	Mod	Month
Name	Next	Not	Now	Oct
On	Open	Object	Option	Optional
Or	On Error	Option Base	Option Explicit	Print
Print #	Put	Randomize	Rem	ReDim
RmDir	Rnd	Rtrim	Right	Seek
SendKeys	Set	SetAttr	Second	Select
Select Case	Shell	Sin	Sqr	Stop
Str	Sng	Single	Space	Static
Step	Stop	Str	String	StrComp
Sub	Tan	Time	Timer	TimeSerial
Then	Trim	True	To	Type
TimeValue	UBound	UCase	Until	Val
Variant	VarType	Write #	Weekday	With
While...Wend	Xor	Year		

Functions and Statements by Type

Flow of Control

Do...Loop	End	Exit For	Exit Loop

For Each...Next	For...Next	GoTo	If...Then...Else...[End If]
OnError	Select Case	Stop	While...Wend

Conversion

Asc	CBool	CDate	CDbl
Chr	CInt	Clng	CSngr
CStr	CVar	Date	DateSerial
DateValue	Day	Fix	Format
Hex	Hour	Int	Minute
Month	Oct	Second	Str
TimeSerial	TimeValue	Val	Weekday
Year			

File I/O

ChDir	ChDrive	Close	CurDir
Dir	EOF	FileCopy	FileLen
FreeFile	Get	GetAttr	Input
Kill	Line Input	LOF	MkDir
Name	Open	Print #	Put
RmDir	Seek	SetAttr	Write #

Math

Abs	Atn	Cos	Exp
Fix	Int	Log	Rnd
Sgn	Sin	Sqr	Tan

Procedures

Call	Declare	End Function	End Sub
Exit	Function	Global	Sub

Strings

Asc	Chr	InStr	LCase
Left	Len	Let	LTrim
Mid	Option Compare	Right	RTrim
Space	StrComp Format	String	Trim
UCase			

Variables and Constants

Const	Dim	Global	IsDate

IsEmpty	IsNull	IsNumeric	Option Explicit
Static	VarType		

Error Trapping

On Error	Resume

Date/Time

Date	Now	Time	Timer

DDE

DDEExecute	DDEInitiate	DDETerminate

Arrays

Dim	Erase	Global	Lbound
Option Base	Option Explicit	ReDim	Static
Ubound			

Miscellaneous

AppActivate	CreateObject	GetObject	Randomize
Rem	SendKeys	Shell	

Objects

Application Properties

Arg	ArgCount	CompanyName	ComputerName
DataDir	Environ	Error	Instance
LogEvent	Opsys	OpsysBuild	OpsysCSD
OpsysRelease	Request	RequestList	RtBuild
RtRelease	Script	ServerUserId	StartLogging
TempDir	ThreadHandle	ThreadId	

Column Properties

Available	IsNull	Length	Name
Nullable	Precision	Scale	Type
Value			

Source Properties

Column	ColumnList	CreateFile	CreatorId

DBAlias	IsColumn	OwnerId	Table

Target Properties

Column	SetFromFile	Value

Global Functions

Function	Syntax
GetAsBinary	GetAsBinary(*column,length*)
GetAsDate	GetAsDate(*column,pointer*)
GetAsDecimalChar	GetAsDecimalChar(*column,precision*[,*scale*])
GetAsDecimalCharSz	GetAsDecimalCharSz(*column,precision,scale*)
GetAsOracleDate	GetAsOracleDate(*column,pointer*)
GetAsSybaseDateTime	GetAsSybaseDateTime(*column,pointer*)
GetAsSybaseDecimal	GetAsSybaseDecimal(*column,pointer*)
GetAsSybaseMoney	GetAsSybaseMoney(*column,pointer*)
GetAsSybaseSmallDateTime	GetAsSybaseSmallDateTime(*column,pointer*)
GetAsSybaseSmallMoney	GetAsSybaseSmallMoney(*column,pointer*)
GetAsTime	GetAsTime(*column,pointer*)
GetAsVarBinary	GetAsVarBinary(*column,pointer*)
GetAsVarChar	GetAsVarChar(*column,pointer*)
GetAsVarCharSz	GetAsVarCharSz(*column,pointer*)
GetGlobalWork	GetGlobalWork(*area,length*)
PutAsBinary	PutAsBinary(*column,length*)
PutAsDate	PutAsDate(*column,pointer*)
PutAsDecimalChar	PutAsDecimalChar(*pointer*)
PutAsDecimalCharSz	PutAsDecimalCharSz(*pointer*)
PutAsOracleDate	PutAsOracleDate(*column,pointer*)
PutAsSybaseDateTime	PutAsSybaseDateTime(*column,pointer*)
PutAsSybaseDecimal	PutAsSybaseDecimal(*column,pointer*)
PutAsSybaseMoney	PutAsSybaseMoney(*column,pointer*)
PutAsSybaseSmallDateTime	PutAsSybaseSmallDateTime(*column,pointer*)
PutAsSybaseSmallMoney	PutAsSybaseSmallMoney(*column,pointer*)
PutAsTime	PutAsTime(*column,pointer*)
PutAsVarBinary	PutAsVarBinary(*column,pointer*)
PutAsVarChar	PutAsVarChar(*column,pointer*)
PutAsVarCharSz	PutAsVarCharSz(*column,pointer*)
SetGlobalWork	SetGlobalWork(*pointer,length*)

Variable Data Types

Variable	Type Specifier	Usage	Size
String	$	Dim Str_Var As String	0 to 16,000 char
Integer	%	Dim Int_Var As Integer	2 bytes
Long	&	Dim Long_Var As Long	4 bytes
Single	!	Dim Sing_Var As Single	4 bytes
Double	#	Dim Sbl_Var As Double	8 bytes
Variant		Dim X As Any	4 bytes
Boolean		Dim X As Boolean	True or False
Byte		Dim X As Byte	0 to 225
Object		Dim X As Object	4 bytes
Date		Dim D As Date	8 bytes
Currency		Dim Cvar As Currency	8 bytes
User Defined Types			size of each element

Arithmetic Operators

Arithmetic operators follow mathematical rules of precedence. '+' or '&' can be used for string concatenation.

Operator	Function	Usage
^	exponentiation	x = y^2
-	negation	x = -2
*	multiplication	x% = 2 * 3
/	division	x = 10/2
Mod	modulo	x = y Mod z
+	addition	x = 2 + 3
-	subtraction	x = 6 - 4

Relational Operators

Operator	Function	Usage
<	less than	x < Y
<=	less than or equal to	x <= Y
=	equals	x = Y
>=	greater than or equal to	x >= Y
>	greater than	x > Y
<>	not equal to	x <> Y

Logical Operators

Operator	Function	Usage
Not	Logical Negation	If Not (x)
And	Logical And	If (x>y) And (x<Z)

Operator	Function	Usage
Or	Logical Or	If (x=y) Or (x=z)

Operator Precedence

Operator	Description	Order
()	parenthesis	highest
^	exponentiation	↓
-	unary minus	↓
/,*	division/multiplication	↓
mod	modulo	↓
+, -, &	addition, subtraction, concatenation	↓
=, <>, <, >,<=,>=	relational	to
not	logical negation	↓
and	logical conjunction	↓
or	logical disjunction	↓
Xor	logical exclusion	↓
Eqv	logical equivalence	↓
Imp	logical implication	lowest

Menu Commands

In addition to the standard commands, you can select the following commands from the **File** menu:

Compile
Compile the Column Map Procedure and indicate errors. This command is also available from the right-click menu.

Export Open the Supply an Export File Name dialog and export the Column Map Procedure displayed in the Column Map Proc Editor. See "Export a Column Map Procedure."

Import
Open the Supply an Import File Name dialog and display the Column Map Procedure in the Column Map Proc Editor. While you can import files created using an external editor, imported code may require modification to be compiled for use with a Column Map. See "Import a Column Map Procedure" on page 186.

In addition to the standard commands, you can select the following command from the **Edit** menu:

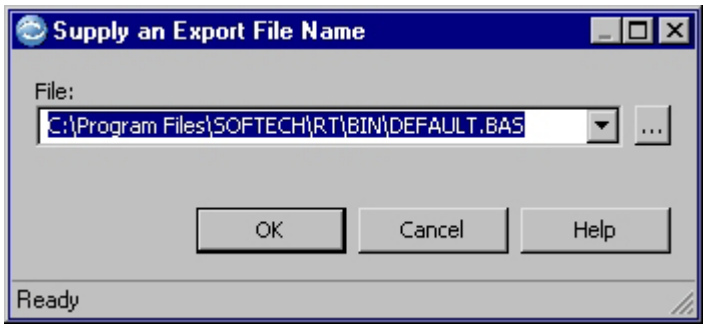
Goto Line
Move the cursor to a specified line in the Column Map Procedure. This command is also available from the right-click menu and is useful for locating a line referenced by number in an error message.

Export a Column Map Procedure

When you save a Column Map Procedure as an object in the Optim Directory, any Column Map in the same Directory can reference it. At times you may also want to generate a file that includes information needed to create a Column Map Procedure. The file can then be used, perhaps on a different machine or while connected to a different Optim Directory, to populate the Column Map Proc Editor. From the Column Map Proc Editor, the information in the file can be saved as an object in the Optim Directory. Use the Column Map export function to create a file that contains a Column Map Procedure.

To export a Column Map Procedure:

1. In the Column Map Proc Editor, open a Column Map Procedure.
2. Select **Export** from the **File** menu to display the Supply an Export File Name dialog.



3. Supply a File name.
   - Click the down arrow to select from a list of recently used file names.
   - Click the browse button to select from another directory folder.
   - Type the full directory path and file name.
4. Click **OK** to create the export file containing the Column Map Procedure.

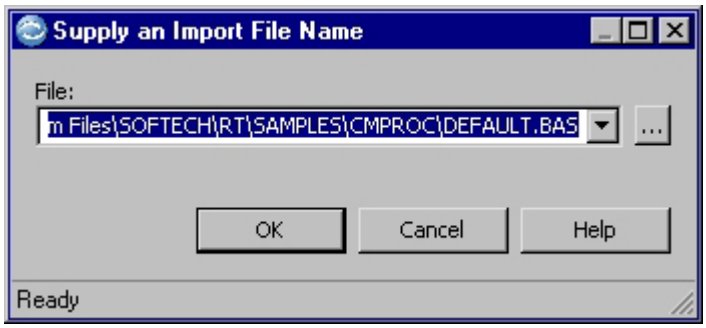
By default, exported Column Map Procedure files have a .BAS extension. You can also use a .TXT or any other extension for these files.

Import a Column Map Procedure

The import function allows you to populate the Column Map Proc Editor with information from a file. The file may have been exported from a different machine or while connected to a different Optim Directory, or created using an external editor (e.g., a text editor). From the Column Map Proc Editor, the information in the file can be saved as an object in the Optim Directory.

To import a Column Map Procedure:

1. In the Column Map Proc Editor, select **Import** from the **File** menu to display the Supply an Import File Name dialog.



2. Supply a File name.
   - Click the down arrow to select from a list of recently used file names.
   - Click the browse button to select from another directory folder.
   - Type the full directory path and file name.
3. Click **OK** to import the file containing the Column Map Procedure.
4. Edit, compile and save the imported Column Map Procedure as desired.

If you do not specify an extension for the file name, it is given a .BAS extension. You can also import a file with a .TXT or any other extension.

Save a Column Map Procedure

The save commands are available from the **File** menu in the Column Map Proc Editor. When you save a Column Map Procedure, you must supply a two-part name: *identifier.name*.

Save

The Save command allows you to save a new Column Map Procedure or update an existing one:

- To save a new procedure, select **Save** from the **File** menu in the Column Map Proc Editor to open the Save the Column Map Proc dialog. Type a name in the **Pattern** box and click **Save**.
- To save an existing procedure, select **Save** from the **File** menu in the Column Map Proc Editor. The current version replaces the original version. You can continue editing the current version.

Save As

Use the Save As command to save a Column Map under a new name, preserve the original, and display the newly named version for editing. To use this command, select **Save As** from the **File** menu in the Column Map Proc Editor. In the Save the Column Map Proc dialog, type a name in the **Pattern** box and click **Save**.

Save Copy As

Use the Save Copy As command to save a copy of a procedure under a new name, preserve the copy, and continue editing the original. To use this command, select **Save Copy As** from the **File** menu in the Column Map Proc Editor. In the Save the Column Map Proc dialog, type a name in the **Pattern** box and click **Save**.

Delete a Column Map Procedure

When you delete a Column Map Procedure, the definition is removed from the Optim Directory. You can delete a Column Map Procedure from the editor or from the Open dialog.

From the Editor

To delete a procedure from the Column Map Proc Editor, display the procedure you want to delete and select **Delete** from the **File** menu. In the confirmation popup, click **Yes** to confirm the delete.

From the Open Dialog

To delete a procedure from the Open dialog, select **Open** from the **File** menu in the Column Map Proc Editor or from the **File** menu in the main window. Select the procedure you want to delete, then right-click to select **Delete** from the shortcut menu. In the confirmation popup, click **Yes** to confirm the delete.

Confirm Option

By default, you are prompted to confirm before deleting a definition. To disable this feature, select **Personal** from the **Options** menu in the main window, then select the **Confirm** tab.

Chapter 7. DB Aliases

A DB Alias is a user-defined object associated with a database. When you define a DB Alias, you provide parameters that Optim uses to communicate with that database. These parameters include the type and version of the database management system (DBMS) and any required connection specifications.

Note: DB Aliases are defined when you configure Optim. You must be authorized to define a DB Alias. See the *Installation and Configuration Guide* for additional information.

A DB Alias name serves as a high-level qualifier that allows you to access a specific database to perform requested functions. For example, in an Access Definition, you must qualify the name of a table with a DB Alias name. The referenced DB Alias supplies the parameters needed to connect to the database in which the table resides.

Databases

Optim supports several different database management systems. Since DB Alias parameters are DBMS-specific, the most common terms are used in this chapter where possible. Terms unique to database management systems are provided when necessary.

A DB Alias identifies a specific database and serves as a prefix in the fully qualified names of primary keys, database tables, and relationships. DB Aliases are essential elements in managing your databases. Keep in mind that each:

- Database can have only one DB Alias.
- DB Alias name must be unique.

Also, objects in the Optim Directory cannot have the same name as a DB Alias.

Contents

This section describes the use of the DB Alias Editor to maintain DB Aliases. Included are specifications for DB Aliases, including general parameters, connection specifications, and server details.

Open the DB Alias Editor

Use the DB Alias Editor to maintain DB Aliases. When you browse a DB Alias, the DB Alias Editor is displayed.

From the Main Window

To browse a DB Alias:

1. In the main window, select **Open** from the **File** menu to display the Open (object selection) dialog. (See "Using the Open Dialog" on page 25 for more information on the Open dialog.)
2. Double-click **DB Alias** to display a list of DB Aliases.
3. *Optional* – Specify a **Pattern** to limit the list based on your criteria and click **Refresh**.
4. Double-click the desired **DB Alias** name to open the DB Alias Editor.

Note: To browse the last DB Alias you reviewed, select **DB Alias** from the **Definitions** menu in the main window to open the DB Alias Editor and last edited DB Alias.

From the DB Alias Editor

To browse a different DB Alias:

1. In the DB Alias Editor, select **Open** from the **File** menu to display the Open a DB Alias dialog. (See "Using the Open Dialog" on page 25for more information on the Open dialog.)
2. *Optional* – Specify a **Pattern** to limit the list based on your criteria and click **Refresh**.
3. Double-click the desired DB Alias to open the DB Alias Editor.

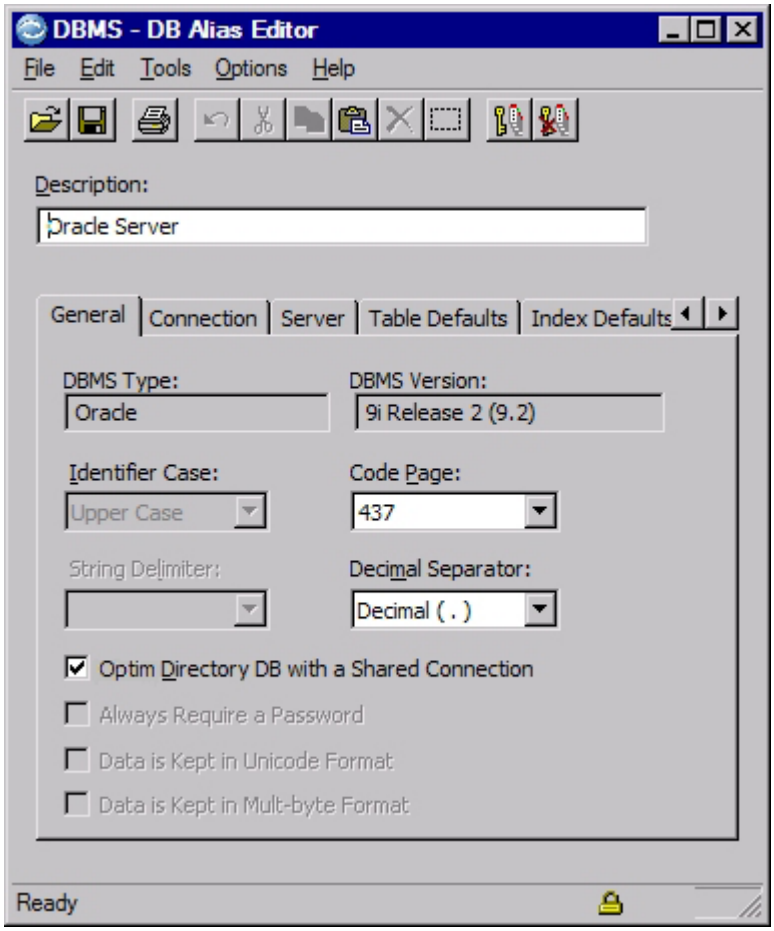
Using the Editor

The DB Alias Editor allows you to browse DB Alias names and other information that permit Optim to communicate with a specific database. You can also set defaults for objects created by Optim.

Use the options on the **Defaults** tabs to set defaults for creating objects. Note that Optim allows you to establish as many as three layers of default settings for the creation of database objects, in addition to target system defaults. The default settings determine the values displayed in the Object List for the Create Utility and can be overridden at the object level by editing the list.

At the broadest level, DB Alias settings, as described further on, establish defaults for creating objects in the associated database. If desired, you can provide Personal Options settings for a user or group of users that override some or all DB Alias settings. A third level of optional defaults apply at the processing level to override Personal Options and DB Alias settings. Use the options on the **Defaults** tabs to set first-level defaults for the Create Utility.

Note: For information on the correct syntax for entries, refer to the documentation provided with your database management system.



Description

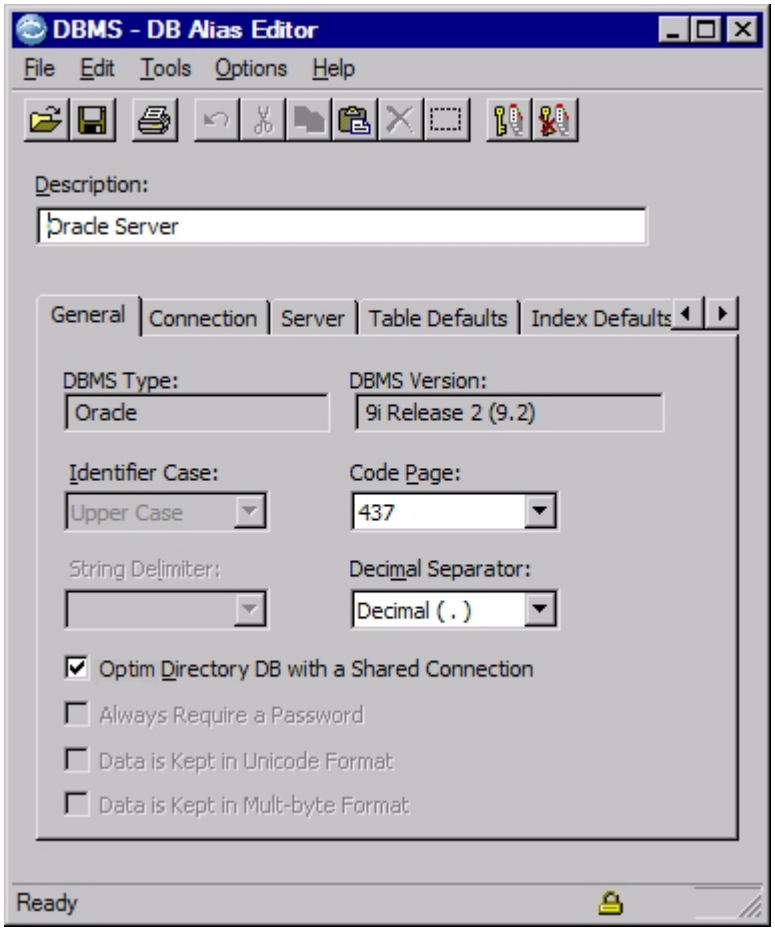
Text to describe the purpose of the DB Alias or provide other information (up to 128 characters).

Tabs

The DB Alias Editor includes the **General**, **Connection**, **Server**, **Table Defaults**, **Index Defaults**, **Synonym Defaults** tabs (for Oracle and Informix), **Alias Defaults** tab (for DB2 CS, DB2 UDB, and DB2 MVS™), and **Trigger Defaults** tab.

General Tab

The **General** tab displays information about the DBMS, including identifier case, character set values, delimiters, and separators.



DBMS Type and Version

The type and version of the database management system for the DB Alias. For example:

- DB2 Common Server and DB2 MVS
- Oracle
- Sybase
- Informix
- SQL Server

Identifier Case

The default case for alphabetic characters used to identify database objects (for example, names of tables or views):

Mixed Case
Accepts and references object identifiers exactly as entered (case-sensitive). For example, *TableName*, *tablename*, and *TABLENAME* are stored and must be referenced exactly as entered.

Upper Case
Translates object identifier entries or references to uppercase. For example, *TableName* or *tablename* translates to *TABLENAME*.

Lower Case
Translates object identifier entries or references to lowercase. For example, *TableName* or *TABLENAME* translates to *tablename*.

Preserve Case

Accepts object identifiers exactly as entered, but references objects without regard to case. For example, *TableName*, *tablename*, and *TABLENAME* are stored as entered, but are referenced in any case.

Note: This option is available only for database management systems that allow you to specify object identifier case (for example, Sybase ASE and SQL Server). If the database management system provides the case identifier option and you do not specify case, Optim uses mixed case.

Code Page

The character set that a DBMS interface uses to communicate with a client. Optim uses the Windows 95 or Windows NT code page to display or write data to a file. The DBMS interface uses a code page to pass data to Optim. In turn, Optim uses a code page to pass, insert, or update data through the DBMS interface.

In most cases, Optim identifies the code page and data received through the DBMS interface. As a result, any code page specified for the DB Alias is ignored. However, in some instances, the code page is not recognized (e.g., ODBC) and the DB Alias code page is used.

To enter the DB Alias code page, click the arrow to select from a list of valid character set values. Supported code pages are:

Code Page	Character Set
437	U.S. English
850	International Multilingual
860	Portuguese
863	Canadian, French
865	Nordic
1250	Eastern European
1251	Cyrillic
1252	U.S./Western European ANSI or Latin 1 (ISO 8859-1)
1253	Greek
1254	Turkish
1255	Hebrew
1256	Arabic
1257	Baltic Rim
blank	Windows 95 or Windows NT

Optim expects to use the following, depending on the DBMS:

DBMS	Code Page
DB2	DB2CODEPAGE environment variable. If not found, the Windows code page is used.
Oracle	Oracle Character Set Name (as defined in the Oracle Windows Registry entries). If not found, the DB Alias code page is used.
Sybase ASE	Sybase Locale Character Set Name (as defined to the Sybase ASE client). If not found, the DB Alias code page is used.
SQL Server	Windows code page.

DBMS	Code Page
Informix	Windows code page.

String Delimiter

The delimiter used to enclose strings that contain special characters or blanks, either single quotes (‘ _ ’) or double quotes (“ _ ”).

Note: This option is available only for DB2 Common Server and DB2 MVS.

Decimal Separator

The separator character needed to designate the decimal point in numeric strings. Select the character that is appropriate for the DBMS:

Separator	Example
Comma (,)	1,50
Period (.)	1.50
Space (_)	1_50

Note: If you do not specify a separator, a period is used. For Oracle, the value specified in the DBMS parameters is used.

Optim Directory DB with a Shared Connection

Select this option if the Optim Directory resides in the database indicated by the DB Alias. The Optim Directory, created when Optim is installed, holds internal system tables and any objects you create using Optim. Several databases can use the same Optim Directory. Access Definitions and other objects stored in the Directory can be shared.

Note: The Optim Directory can reside in only one database.

Always Require a Password

Indicates whether Optim is configured to require a password for the DB Alias. You can change this setting using the Configuration program. For details, see the *Installation and Configuration Guide* .

Data is Kept in Unicode Format

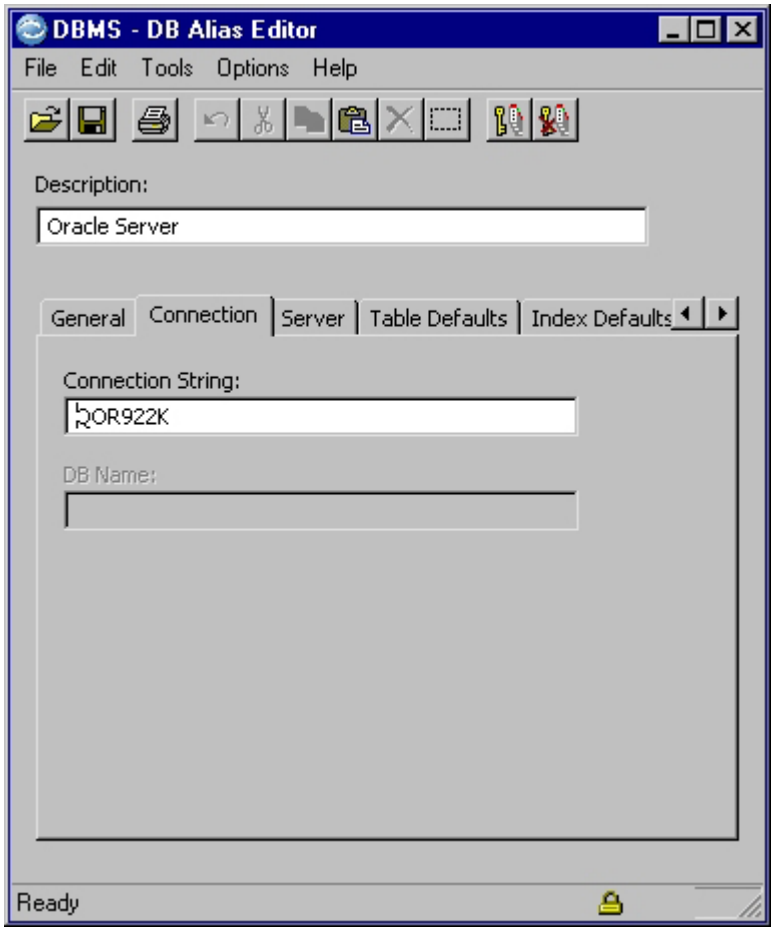
Indicates whether DB Alias data is kept in Unicode format. **Data is Kept in Unicode Format** is displayed only if you are connected to an Optim Directory in a DBMS for which Unicode is supported. You can change this setting using the Configuration program.

Data is Kept in Multi-byte Format

Indicates whether DB Alias data is kept in multi-byte format. **Data is Kept in Multi-byte Format** is displayed only if you are connected to an Optim Directory in a DBMS for which multi-byte is supported. You can change this setting using the Configuration program.

Connection Tab

The **Connection** tab displays the name of the database associated with the DB Alias and a connection string that enables Optim to access it.



Connection String

The value used by Optim to connect to the database associated with the DB Alias. This entry varies according to the database management system:

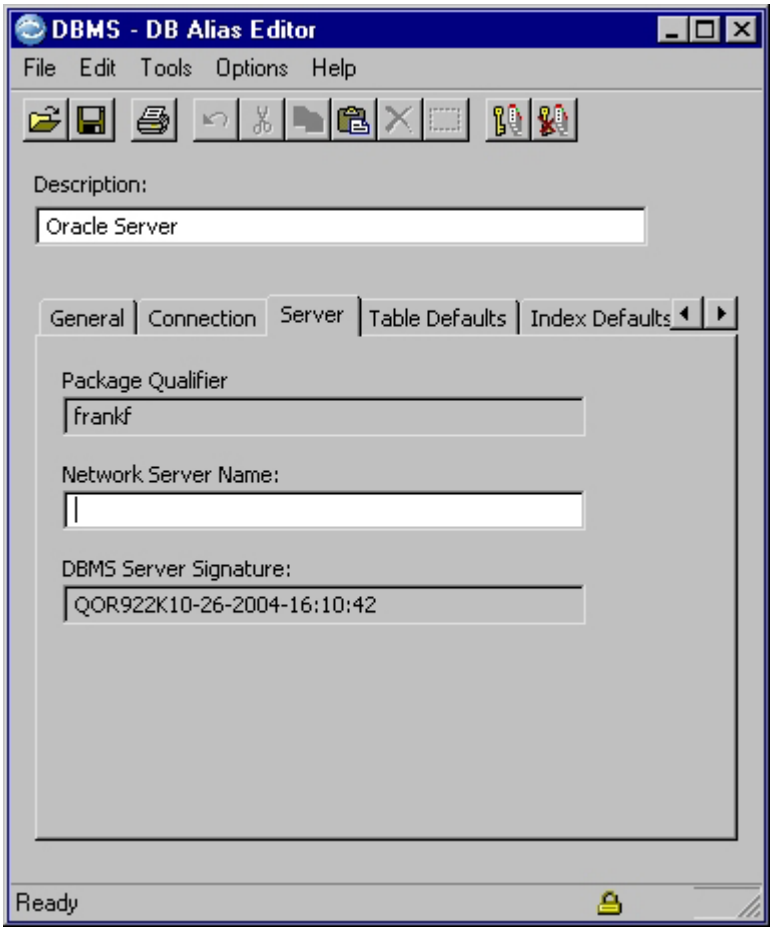
DBMS	Connection String Value
DB2	Database Name or Identifier
Oracle	Host Machine Identifier
Sybase ASE	Server Name
SQL Server	Server Name
Informix	Server Name

DB Name

The DB Name (1 to 64 characters) identifies the database associated with the DB Alias. This parameter applies to database management systems that support multiple database instances within a single database server (for example, Sybase ASE and SQL Server).

Server Tab

The **Server** tab displays the DBMS qualifier, network server name, and DBMS server signature that enables Optim to access the database.



DBMS Qualifier

The qualifier for the package, plan, or procedure Optim uses to access the database identified by this DB Alias. Each database management system uses a different term (qualifier) to identify this procedure:

DBMS	Qualifier
DB2	Collection Name
Oracle	Package Qualifier
Sybase	Procedure Qualifier
SQL Server	Procedure Qualifier
Informix	Procedure Qualifier

Network Server Name

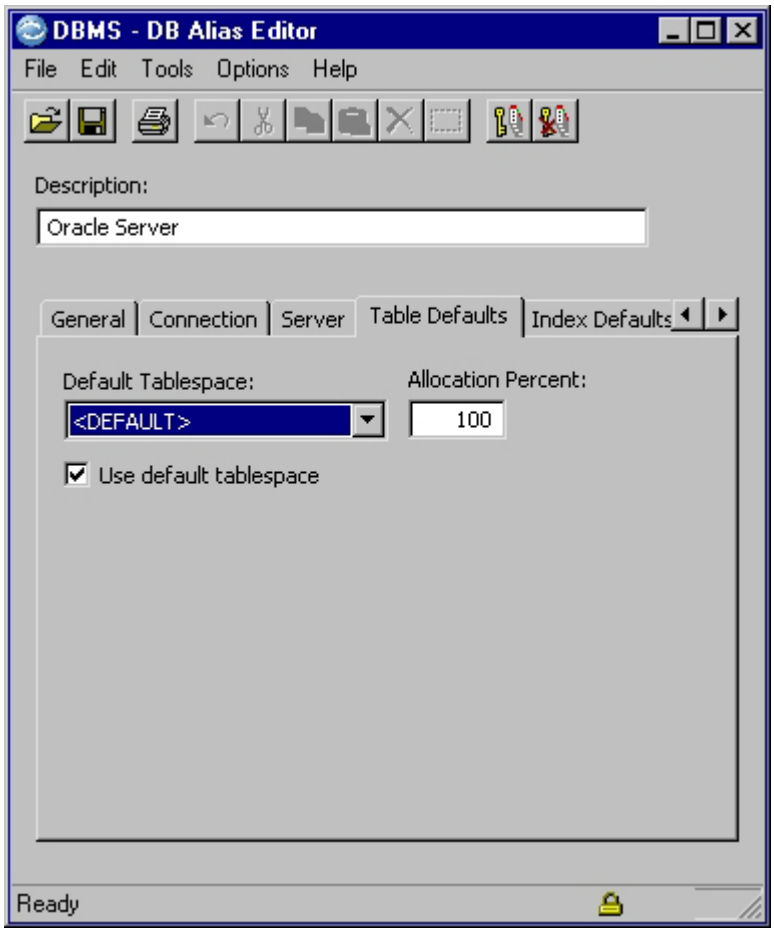
The network server name identifies the server where the database identified by the Connection String resides (see “General Tab” on page 191). The name is assigned to a network server and must be used for all DB Aliases associated with databases residing on that network server. You can specify a Network Server Name only for database management systems that allow multiple databases on a single server (for example, Sybase ASE).

DBMS Server Signature

When you install a DBMS on a specific server, the installation process records a server signature to identify the DBMS and indicate the date and time the DBMS was installed.

Table Defaults

The **Table Defaults** tab indicates the default tablespace associated with a DB Alias. The default tablespace varies according to the database management system.



Default Database

A default database is required only for DB2 MVS. Enter the default database appropriate for the DB Alias.

Default Tablespace

Enter the default tablespace, Dbspace, or segment appropriate for the DB Alias. Click the arrow to select from a list, or select "<DEFAULT>".

Use default tablespace

Select the check box to use the default parameters in all cases, or clear the check box to use source information from the Extract or Archive File.

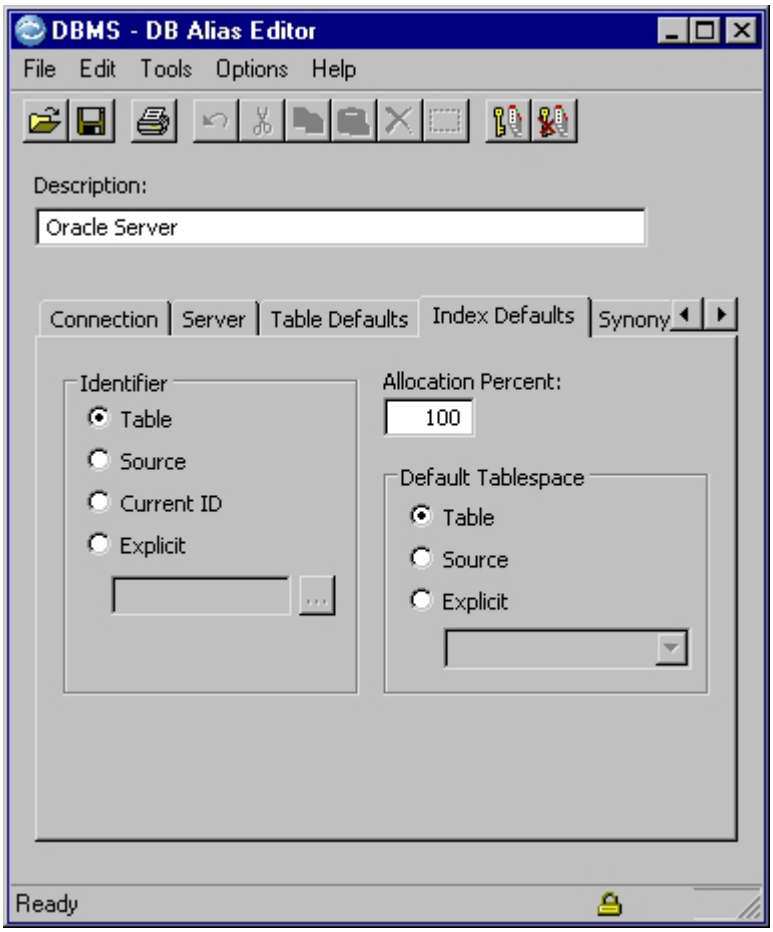
Allocation Percent

Allocation Percent allows the adjustment of storage-related parameters on Create Table and Alter statements for tables. The allowable range is 0 to 999. The default value is 100. Allocation Percent is required only for Oracle databases.

- By default, the target SQL is created using the values for the source objects in the Extract or Archive File.
- If you specify zero (0), the storage-related clause in the SQL statement is omitted.
- Any value other than zero results in a percentage of the source value being used in the target clause.

Index Defaults

The **Index Defaults** tab displays the index defaults associated with a DB Alias. The default index varies depending on the database management system for which the DB Alias is defined.



Identifier

Select default options used to specify the identifier for new indexes. You can create a new index based on the following:

Table Identifier from the target table.

Source
Identifier from the source index.

Current ID
Current database user ID from a destination.

Explicit
New identifier. If you select this option, you must specify an identifier.

Allocation Percent

Allocation Percent, required for Oracle and DB2 MVS only, allows the adjustment of storage-related parameters on Create and Alter statements for indexes. The allowable range is 0 to 999. The default value is 100.

- By default, the target SQL is created using the values for the source objects in the Extract File.
- If you specify zero (0), the storage-related clause in the SQL statement is omitted.
- Using any value other than zero results in a percentage of the source value being used in the target clause.

Default Tablespace

Default Tablespace (segment, filegroup, or Dbspace) is required for Oracle, Sybase ASE, Informix and SQL Server. Select default options used to specify the tablespace for new indexes. You can create a new index based on the following:

Table Same tablespace as the owning table.

Source
Same tablespace as the index in the Extract File (if possible).

Explicit
Select the tablespace from the list of available tablespaces for the database. Select <DEFAULT> to eliminate any tablespace from the Create Index SQL statement and cause the index to be created in the default tablespace.

Buffer Pool

The buffer pool (e.g., BP1) that is to be used when creating an Index. You can enter a specific value for the buffer pool or select a value from the list. A Buffer Pool value is required for DB2 MVS only.

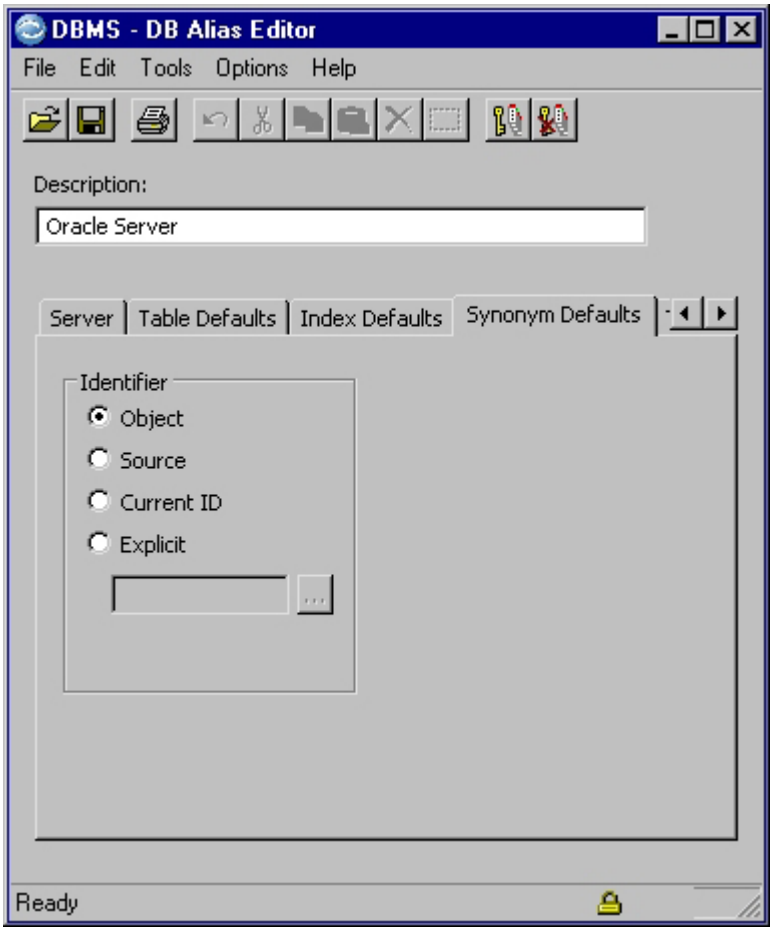
The list displays any index buffer pools previously specified for this DB Alias, as well as **<DEFAULT>** and **<SOURCE>**.

<DEFAULT>
Select to use the default buffer pool specified by DB2 MVS. When creating an index, Optim does not generate a BUFFERPOOL clause in the Create statement.

<SOURCE>
Select to use the same buffer pool as the index of the source Archive or Extract File.

Synonym Defaults

The **Synonym Defaults** tab displays the default settings for synonyms associated with the DB Alias. Synonyms apply to Oracle and Informix databases.



Select default options used to provide the identifier for new synonyms. You can create a new synonym based on the following:

Object
Identifier from a corresponding target object. For synonyms, the corresponding target object is the table, synonym, function, package, package body, procedure, sequence, trigger, or view referenced in the synonym.

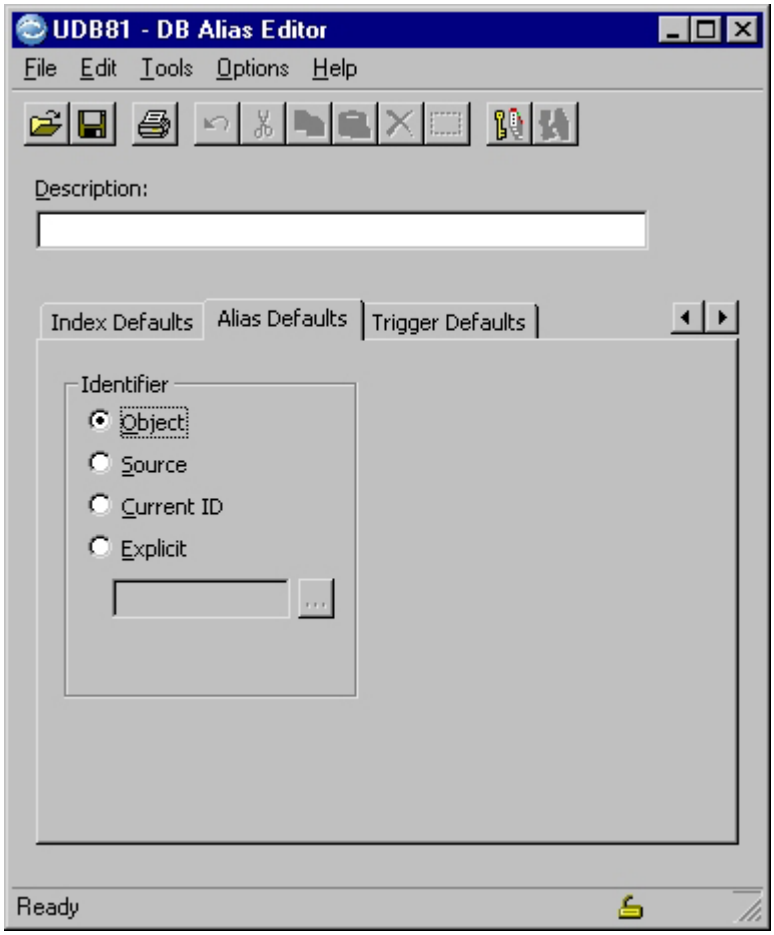
Source
Identifier from the source synonym.

Current ID
Current database user ID from the destination.

Explicit
New identifier. If you select this option, you must specify an identifier.

Alias Defaults

The **Alias Defaults** tab displays default options for creating new aliases. You can specify a default alias if you are using DB2 CS, DB2 UDB, or DB2 MVS.



Select default options used to provide the identifier for new aliases. You can create a new alias based on the following:

Object
Identifier from a corresponding target object. For aliases, the corresponding target object is the table, view or alias referenced in the alias.

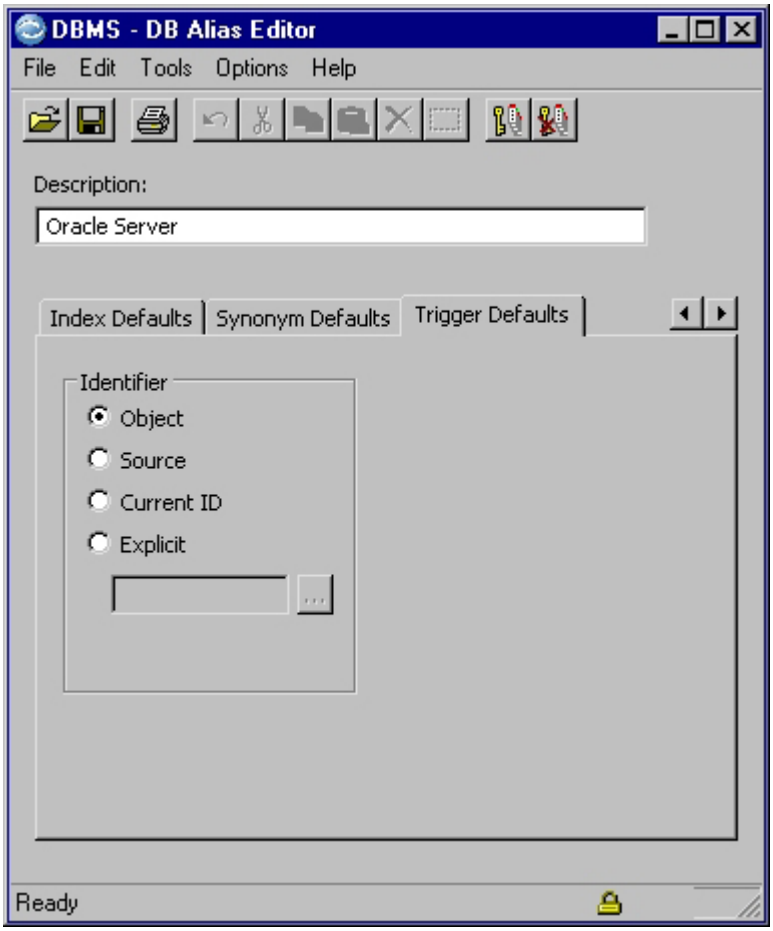
Source
Identifier from the source alias.

Current ID
Current database user ID from the destination.

Explicit
New identifier. If you select this option, you must specify an identifier.

Trigger Defaults

The **Trigger Defaults** tab displays the default settings for triggers in a database associated with a DB Alias.



Select default options used to provide the identifier for new triggers. You can create a new trigger based on the following:

Object
Identifier from a corresponding target object. For triggers, the corresponding target object is the table referenced in the trigger.

Source
Identifier from the source trigger.

Current ID
Current database user ID from the destination.

Explicit
New identifier. If you select this option, you must specify an identifier.

Chapter 8. Primary Keys

A primary key is the column or columns that contain values that uniquely identify each row in a table. A database table must have a primary key for Optim to insert, update, restore, or delete data from a database table. Optim uses primary keys that are defined to the database. However, you can also define primary keys to supplement those in the database.

A primary key is needed:

- In any table that is visited more than once in a process, for example, a child table that has two or more parent tables referenced in the Access Definition.
- To enable the Point and Shoot feature for a Start Table. For details, see "Edit Point and Shoot List" on page 109.

Note: If a primary key is not defined and is required to perform a specific task, an error message appears. Use the Primary Key Editor to create the necessary primary key.

Types of Optim Primary Keys

You can define two types of primary keys that are stored in the Optim Directory:

- An **Explicit** primary key applies to a single table.
- A **Generic** primary key applies to any tables that have the same base name, column names, and attribute specifications, but different Creator IDs.

There is no difference in function or appearance between generic and explicit primary keys. However, if a table has keys of both types, the explicit primary key is used.

Naming Conventions

The fully qualified name of a primary key is the same as the fully qualified name of the database table for which it is defined. This name consists of: *dbalias.creatorid.tablename*.

dbalias Alias that identifies the database where the table resides (1 to 12 characters).

creatorid
Creator ID assigned to the table (1 to 64 characters).

tablename
Base table name (1 to 64 characters).

Note:

- The combined total length of columns for a primary key is limited to 3584 bytes.
- When Object Security is enabled, the second and third parts of a primary key (i.e., the *creatorid* and *tablename*) are restricted to a combined total of 64 characters.
- When Object Security is not enabled, each of those parts may consist of up to 64 characters. See the *Installation and Configuration Guide* for detailed information about Object Security.

Contents

This section explains how to create and maintain explicit and generic primary keys defined to the Optim Directory, including how to:

- Create and edit an explicit or generic primary key.
- Create a primary key from a unique index.

- Convert a primary key to a generic primary key.

Open the Primary Key Editor

Use the Primary Key Editor to create and maintain explicit and generic primary keys stored in the Optim Directory and to browse primary keys defined to a database. There are different ways to open the editor, depending on whether you want to create a new primary key or select a primary key to browse or edit.

Create a Primary Key

You can create a primary key from the main window or from the Primary Key Editor.

From the Main Window

To create a primary key:

1. In the main window, select **New** from the **File** menu.
2. Select **Primary Key** from the **Definitions** submenu to open the Select a Table for the Primary Key Definition dialog.
3. Double-click the desired DB Alias.
4. ***Optional*** – Specify a **Pattern** to limit the list of tables based on your criteria and click **Refresh**.
5. Double-click the desired database table to open the Primary Key Editor.
6. Select columns to include in the primary key by clicking on a column name in the **Available Columns** box and dragging it to the **Key Columns** box.
7. Use the drag and drop feature to arrange Key Columns in priority order.
8. ***Optional*** – Type a brief description of the primary key.
9. Select **Save** from the **File** menu.

From the Primary Key Editor

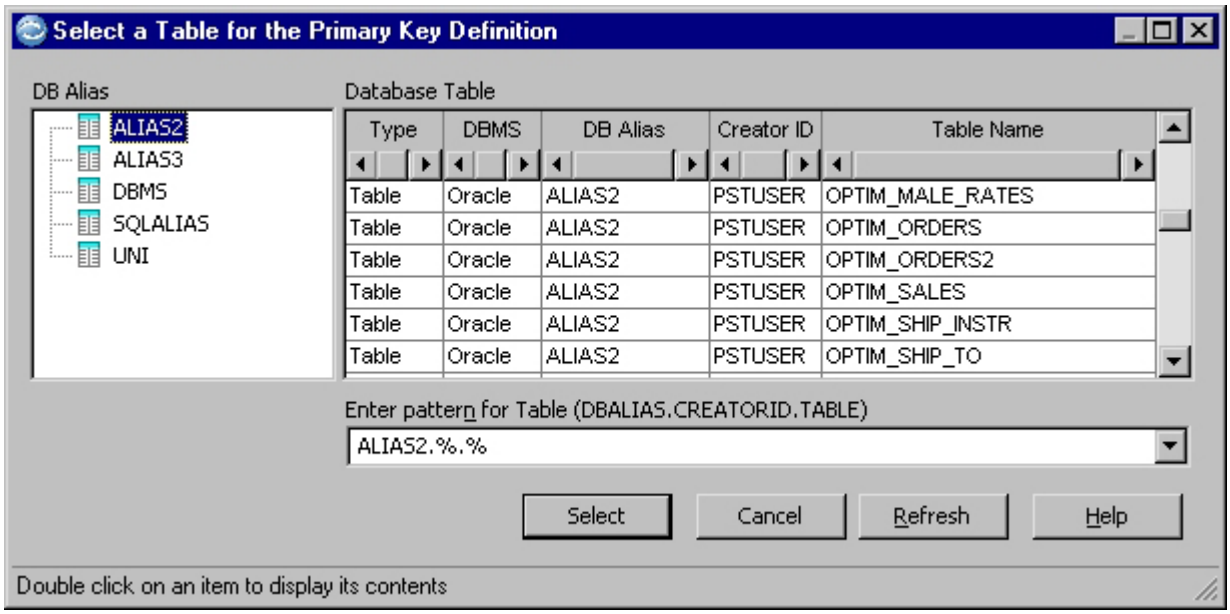
To create a primary key:

1. In the main window, select **Primary Key** from the **Definitions** menu to open the Primary Key Editor and last edited primary key.
2. In the Primary Key Editor, select **New** from the **File** menu to open the Select a Table for the Primary Key Definition dialog.
3. Double-click the desired DB Alias.
4. ***Optional*** – Specify a **Pattern** to limit the list of tables based on your criteria and click **Refresh**.
5. Double-click the desired database table to open the Primary Key Editor.
6. Select columns to include in the primary key by clicking on a column name in the **Available Columns** box and dragging it to the **Key Columns** box.
7. Use the drag and drop feature to arrange Key Columns in priority order.
8. ***Optional*** – Type a brief description of the primary key.
9. Select **Save** from the **File** menu.

These steps are the minimum required to create a primary key. Because the options to create and modify a primary key are similar, see "Using the Editor" on page 207 for details.

Select a Table for the Primary Key Definition Dialog

The Select a Table for the Primary Key Definition dialog provides a list of database tables for each DB Alias:



- The DB Aliases that correspond to the available databases are listed on the left. The selected DB Alias is shaded. To change the selection, double-click a different DB Alias or overtype the DB Alias in the **Pattern** box. (If you change this selection in either place, it automatically changes in the other.)
- The type of object (table, view, alias, synonym) and fully qualified table names appear in the right list box. The list is sorted in alphabetical order by Creator ID and table name.

Qualifiers

Identify a database table, view, alias, or synonym by the following qualifiers:

Type Type of object: table, view, alias, or synonym.

DBMS Type
Type of DBMS to which the selected table is defined.

DB Alias
Alias name to identify the database where the table resides.

Creator ID
Creator ID assigned to the table.

Note: The DB Alias and the Creator ID qualify the table name.

Table Name
Indicates the name assigned to a table.

Pattern

Use a **Pattern** to limit the list of database tables in the Select a Table for the Primary Key Definition dialog. After you specify a pattern, click **Refresh** to redisplay the list based on your criteria. See "Use a Pattern" on page 27 for more information.

Primary Keys

After you select a table, the database is checked to determine whether a primary key is defined for that table and attempts to populate the Primary Key Editor with meaningful information:

- If the database contains a primary key for the selected table, the details are shown in the Primary Key Editor. You cannot modify or delete a primary key defined in the database. However, you can display the primary key information in the Primary Key Editor.
- If the database does not contain a primary key for the selected table, the Optim Directory is checked. If an Optim primary key exists, the details are shown in the Primary Key Editor. You can browse, modify, or delete a primary key.
- If neither the database nor the Optim Directory contains a primary key for the selected table, the table and its columns are shown in the Primary Key Editor. You can create a primary key.

Note: You can create and modify primary keys defined in the Optim Directory. However, if you have authority, you can use the Create command available from the **Utilities** menu to create or drop primary keys defined in the database. For more information about the Create Utility, see Chapter 17, "Create," on page 411.

Select a Primary Key to Edit

You can select any primary key stored in the Optim Directory for editing from the main window or from the Primary Key Editor.

From the Main Window

To select a primary key:

1. In the main window, select **Open** from the **File** menu to display the Open (object selection) dialog. (See "Using the Open Dialog" on page 25 for more information on the Open dialog.)
2. Double-click **Primary Key** to expand the identifier list.
3. Double-click the desired DB Alias to display a list of primary keys.
4. ***Optional*** – Specify a **Pattern** to limit the list based on your criteria and click **Refresh**.
5. Double-click to select the desired explicit or generic primary key and open the Primary Key Editor.

Note: To select the last primary key you edited, select **Primary Key** from the **Definitions** menu in the main window to open the Primary Key Editor and the last edited Primary Key.

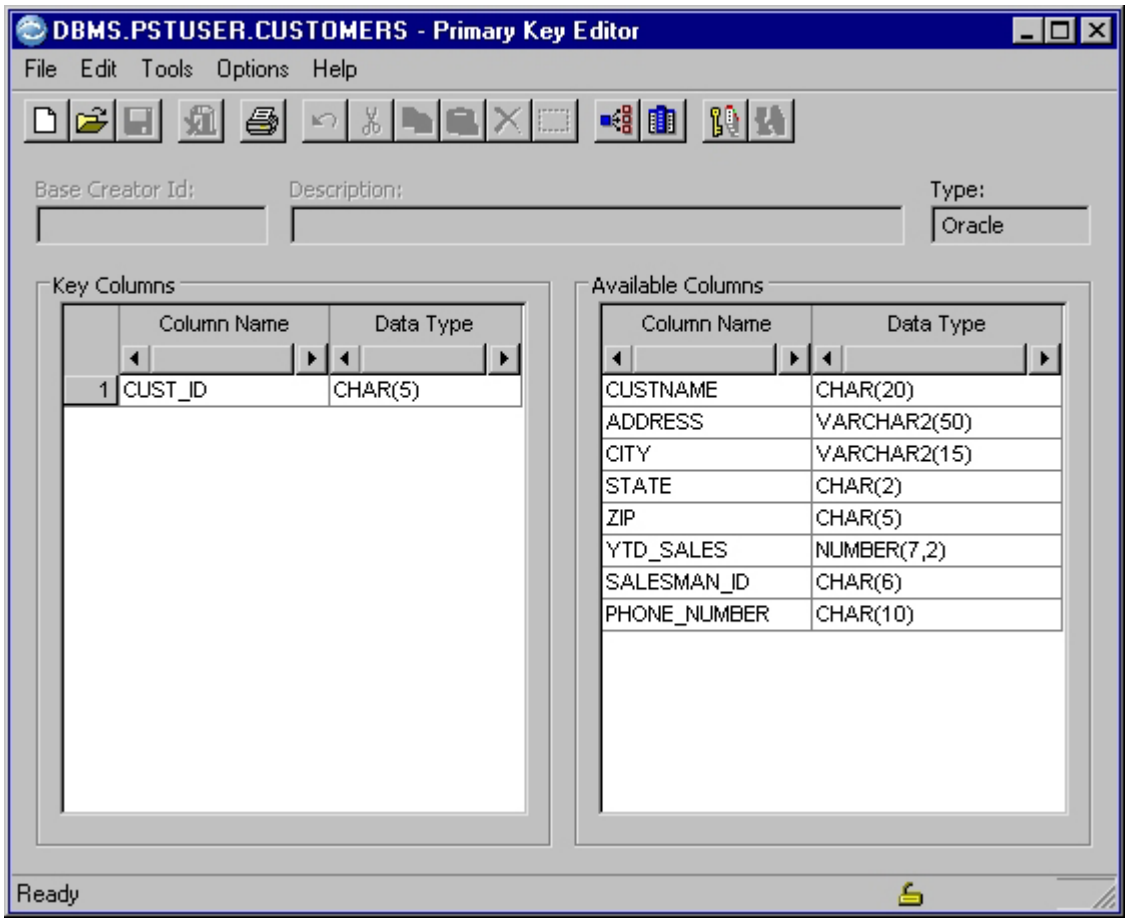
From the Primary Key Editor

To select a different primary key:

1. In the Primary Key Editor, select **Open** from the **File** menu to display the Open a Primary Key dialog. (See "Using the Open Dialog" on page 25 for more information on the Open dialog.)
2. Double-click the desired DB Alias to display a list of primary keys.
3. ***Optional*** – Specify a **Pattern** to limit the list based on your criteria and click **Refresh**.
4. ***Optional*** – Select a **Show pattern for** option to display a specific type of primary key.
5. Double-click the desired primary key to open the Primary Key Editor.

Using the Editor

After you select a table, Optim checks the database and the Optim Directory to determine if a primary key is defined for the selected table and lists columns in the Primary Key Editor:



Base Creator ID

If a generic primary key applies to the table, the Creator ID of the base table is displayed. If you select an explicit primary key, the box is blank and protected.

Description

Description of the content or purpose of the Optim primary key (up to 40 characters). You can modify this description for primary keys.

Type

Type of primary key: Optim (explicit), Generic, or specific DBMS (for example, Oracle).

Key Columns

List of columns in the primary key, in priority order.

Column Name
Name of each column in the primary key.

Data Type
Attributes of each column in the primary key.

To create or modify a primary key, drag column names between the **Available Columns** box and the **Key Columns** box. You can also drag column names to arrange the Key Columns in priority order. You cannot create a Primary Key using an SQL Variant column.

To remove columns from the **Key Columns** box, right-click to open a shortcut menu and select **Remove** or **Remove All**. You can also drag column names from the **Key Columns** box to the **Available Columns** box.

Available Columns

Columns that are available to be included in the primary key. Columns currently included in the primary key are not listed. Initially, the columns are listed in the order that they are defined in the database.

Column Name
Names of the available columns in the table.

Data Type
Attributes of each available column.

Note: Large Object (LOB) and SQL variant columns are not available for use as primary key columns.

Menu Commands

In addition to the standard **File**, **Edit**, and **Tools** menu commands, you can select the following commands from the **Tools** menu in the Primary Key Editor:

Generic
Convert a primary key to a generic primary key.

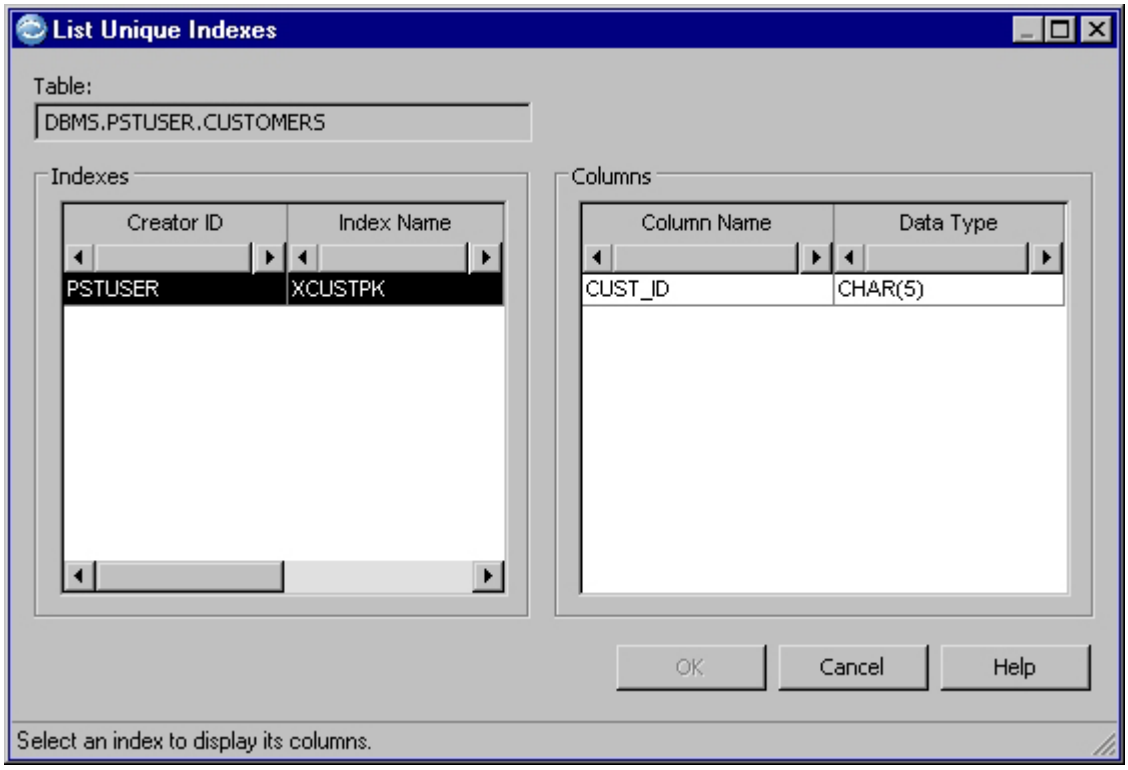
Indexes
Open the List Unique Indexes dialog to create or modify a primary key using a unique index.

Modify Base Table
Open the Respecify Base Table Creator ID dialog to change the Creator ID for the base table associated with the generic primary key.

Note: This command is enabled for generic primary keys. Select this command if the base table used to create the generic primary key is no longer identical to the other tables that use the generic primary key.

Define a Primary Key Based on a Unique Index

Database tables often have one or more unique indexes defined for various purposes. It may be useful to define a primary key based on an existing index. To view the contents of a unique index in the Primary Key Editor, select **Indexes** from the **Tools** menu.



Table

Name of the table for which the primary key is being defined.

Indexes

List of unique indexes for the table. To display the associated columns, select an index.

Creator ID
Creator ID of the table associated with the selected index.

Index Name
Name of the selected index.

To create or modify a primary key based on a unique index, select an index and select **OK**. The columns of the unique index replace all previously specified columns in the **Key Columns** box of the Primary Key Editor.

Columns

List of columns included in a selected unique index.

Column Name
Names of the columns that define the index.

Data Type
Attributes of each column in the index.

Manage Generic Primary Keys

Some databases contain sets of tables that are identical except for the Creator ID. Rather than define an explicit primary key for each table, define a generic primary key for all tables that have the same base name, regardless of the Creator ID.

You can:

- Create a new generic primary key in the same way you create a new explicit primary key and select **Generic** from the **Tools** menu in the Primary Key Editor before saving.
- Modify a generic primary key in the same way you modify an explicit primary key. For details, see "Using the Editor" on page 207.
- Convert a DBMS or Optim primary key to a generic primary key. (However, you cannot convert a generic primary key to an explicit primary key.)
- Change the Creator ID to identify the base table associated with a generic primary key. Use this option when the base table associated with a generic primary key changes.

Convert to a Generic Primary Key

In the Primary Key Editor, select **Generic** from the **Tools** menu to convert a database primary key or explicit Optim primary key to a generic primary key.

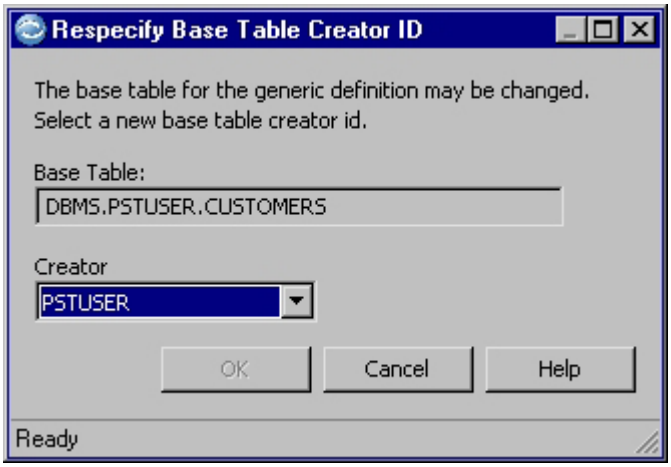
A primary key defined to the database is immediately converted to generic. If you select an explicit Optim primary key, you are prompted to retain or delete the original primary key. If you choose to retain the original, changes to the generic primary key have no effect on the original explicit primary key.

When you convert a primary key to generic, the Base Creator ID in the Primary Key Editor displays the Creator ID of the explicit primary key. The notation, *Untitled* is displayed in the title bar until you save the generic primary key. After you save a generic primary key, the Creator ID portion of the primary key name is replaced with an asterisk (*), for example, *dbalias.*.tablename*.

Respecify the Base Table Creator ID

You can use generic primary keys for any tables that are identical except for the Creator ID. If the base table associated with the generic primary key changes, you can specify a different Creator ID.

To display the Respecify Base Table Creator ID dialog, select **Modify Base Table** from the **Tools** menu in the Primary Key Editor.



Base Table

Name of the base table used to create the generic primary key.

Creator ID

Creator ID associated with the base table. To change the Creator ID, click the down arrow to select from a list, and select **OK**.

Your selection is displayed as the Base Creator ID in the Primary Key Editor. You may then save the generic primary key.

Chapter 9. Relationships

A relationship is a defined connection between the rows of two tables. This connection is generally determined by values in selected columns from a parent table that correspond to values in the child table. Optim uses relationships to determine the data to be retrieved from related tables and relies upon relationships defined to the database, when available. However, you can also define relationships to supplement those in the database.

The relationships you define are stored in the Optim Directory. Although you can define Optim relationships that conform to database management system requirements, much greater flexibility is possible. For example, some database management systems have the following requirements for relationships:

- The parent table must have a primary key that is related to the foreign key in the child table.
- Corresponding columns must have identical data types and attributes.

When defining Optim relationships, a number of the database restrictions are relaxed. For example:

- Primary keys and foreign keys are not required.
- Corresponding columns need not be identical, but must be compatible.
- At least one of a pair of corresponding columns must be specified by column name. However, you can use an expression to evaluate or define the value in the second column. Expressions can include string literals, numeric constants, NULL, concatenation, and substrings.

The more flexible Optim relationships are called “extended” relationships. Extended relationships can replicate implicit or application-managed relationships in your database, allowing you to manipulate sets of relational data in the same manner as in your production environment.

In addition, an Optim relationship can be stored in the Optim Directory as:

- An **explicit** relationship, used for a single pair of tables.
- A **generic** relationship, used for one or more pairs of tables that have the same base name, column names, and attributes, but different Creator IDs.

Generic relationships are useful when several sets of tables differ only by Creator ID. (For example, in a test environment, each programmer may use a separate copy of the same production tables. Each set of tables can be distinguished by the Creator ID.) Using generic relationships, you define one set of relationships that applies to all sets of tables. Also, when a set of these tables is added, the generic relationships automatically apply.

Primary Keys

Although primary keys are not required to define relationships, they do make it easier. When you define a relationship using a parent table that has a primary key, the names of primary key columns are automatically inserted into the Relationship Editor, which also displays the names of matching columns in the child table. You can then edit the column expressions, as needed.

If a parent table does not have a primary key, a blank line is inserted into the Relationship Editor for you to enter column names for the parent and child tables, as required.

Naming Conventions

The name of a relationship is: *dbalias.creatorid.tablename.constraint*. This name consists of the fully qualified name of the child table suffixed with the constraint assigned to the relationship. It is helpful to use a consistent convention for naming relationships.

dbalias Alias that identifies the database where the child table resides (1 to 12 characters).

creatorid
Creator ID assigned to the child table (1 to 64 characters).

tablename
Base name of the child table (1 to 64 characters).

constraint
Name of the relationship (1 to 64 characters).

Note:

- When Object Security is enabled, the second, third, and fourth parts of a relationship (i.e., the *creatorid*, *tablename*, and *constraint*) are restricted to a combined total of 64 characters. When Object Security is not enabled, each of those parts may consist of up to 64 characters. See the *Installation and Configuration Guide* for detailed information about Object Security.
- If **Enforce DBMS Rel. Name Lengths** is selected on the Product Options dialog **Database** tab, the *constraint* length must comply with the database management system's limit. See the *Installation and Configuration Guide* for detailed information on the Product Options dialog.

Section Contents

This section explains how to create and maintain explicit and generic relationships defined to the Optim Directory. You can include expressions in a relationship to define column data. You may also convert a database relationship or an explicit relationship to a generic relationship.

Open the Relationship Editor

Use the Relationship Editor to create and maintain relationship definitions stored in the Optim Directory and to browse relationships defined to a database.

Create a Relationship

You can create a relationship from the main window or the Relationship Editor.

From the Main Window

To create a relationship:

1. In the main window, select **New** from the **File** menu.
2. Select **Relationship** from the **Definitions** submenu to open the Select Relationship's Parent Table dialog.
3. ***Optional*** – Specify a **Pattern** to limit the list of tables based on your criteria and click **Refresh**.
4. Double-click the desired parent table name to open the Select Relationship's Child Table dialog.
5. Double-click the desired child table name to open the Relationship Editor.
6. To add columns to the Relationship List, select **Parent Columns** (or **Child Columns**) from the **Tools** menu and drag the column name to the appropriate grid cell in the Relationship Editor.
7. Modify the parent or child column expressions, as needed.
8. Select **Save** from the **File** menu to open the Save the Relationship Definition dialog.
9. In the **Constraint** box, specify a unique relationship name and then click **Save**.

From the Relationship Editor

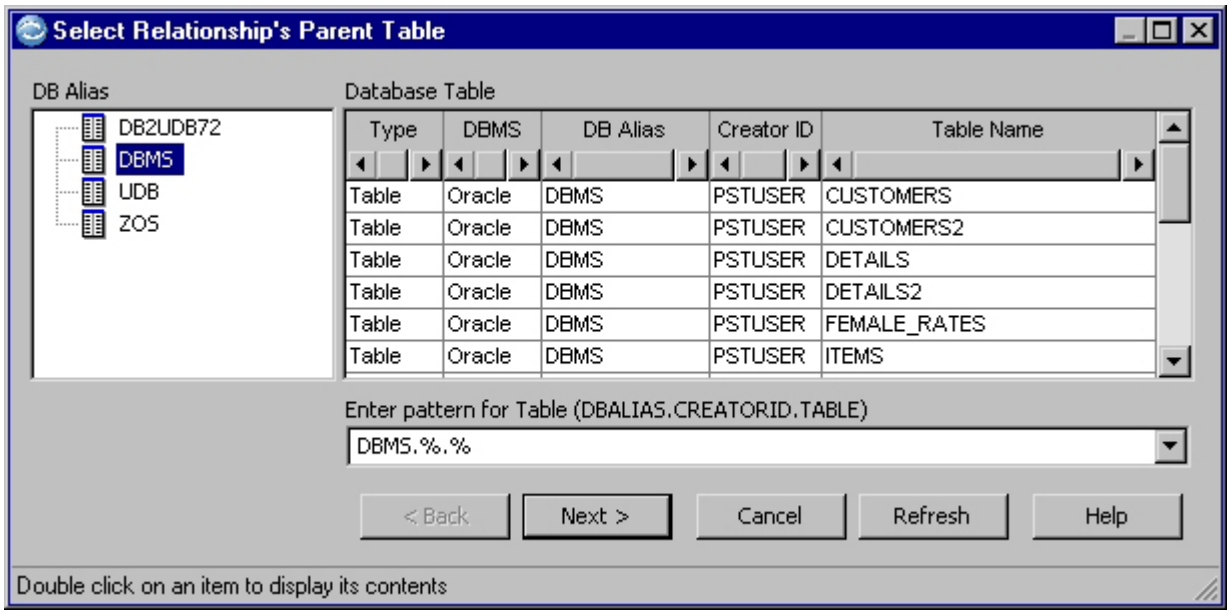
To create a relationship:

1. In the main window, select **Relationship** from the **Definitions** menu to open the Relationship Editor and last edited relationship.
2. Your next step depends on your purpose:
   - To create a new relationship, select **New** from the **File** menu in the Relationship Editor to open the Select a Parent Table dialog. Select a parent table to open the Select a Child Table dialog and then select a child table. The names of the selected tables appear in the Relationship Editor. You can edit the relationship, as needed.
   - To create a new relationship modeled on an existing one, open the desired relationship and select **Save As** from the **File** menu in the Relationship Editor.
   - To create and store a copy of the active relationship and continue editing, select **Save Copy As** from the **File** menu in the Relationship Editor.

These steps are the minimum required to create a relationship. Because similar methods are used to create and modify a relationship, refer to "Using the Editor" on page 217 for more information.

Select Relationship's Parent Table Dialog

To create a new relationship, you must select a parent table, then select a child table.



Note: The dialogs to select the parent and child tables are nearly identical. In this example, the Select Relationship's Parent Table dialog is used to illustrate the dialog.

- DB Aliases that correspond to the available databases are listed on the left. Double-click a DB Alias name or overtype the DB Alias in the **Pattern** box to display a list of objects in the corresponding database. (If you change this selection in either place, it changes in the other.)
- The list is in alphabetical order by Creator ID and table name.

Pattern

Use a **Pattern** to limit the list of database tables in the Select Relationship's Parent Table dialog. After you specify a pattern, click **Refresh** to redisplay the list based on your criteria. See "Use a Pattern" on page 27 for more information.

After you select the parent table and the child table for the relationship, the Relationship Editor is populated with as much meaningful relationship information as possible.

If the parent table has a primary key, the column names and data types of the primary key columns are inserted automatically in the Parent Expression grid column. Processing then continues as follows:

- If the data types are compatible, the name of a column in the child table that matches the name of a primary key column in the parent table is inserted in the Child Expression grid column.
- If no child table column names match, the child table is checked for columns with the attributes of a primary key column in the parent table. If a single child table column is found, its column name is inserted in the Child Expression grid column. If no columns or multiple columns are found, no column name is inserted.

If the parent table does not have a primary key, a blank line is inserted. You can select column names from a list or type the expressions directly.

Note: For information about compatible columns, see "Compatibility Rules for Relationships" on page 226.

Select a Relationship to Edit

You can select an Optim relationship for editing or browse a database relationship from the main window or the Relationship Editor. The most common method is explained first.

From the Main Window

To select a relationship:

1. In the main window, select **Open** from the **File** menu to display the Open (object selection) dialog. (See "Using the Open Dialog" on page 25 for more information on the Open dialog.)
2. Double-click **Relationship** to expand the identifier list.
3. Double-click the desired DB Alias to display a list of relationships.
4. ***Optional*** – Specify a **Pattern** to limit the list based on your criteria and click **Refresh**.
5. Double-click the desired explicit or generic relationship to open the Relationship Editor.

Note: To select the last relationship you edited, select **Relationship** from the **Definitions** menu in the main window to open the Relationship Editor and the last edited relationship.

From the Relationship Editor

To select a different relationship:

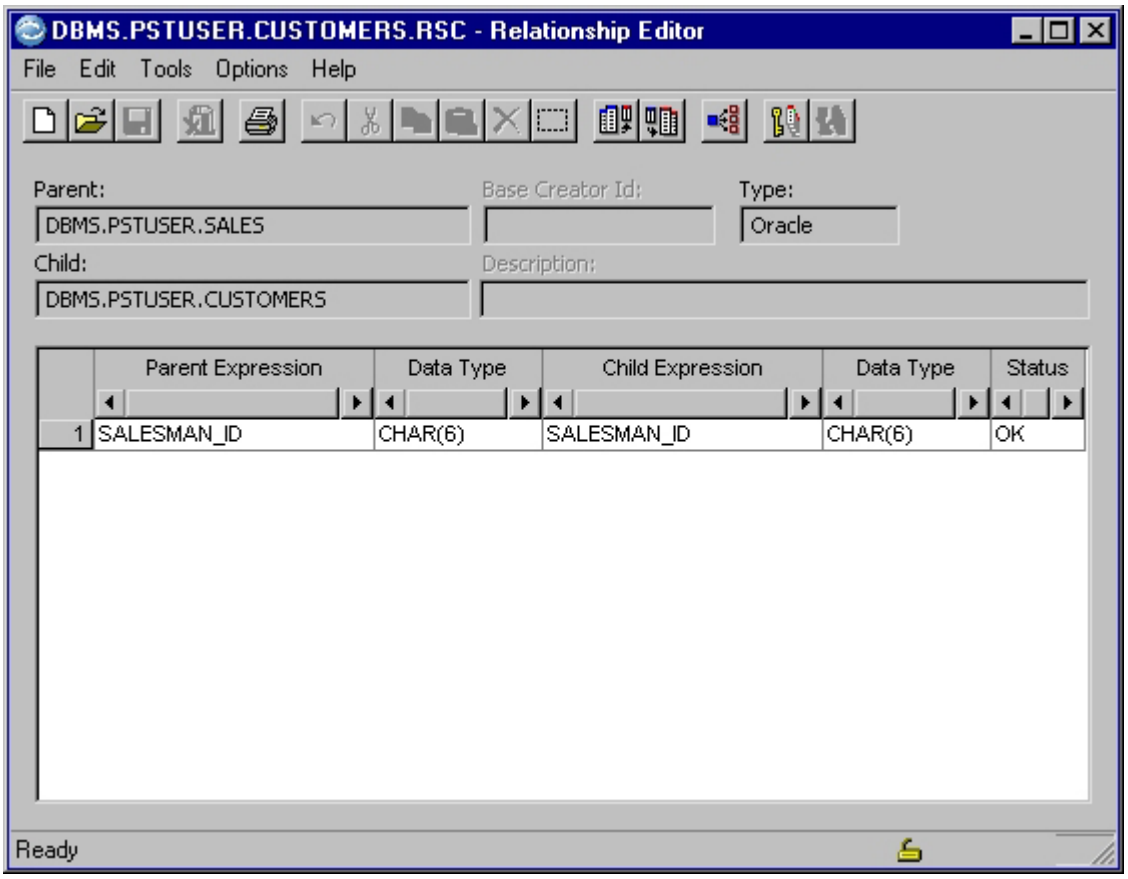
1. In the Relationship Editor, select **Open** from the **File** menu to display the Open a Relationship dialog. (See "Using the Open Dialog" on page 25 for more information on the Open dialog.)
2. Double-click the desired DB Alias to display a list of relationships.
3. ***Optional*** – Specify a **Pattern** to limit the list based on your criteria and click **Refresh**.
4. ***Optional*** – Select a **Show pattern for** option to display a specific type of relationship.
5. ***Optional*** – Select the **Relationships Related to Table** check box and enter a table name to retrieve a list of relationships that include the table. To limit the list further, select the **Show Only Directly Related Tables** check box.
6. Double-click the desired relationship to open the Relationship Editor.

Using the Editor

Use the Relationship Editor to create and maintain Optim relationships and browse database relationships.

You can edit an Optim relationship in any of the following ways:

- Select or specify the names of columns from the parent and child tables to define the relationship.
- Specify a column entry as a column name, NULL, constant, literal, substring, or concatenated expression.
- Arrange the column entries in priority order.



Parent

Fully qualified name of the parent table in the relationship. You cannot modify the name of the table.

Base Creator ID

Creator ID of the child table used to create the generic relationship. If the relationship is explicit, the Base Creator ID is blank and grayed.

Type

Type of relationship: Optim (explicit), Generic, or a specific DBMS, such as Oracle.

Child

Fully qualified name of the child table in the relationship. You cannot modify the name of the table.

Description

Text that describes or explains the purpose of the relationship (1 to 40 characters). You can modify this description for Optim and generic relationships.

Relationship List

The relationship list includes the pairs of parent and child table column entries that form the relationship. You cannot create a Relationship using an SQL Variant column.

Parent Expression
List of parent table entries that relate to corresponding entries for the child table. You may use column names automatically inserted into the Relationship Editor, or you can edit the entries. For details, see "Specify Column Values in a Relationship" on page 219.

Data Type
Data type for each column entry. The data types for any pair of parent and child columns may differ, but must be compatible. You cannot modify this value.

If the data types are invalid, a notation is displayed:

Required
Only one table in a pair has a column entry. You must provide a column entry for the other table.

Inconsistent
A substring uses a value that is not CHAR or an expression references a column that is not CHAR or VARCHAR. You must use character data in an expression that includes concatenated values or substrings.

Nonexistent
The named column does not exist in either the parent or child table. Verify the column name.

Not Supported
The data type or float constant is not supported. Numeric constants are valid only when the column is DECIMAL, INTEGER, or SMALLINT.

Too Large
A column entry exceeds 254 characters.

Too Many
The number of columns in the relationship exceeds 64.

Child Expression
A list of child table entries that relate to corresponding entries for the parent table. You may use column names automatically inserted into the Relationship Editor, or you can edit the entries. For details, see "Specify Column Values in a Relationship" on page 219.

Data Type
Data type for each column in the child table. The data types for any pair of parent and child column entries may differ, but they must be compatible. You cannot modify this value.

If the data types are invalid, a notation is displayed. Refer to the description of Data Type for the parent table for details.

Status The validity of the entries for the corresponding parent and child table columns. If valid, the status is **OK**. If one or both entries are invalid, the status is *ERROR* and a corresponding message explains the error.

Menu Commands

In addition to the standard **File**, **Edit**, and **Tools** menu commands, you can select the following commands from the **Tools** menu:

Parent Columns
Open the Parent Table Columns dialog. You can select columns from the parent table to define a relationship.

Child Columns
Open the Child Table Columns dialog. You can select columns from the child table to define a relationship.

Reverse Parent /Child Tables
Switch the names of tables from parent to child and child to parent when you define a new relationship.

Generic
Convert a database relationship or explicit Optim relationship to a generic Optim relationship.

Modify Base Tables
Open the Respecify Base Table Creator ID dialog. You can select a different Creator ID for the base table associated with a generic relationship.

You must respecify the Creator ID for the base table when the base table used to create the generic relationship is no longer identical to other tables that use the generic relationship.

Specify Column Values in a Relationship

The Relationship Editor lists the pairs of corresponding parent table and child table entries that make up the relationship.

Define or edit relationships by:

- Adding or replacing column names by selecting names from a list.
- Editing column entries. You can specify explicit column names (up to 75 characters) or a constant, a literal, a function, or expression.

Note: When you edit a relationship definition, the **Data Type** and **Status** values are inserted automatically.

Restrictions

Although the rules for creating Optim relationships are more flexible than those for creating database-defined relationships, there are some restrictions:

- You must reference at least one column for each table in the relationship.
- You can reference a maximum of 64 columns for any table in the relationship.
- You cannot match a literal or constant to a literal or constant.
- You cannot use a Large Object (LOB) or SQL variant column.
- The total length of all values specified in either the **Parent Table** or the **Child Table** cannot exceed 3584 bytes.
- You cannot create a Relationship using an SQL Variant column.

In a Relationship definition for a multi-byte or Unicode database:

- You cannot use the Substring Function.
- You cannot concatenate character data (CHAR or NCHAR) with binary (RAW).
- If Oracle character semantics are used for any CHAR column, all CHAR columns in the relationship must have character semantics or an NCHAR data type.

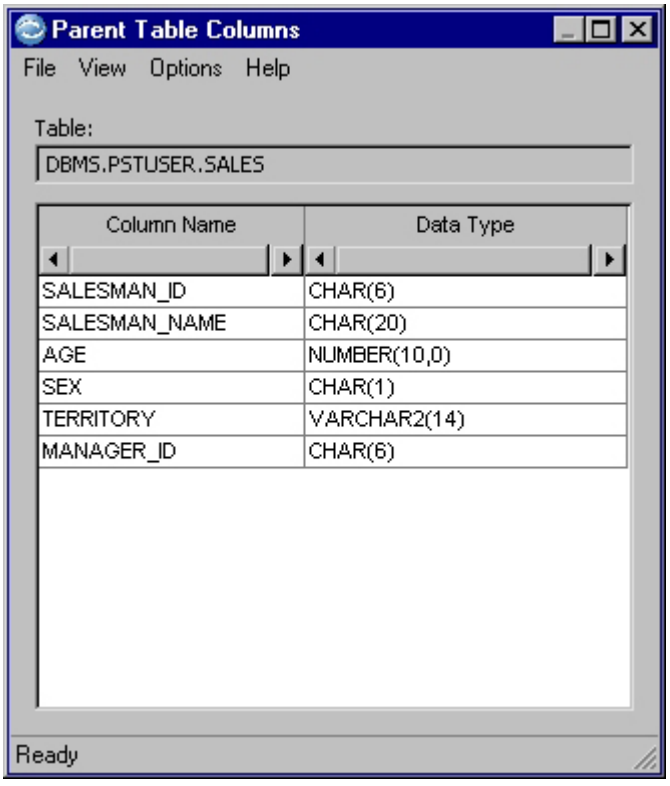
EXAMPLES:

CHAR	→	CHAR	Supported, semantics must match
NCHAR	→	NCHAR	Supported, semantics irrelevant.
CHAR	↛	NCHAR	Not Supported
CHAR	→	VARCHAR	Supported, semantics must match.
NCHAR	→	NVARCHAR	Supported, semantics irrelevant.
CHAR||NCHAR	→	NCHAR||CHAR	Supported, if character semantics; not supported if byte semantics.
CHAR||NCHAR	→	NCHAR||NCHAR	Supported, if character semantics; not supported if byte semantics.

Violating these restrictions causes an error. For further details about column compatibility, see "Compatibility Rules for Relationships" on page 226.

Select Column Names from a List

Select **Parent Columns** or **Child Columns** from the **Tools** menu to add or replace column names. You can switch between the Parent Table Columns dialog and Child Table Columns dialog by selecting the appropriate command from the **Tools** menu in the Relationship Editor.



Note: Because the dialogs to select parent and child columns are identical, the Parent Table Columns dialog is used to explain dialog details.

Table

Fully qualified name of the parent table from which you can select names of columns to include in the relationship.

Column Name

To select from the list of columns, drag column names from the dialog to the appropriate grid cell in the Relationship Editor. The inserted column name replaces any value in the current grid cell. The data types of the paired columns in the editor are evaluated and the Status is automatically updated.

Change the column display by selecting **View** menu commands in the Parent Table Columns dialog:

- To view a list of all columns in the table, select **All**.
- To view all unused columns, select **Unused**. If the table contains many columns, it may be helpful to display only unused columns.

Data Type

Data type for each column in the parent table.

Edit Parent and Child Columns

You can specify or modify a column name by typing over the name or using shortcut menu commands to **Insert**, **Remove**, or **Remove All** column names.

In addition, you can specify expressions, including substrings, concatenations, constants and literals for either the parent or child table column in a relationship. This flexibility is one of the most powerful features of Optim relationships. You can use any of the following values:

Column Name
An explicit column name in the parent or child table. (Column names are case-insensitive.)

NULL NULL.

Numeric Constant
A numeric constant. The value must conform to the data type, precision and scale defined for the column.

Boolean Constant
A Boolean constant as TRUE or FALSE.

String Literal
A string literal for a non-arithmetic value:

- You can specify a string literal when the corresponding column contains character data.
- You must enclose a string literal in single quotes.

You can define a string literal containing any characters, for example: 'CA' or '90210'.

Hexadecimal Literal
A hexadecimal literal: X'1234567890ABCDEF'
or 0X12 34567890ABCDEF

Substring
A Substring Function to use a portion of a source column value.

Concatenated Expression
A Concatenated Character or Binary expression to obtain a derived value.

Substring Function

Use the Substring Function to select part of a column value for a relationship. The Substring Function returns a substring of the contents of the named column. The Substring Function format is:

SUBSTR(*columnname, start,* [*length*])

columnname
Name of a character or binary column.

start The position of the first character in the string.

length The number of characters to use.

- If the locale uses a comma as the decimal separator, you must leave a space after each comma that separates numeric parameters (for example, after the comma between *start* and *length*.
- *start* and *length* are integers greater than or equal to 1.
- *start* plus *length* cannot exceed the total data length plus 1.
- *column-name* and *start* value are required. If you specify only one integer, it is used as the *start* value. The substring begins at *start* and includes the remainder of the column value.

Example

If the PHONE_NUMBER column is defined as CHAR(10), you can use the Substring Function to map the area code. To obtain a substring of the first three positions of the phone number (area code) for the destination column, specify:

SUBSTR(PHONE_NUMBER, 1, 3)

Concatenated Expressions

Concatenation allows you to combine column values or combine a column value with another value, using a concatenation operator (CONCAT, ||, or +). A concatenated expression can include character or binary values, but not both:

Character Values
Concatenated character values can be character columns, string literals, or substrings of character columns.

Binary Values
Concatenated character values can be binary columns, hexadecimal literals, or substrings of binary columns.

Note: A concatenated expression cannot include a zero-length string literal (' ') or a special register.

Example 1

You can define relationships using different data structures. For example, the data in two or more columns in one table may correspond to data in a single column in another table. You can define this type of relationship using concatenation or the Substring Function.

Assume that an address in the CUSTOMERS table is in two columns, ADDRESS1 and ADDRESS2, and in one column, ADDRESS, in the SHIP_TO table. You can define a relationship between the two tables by concatenating the columns in the CUSTOMERS table:

CUSTOMERS Table	SHIP_TO Table
ADDRESS1 || ADDRESS2	ADDRESS
ADDRESS1 CONCAT ADDRESS2	ADDRESS
ADDRESS1 + ADDRESS2	ADDRESS

Example 2

You can define the same relationship as in the prior example, using substrings:

CUSTOMERS Table	SHIP_TO Table
ADDRESS1	SUBSTR(ADDRESS,1,25)
ADDRESS2	SUBSTR(ADDRESS,26,25)

To compare the results of using a concatenation operator or the Substring Function, examine the generated SQL in each case.

SQL for a relationship using the concatenation operator:

```
SELECT * FROM TABLE2
 WHERE TABLE2.ADDRESS = 'composite value from both parent columns'
```

SQL for the relationship defined using the Substring Function:

```
SELECT * FROM TABLE2
 WHERE SUBSTR(TABLE2.ADDRESS, 1,25)='value from parent ADDRESS1
column'
 AND   SUBSTR(TABLE2.ADDRESS,26,25)='value from parent ADDRESS2
column'
```

Complex SQL is generated in the substring example; the DBMS must scan all rows in the table to achieve the desired result and may not use an index, even if one exists. In general, concatenation provides better performance.

Data-Driven Relationships

A data-driven relationship exists when a row in the parent table is related to rows in one of several child tables on the basis of the value in a particular column. You can define a data-driven relationship using:

- String literals or hexadecimal literals
- Numeric Constants and Boolean Constants
- NULL

For example, consider a pair of sample tables used to determine employee insurance rates for males and females. The EMPLOYEE table stores the identification, age, and gender details for each employee:

EMPLOYEE_ID	AGE	GENDER
058-44-2244	38	F
106-46-0620	40	M
248-91-2890	27	M

The FEMALE_RATES table contains insurance rates for women based on age; the MALE_RATES table contains insurance rates for men based on age:

FEMALE_RATES		MALE_RATES	
AGE	RATE	AGE	RATE
38	.25	38	.31
39	.33	39	.38
40	.43	40	.47

Two relationships are needed, one for rows in the EMPLOYEE table that contain data about female employees, and one for rows in the EMPLOYEE table that contain data about male employees.

Rows in the EMPLOYEE table are related to the FEMALE_RATES table when the value in the GENDER column matches the string literal 'F'. The AGE column in the EMPLOYEE table corresponds to the AGE column in the FEMALE_RATES table.

EMPLOYEE	FEMALE_RATES
GENDER	'F'
AGE	AGE

In the second relationship, rows in the EMPLOYEE table are related to the MALE_RATES table. This relationship is identical to the first, except that the value in the GENDER column matches the string literal 'M'.

EMPLOYEE	MALE_RATES
GENDER	'M'
AGE	AGE

For any row in the EMPLOYEE table, only one of the relationships can be satisfied because the column EMPLOYEE.GENDER must be 'M' or 'F'. After the appropriate relationship is selected, the related rows are identified by comparing values in the AGE columns.

Manage Generic Relationships

Some databases contain sets of tables that are identical except for the Creator ID. Rather than define a relationship for each set of tables, you can define a generic relationship that applies for all sets of tables that have the same base name, regardless of the Creator ID.

You can:

- Create a generic relationship in the same way you create an explicit relationship, except that you select **Generic** from the **Tools** menu in the Relationship Editor before saving.
- Modify a generic relationship in the same way you modify an explicit relationship.
- Convert a new or existing explicit Optim relationship or a database relationship to a generic relationship. (However, you cannot convert a generic relationship to an explicit relationship.)
- Specify a different Creator ID to identify the base tables associated with a generic relationship. Use this option when a base table associated with a generic relationship changes.

Note: The Creator ID of the parent and child tables used to create the generic relationship must be the same.

Convert to a Generic Relationship

To convert a database relationship or explicit Optim relationship to a generic relationship, select an existing relationship or select parent and child tables for a new relationship and open the Relationship Editor.

On the **Tools** menu, select **Generic**. (The Creator ID for parent and child tables must be the same to enable this option.) If the relationship is defined to the database, there are no additional prompts.

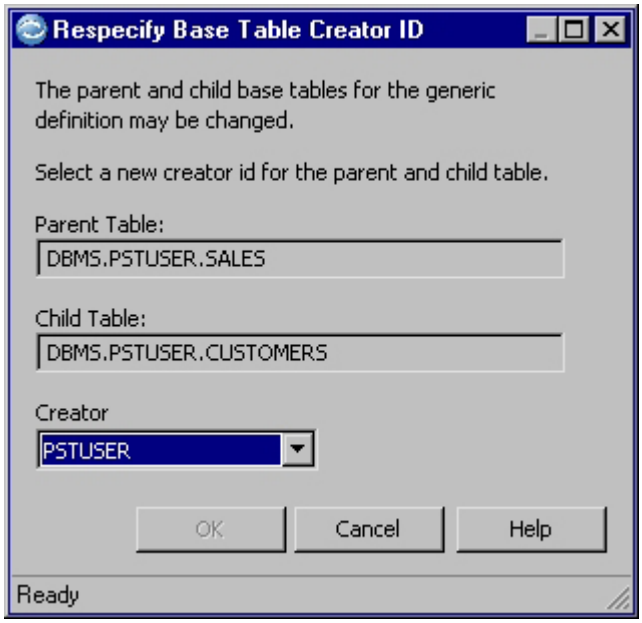
If you select an Optim relationship, a prompt asks whether to retain or delete the original explicit relationship. If you choose to retain the original, changes to the generic relationship have no effect on it.

When you convert a relationship to generic, the Relationship Editor displays the Creator ID of the explicit relationship as the Base Creator ID. The relationship is *Untitled* until you save the generic relationship. After you save a generic relationship, the Creator ID portion of the relationship name is replaced with an asterisk (*), for example, *dbalias.*.tablename constraint*.

Respecify the Base Table Creator ID

You can use a generic relationship for any pair of identical tables that have a different Creator ID. If the base tables used to define the generic relationship change, you can respecify the Creator ID.

In the Relationship Editor, select **Modify Base Table** from the **Tools** menu to display the Respecify Base Table Creator ID dialog:



Parent Table

Name of the parent table used to create the generic relationship.

Child Table

Name of the child table used to create the generic relationship.

Creator ID

Enter a different Creator ID, or click the down arrow to select from a list of Creator IDs. The specified Creator ID must be valid for both tables.

Note: Your selection appears as the Base Creator ID in the Relationship Editor. You may then save the generic relationship.

Confirm Option

By default, you are prompted to confirm before deleting a definition. To disable this feature, select **Personal** from the **Options** menu in the main window. See Chapter 18, “Personal Options,” on page 427 for more information.

Compatibility Rules for Relationships

A relationship is a defined connection between the rows of two tables that is determined by values in selected parent table columns that correspond to values in child table columns. When you define an Optim relationship, the corresponding values must be compatible.

Column Type	Is Compatible With
Character Column	Character Column
	Numeric Column
	String Literal
	Character Expression
Numeric Column	Numeric Column
	Numeric Constant
	Character Column
Binary Column	Binary Column
	Hexadecimal Literal
	Binary Expression
Boolean Column	Boolean Column
	Boolean Constant (True or False)
Date Time Column	Date Time Column
Date Column	Date Column
Time Column	Time Column
Interval Column	Interval Column

Note:

- In processing, a value is converted to the data type needed to select related rows. By default, the result of converting a numeric value to a character datatype is right-justified with leading zeros. Special registry settings allow you to change the default to left justification with leading or trailing spaces. Also, a character to numeric pairing requires a scale equal to 0 for the numeric column.
- You can use NULL for any null eligible column.
- Unicode or multi-byte columns must be of the same character set.

Data Types

The following classes of data and associated data types are supported. These data classes are important for data compatibility when you use column values in relationships.

Class	DBMS	Data Types
Character	DB2	CHAR, VARCHAR, CLOB
	Oracle	CHAR, VARCHAR2, LONG, CLOB, NCLOB, NCHAR, NVARCHAR

Class	DBMS	Data Types
	Sybase ASE	CHAR, VARCHAR, TXT
	SQL Server	CHAR, VARCHAR, TXT
	Informix	CHAR, VARCHAR, TXT
	Note: Single-byte character columns are not compatible with multi-byte or Unicode character columns.	
Numeric	DB2	INTEGER, SMALLINT, DECIMAL, FLOAT, DOUBLE
	Oracle	NUMBER, FLOAT
	Sybase ASE	TINYINT, INT, SMALLINT, DECIMAL, FLOAT, REAL, MONEY, SMALL MONEY
	SQL Server	TINYINT, INT, SMALLINT, DECIMAL, FLOAT, REAL, MONEY, SMALL MONEY
	Informix	INTEGER, SMALLINT, DECIMAL, FLOAT, REAL, DOUBLE PRECISION, SMALLFLOAT, SERIAL, MONEY, NUMERIC
Binary	DB2	CHAR (for Bit Data), VARCHAR (for Bit Data), BLOB
	Oracle	RAW, LONG RAW
	Sybase ASE	BINARY, VARBINARY, IMAGE
	SQL Server	BINARY, VARBINARY, IMAGE
	Informix	BYTE
Boolean	Sybase ASE	BOOLEAN (TRUE or FALSE)
Datetime	DB2	TIMESTAMP
	Oracle	DATE, TIMESTAMP, TIMESTAMP WITH LOCAL TIME ZONE, TIMESTAMP WITH TIME ZONE
	Sybase ASE	DATETIME, SMALL DATE TIME
	SQL Server	DATETIME, SMALL DATE TIME
	Informix	DATE, DATETIME
Date	DB2	DATE
	Oracle	DATE
	Informix	DATE
Time	DB2	TIME
Interval	Oracle	YEAR/MONTH INTERVAL, DAY/SECOND INTERVAL
	Informix	YEAR/MONTH INTERVAL, DAY/TIME INTERVAL

Chapter 10. Table Maps

A Table Map identifies and matches two sets of tables.

Use a Table Map to:

- Direct the placement of data in a Convert, Create, Insert, Load, or Restore Process.
- Exclude one or more tables from a Compare, Convert, Create, Insert, Load, or Restore Process.

Depending on the process for which you are using a Table Map, the two sets of tables are referred to as Source Tables and Destination Tables, or Source 1 Tables and Source 2 Tables:

- **Source Tables** are extracted or archived tables that contain data to be used in a Create, Convert, Insert, Load, or Restore Process.
- **Destination Tables** are the tables into which data is created, converted, inserted, loaded, or restored.
- **Source 1 Tables** and **Source 2 Tables** contain data to be used in a Compare Process.

Note: Matched tables can have different Creator IDs or names.

Table Maps are stored in the Optim Directory or embedded in a process request.

As an option, you can use a Column Map for any pair of tables in a Table Map. Column Maps identify and match columns in a pair of tables. A Column Map must be used when column names or attributes are dissimilar or data transformations are needed. For complete details on defining and editing Column Maps, see Chapter 5, "Column Maps," on page 121.

Naming Conventions

The fully qualified name of a Table Map is in the form: *identifier.name*.

identifier
: Identifier assigned to the Table Map (1 to 8 characters).

name
: Name assigned to the Table Map (1 to 12 characters).

It is helpful to use a logical set of naming conventions to identify the use for each and to organize definitions for easy access.

Section Contents

This section explains how to create and maintain Table Maps, including how to:

- Specify tables to correlate in a Table Map.
- Specify Column Maps in a Table Map.
- Merge Table Maps.
- Edit Archive Actions, when a Table Map is used in a Restore Request.

Open the Table Map Editor

Use the Table Map Editor to create and maintain Table Maps. There are different ways to open the editor depending on whether you want to create a new Table Map or select a Table Map to edit.

Create a Table Map

This section explains how to create a Table Map from the main window or the Table Map Editor.

From the Main Window for Move or Archive

To create a Table Map for Move or Archive:

1. In the main window, select **New** from the **File** menu.
2. Select **Table Map** from the **Definitions** submenu to open the New Table Map dialog.
3. For Validation Rules, select **Move/Archive**.
4. Specify the source for the tables you want to map:
   - If you select the **File** option, supply the file name.
   - If you select the **Access Definition** option, supply the name of the Access Definition.
5. Click **OK** to open the Table Map Editor.
6. Specify the Default Qualifier in the **Destination** box for the tables you want to map.
7. Modify the Destination Table names, as needed to map the data to the destination tables.
8. In the Table Map Editor, select **Save** from the **File** menu to open the Save the Table Map dialog.
9. In the **Pattern** box, specify a unique Table Map name and then click **Save**

From the Main Window for Compare

To create a Table Map for Compare:

1. In the main window, select **New** from the **File** menu.
2. Select **Table Map** from the **Definitions** submenu to open the New Table Map dialog.
3. For Validation Rules, select **Compare**.
4. Specify Source 1 for the tables you want to map:
   - If you select the **File** option, supply the file name.
   - If you select the **Access Definition** option, supply the name of the Access Definition.
5. Specify Source 2 for the tables you want to map:
   - If you select the **File** option, supply the file name.
   - If you select the **Access Definition** option, supply the name of the Access Definition.
6. Click **OK** to open the Table Map Editor.
7. If you select **Database Tables** for Source 2, specify the Default Qualifier for the Source 2 tables you want to map.
8. Modify the Source 2 Table names, as needed, to map the data.
9. In the Table Map Editor, select **Save** from the **File** menu to open the Save the Table Map dialog.
10. In the **Pattern** box, specify a unique Table Map name and then click **Save**.

From the Table Map Editor

To create a Table Map:

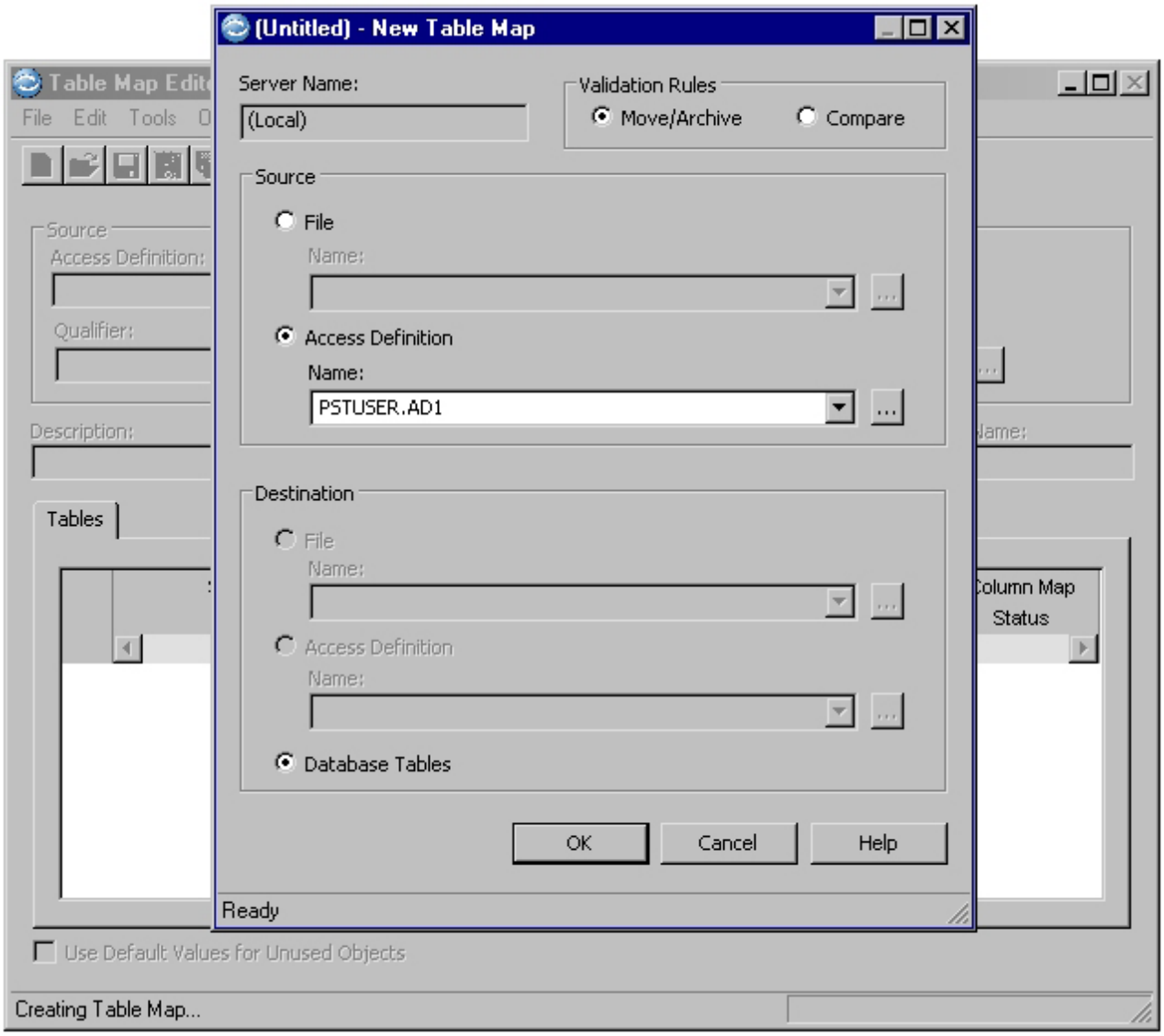
1. In the main window, select **Table Map** from the **Definitions** menu to open the Table Map Editor and last edited Table Map.
2. Your next step depends on your purpose:
   - To create a new Table Map, select **New** from the **File** menu in the Table Map Editor to open the New Table Map dialog. Specify the source/destination tables OR Source 1/Source 2 tables, and click **OK** to open the Table Map Editor.
   - To create a new Table Map modeled on an existing one, open the desired Table Map and select **Save As** from the **File** menu in the Table Map Editor.
   - To create and store a copy of the active Table Map and continue editing, select **Save Copy As** from the **File** menu in the Table Map Editor.

These steps are the minimum required to create a Table Map. Because the options to create and modify Table Maps are similar, see “Using the Editor” on page 233 for details.

Note: In addition, you can open the Table Map Editor from the **Tools** menu in the Convert Request Editor, Insert Request Editor, or Load Request Editor. For details, see the appropriate chapter in the corresponding user manual.

New Table Map Dialog

When you create a Table Map, the New Table Map dialog and the Table Map Editor open at the same time. After selecting the appropriate Source Table options, select **OK** to return to the Table Map Editor.



Server Name

If the optional Optim Server component is installed on your network, click the down arrow to select **Local** if the file is accessed from the client workstation, or name of the server on which the file resides.

Validation Rules

If the Table Map is to be used in a Convert, Create, Insert, or Load Process, select **Move/Archive** validation rules. Select **Compare** for use in a Compare Process. The group box labels vary according to your selection. In the previous illustration, Move/Archive validation rules are selected and the group

boxes are labeled **Source** and **Destination**. For Compare validation rules, the labels are **Source 1** and **Source 2**.

Source or Source 1

When you create a new Table Map, you must indicate the source for the objects you want to map. You can specify the source as an Extract or Archive File, or as an Access Definition:

File Select this option to map source tables from an Extract or Archive File.

Name Directory path and name of the file. If a directory path is not specified, the data directory path specified in Personal Options is used.

- To select from a list of recently used file names, click the down arrow.
- To select from a directory, click the browse button.

Access Definition
Select this option to map source objects referenced by an Access Definition.

Name Fully qualified name of the Access Definition. If a directory path is not specified, the data directory path specified in Personal Options is used by default.

- To select from a list of recently used Access Definitions, click the down arrow.
- To select from a list of available definitions, click the browse button.

Destination or Source 2

For Move/Archive validation rules, the destination can only be Database Tables. For Compare validation rules, all Source 2 options apply. Select:

File Select this option to map to source tables from an Extract or Archive File. This option is enabled if Compare validation rules are selected.

Name Directory path and name of the file. If a directory path is not specified, the data directory path specified in Personal Options is used.

- To select from a list of recently used file names, click the down arrow.
- To select from a directory, click the browse button.

Access Definition
Select this option to map to source objects referenced by an Access Definition. This option is enabled if Compare validation rules are selected.

Name Fully qualified name of the Access Definition. If a directory path is not specified, the data directory path specified in Personal Options is used by default.

- To select from a list of recently used Access Definitions, click the down arrow.
- To select from a list of available definitions, click the browse button.

Database Tables
Select this option to map destination or source objects in a database.

Select a Table Map to Edit

You can select a Table Map for editing from the main window or the Table Map Editor. The most common method is explained first.

From the Main Window

To select a Table Map:

1. In the main window, select **Open** from the **File** menu to display the Open (object selection) dialog. (See "Using the Open Dialog" on page 25 for more information on the Open dialog.)
2. Double-click **Table Map** to expand the identifier list.

3. Double-click the desired identifier to display a list of Table Maps.
4. *Optional* – Specify a **Pattern** to limit the list based on your criteria and click **Refresh**.
5. Double-click the desired Table Map to open the Table Map Editor.

Note: To select the last Table Map you edited, select **Table Map** from the **Definitions** menu in the main window to open the Table Map Editor and the last edited Table Map.

From the Table Map Editor

To select a different Table Map:

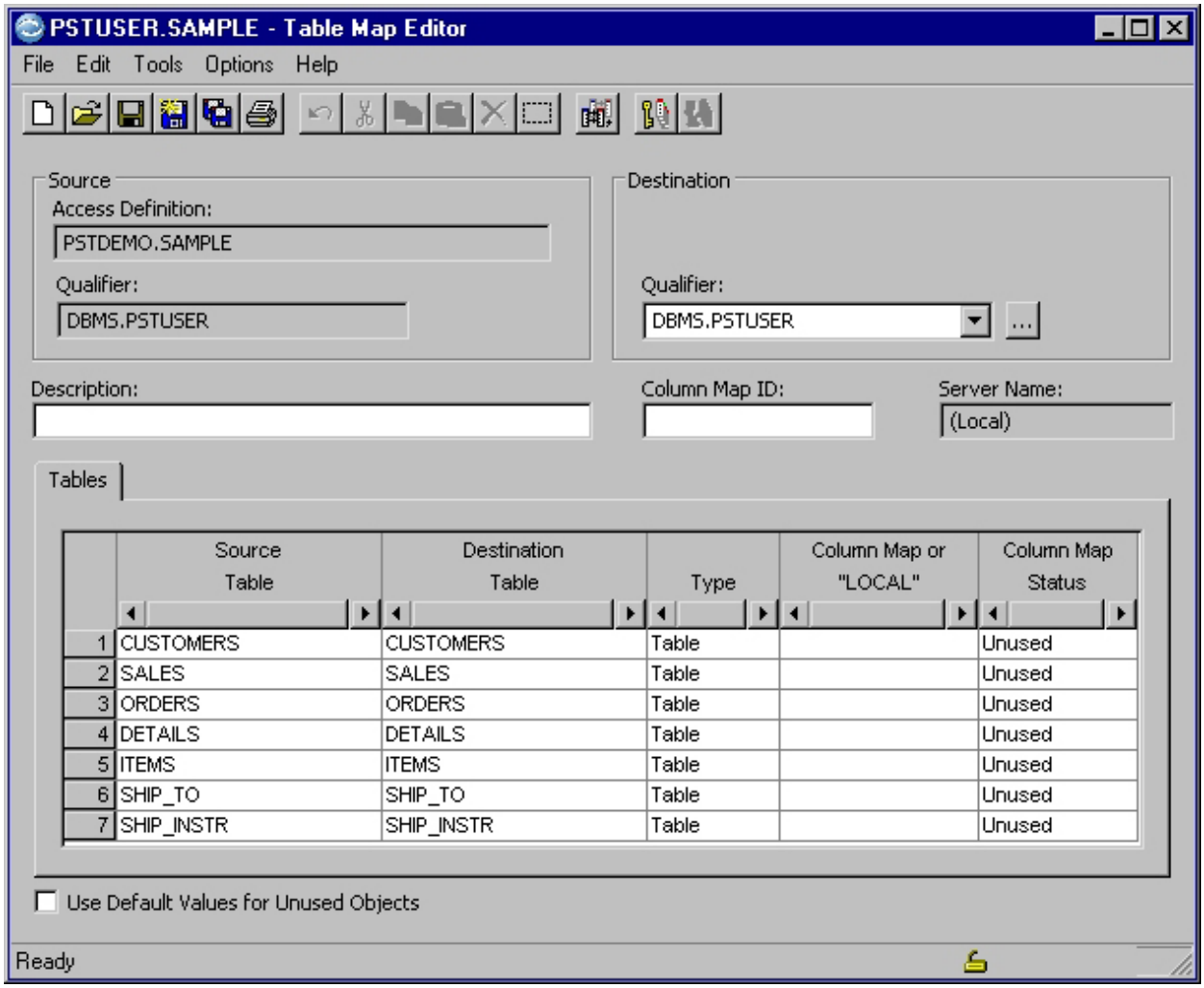
1. In the Table Map Editor, select **Open** from the **File** menu to display the Open a Table Map dialog. (See"Using the Open Dialog" on page 25 for more information on the Open dialog.)
2. Double-click the desired identifier to display a list of Table Maps.
3. *Optional* – Specify a **Pattern** to limit the list based on your criteria and click **Refresh**.
4. Double-click the desired Table Map to open the Table Map Editor.

Using the Editor

In the Table Map Editor, you can create, modify, or delete Table Maps stored in the Optim Directory or defined within an Action Request.

You can edit a Table Map in any or all of the following ways:

- Specify Destination or Source 2 table names or select from a list.
- Merge Table Maps to include tables, Column Maps, or both.
- Include Column Maps in a Table Map manually, select from a list, or specify Column Maps automatically.



When you create a new Table Map, the source and destination table grid cells in the Table Map Editor are populated automatically, according to options selected on the New Table Map dialog (see "Create a Table Map" on page 229). You can edit the destination table names and you can create or specify Column Maps, as needed.

Source or Source 1

Qualifier provides the default DB Alias and Creator ID for source tables in the Extract or Archive File or referenced in the Access Definition that are not fully qualified.

Destination or Source 2

Enter the default qualifier (*dbalias.creatorid*) for the destination table(s) listed in the Table Map. Click the down arrow or the browse button to select from a list.

Note:

- You must specify a valid default qualifier before you can save the Table Map.
- If you select an Archive File stored on tape or Centera from the drop-down list, the file is recalled to the Alternate File Path specified in the Storage Profile.

Description

Text to describe or explain the purpose of the Table Map (up to 40 characters).

Column Map ID

The Column Map Identifier qualifies any Column Maps specified by the base name only. If you specify a fully qualified Column Map name in the **Column Map or "LOCAL"** grid column, the **Column Map ID** is ignored. When you create a new Table Map, the **Column Map ID** is initially blank.

Server Name

Location of the Extract File containing the source tables, if applicable, is displayed.

Tabs

Tabs on the Table Map Editor correspond to the types of objects in the Extract or Archive File or referenced in the Access Definition. If an object type is not in the source, the tab is hidden. Only the **Tables** tab is enabled when Compare validation rules are used.

Use Default Values for Unused Objects

The effect of this option depends upon the method used to open the Table Map Editor:

- If you open the Table Map Editor from a process request, select this option to update the Table Map to include all unused Extract or Archive File objects in the Table Map. The Table Map is updated by the process request.
- If you open the Table Map Editor from the main window, select this option to include unused objects automatically whenever the Table Map is specified in a process request.

Tools Menu Commands

In addition to the standard commands on the **File**, **Edit**, and **Tools** menus, you can select the following commands from the **Tools** menu:

Convert to Local
: When referencing a named Table Map in an action request, select this command to save the Table Map within the action request.

Merge...
: Open the Merge Table Map dialog to allow you to select a Table Map to overlay all or part of the active Table Map.

Populate...
: Open the Populate Column Map dialog to allow you to specify criteria for selecting a Column Map for each pair of tables in the Table Map. If the Populate routine finds more than one Column Map that meets your criteria, you can select from a list of possible matches.

Populate
: Populate Destination or Source 2 object names, as follows:
  - Select **Add** to insert the source object names in any blank destinations.
  - Select **Clear** to remove all destination object names and insert the source object names.
  - Select **Empty** to remove all destination object names and leave each cell blank.
  - Select **Copy This Entry** to copy the source object name to the current destination object name.

Default All Archive Actions
Reset Archive Actions for all tables to the Access Definition defaults. You can also right-click the table list and select this command from the shortcut menu. See "Create or Modify Archive Actions in a Table Map" on page 247.

Tables Tab

Select the **Tables** tab to review mapped database tables or views. The source must be a table, but the destination can be a table or a view.

Source Table or Source 1 Table
List of source table names. You cannot modify these values.

Destination Table or Source 2 Table
List of destination tables. When you create a new Table Map, the destination names are initially the same as those listed for the source. You can modify these values.

Type Type of object named in **Destination Table** or **Source 2 Table**:

Table Destination table is valid.

View Destination view is valid.

Unknown
Destination table or view does not exist or is not recognized.

Unused
Destination table or view is not specified.

A-Table
Destination is an alias of a table.

A-View
Destination is an alias of a view.

View Error
View is invalid and not usable.

Pending
A default qualifier is needed for the destination. To avoid validation errors, specify a valid destination default qualifier for the mapped tables.

Inaccessible
DB Alias cannot be accessed.

Incomplete
Destination table or view name is not fully qualified. You must specify:

- A valid Default Qualifier under **Destination** or **Source 2**, or
- A fully qualified name in the **Destination Table** or **Source 2 Table** grid column.

Column Map or "LOCAL"
Name of the Column Map for a specific pair of tables in the Table Map:

- If you specify unqualified Column Map names, you must specify a **Column Map ID**.
- If you specify **LOCAL**, the Column Map is saved with the Table Map and is available only to that Table Map.

For information on referencing Column Maps in a Table Map, see "Specify Column Maps in a Table Map" on page 247.

Column Map Status
Status of the Column Maps referenced in the Table Map. This status is provided automatically:

Defined
Column Map exists.

Incomplete
Fully qualified name of the Column Map or the Column Map ID must be specified.

Unknown
Column Map does not exist.

Unused
Column Map is not used.

Defaults Tab

Select the **Defaults** tab to review mapped source and destination defaults.

Source Default
List of source defaults. You cannot modify these values.

Destination Default
List of destination defaults. When you create a new Table Map, these names are initially the same as those listed for the source. You can modify these values.

Status Automatically provides information about the destination default:

Defined
Default exists in the destination database.

Inaccessible
Destination database cannot be accessed.

Incomplete
Destination default is not fully qualified. Specify a valid destination qualifier, or specify a fully qualified default name.

Pending
When creating a new Table Map, the destination defaults are initially listed, but you must specify a valid destination Qualifier.

Unknown
Destination default does not exist.

Unsupported
Defaults not supported in the database.

Unused
Destination default is not specified.

Functions Tab

Select the **Functions** tab to review mapped source and destination database functions.

Source Function
List of source functions. You cannot modify these values.

Destination Function
List of destination functions. When you create a new Table Map, these names are initially the same as those listed for the source. You can modify these values.

Status Automatically provides information about the destination function:

Defined
Function exists in the destination database.

Inaccessible
Destination database cannot be accessed.

Incomplete
Destination function is not fully qualified. Specify a valid destination qualifier, or specify a fully qualified function name.

Pending
When creating a new Table Map, the destination functions are initially listed, but you must specify a valid destination Qualifier.

Unknown
Destination function does not exist.

Unsupported
Functions not supported in the database.

Unused
Destination function is not specified.

Packages Tab

Select the **Packages** tab to review mapped source and destination database packages.

Source Package
List of source packages. You cannot modify these values.

Destination Package
List of destination packages. When you create a new Table Map, these names are initially the same as those listed for the source. You can modify these values.

Status Automatically provides information about the destination package:

Defined
Package exists in the destination database.

Inaccessible
Destination database cannot be accessed.

Incomplete
Destination package is not fully qualified. Specify a valid destination qualifier, or specify a fully qualified package name.

Pending
When creating a new Table Map, the destination packages are initially listed, but you must specify a valid destination qualifier.

Unknown
Destination package does not exist.

Unsupported
Packages not supported in the database.

Unused
Destination package is not specified.

Procedures Tab

Select the **Procedures** tab to review mapped source and destination database procedures.

Source Procedure
List of source procedures. You cannot modify these values.

Destination Procedure
List of destination procedures. When you create a new Table Map, these names are initially the same as those listed for the source. You can modify these values.

Status Automatically provides information about the destination procedure:

Defined
Procedure exists in the destination database.

Inaccessible
Destination database cannot be accessed.

Incomplete
Destination procedure is not fully qualified. Specify a valid destination qualifier, or specify a fully qualified procedure name.

Pending
When creating a new Table Map, the destination procedures are initially listed, but you must specify a valid destination qualifier.

Unknown
Destination procedure does not exist.

Unsupported
Procedures not supported in the database.

Unused
Destination procedure is not specified.

Rules Tab

Select the **Rules** tab to review mapped source and destination rules.

Source Rule
List of source rules. You cannot modify these values.

Destination Rule
List of destination rules. When you create a new Table Map, these names are initially the same as those listed for the source. You can modify these values.

Status Automatically provides information about the destination rule:

Defined
Rule exists in the destination database.

Inaccessible
Destination database cannot be accessed.

Incomplete
Destination rule is not fully qualified. Specify a valid destination qualifier, or specify a fully qualified rule name.

Pending
When creating a new Table Map, the destination rules are initially listed, but you must specify a valid destination Qualifier.

Unknown
Destination rule does not exist.

Unsupported
Rules not supported in the database.

Unused
Destination rule is not specified.

Sequences Tab

Select the **Sequences** tab to review mapped source and destination sequences.

Source Sequence
List of source sequences. You cannot modify these values.

Destination Sequence
List of destination sequences. When you create a new Table Map, these names are initially the same as those listed for the source. You can modify these values.

Status Automatically provides information about the destination sequence:

Defined
Sequence exists in the destination database.

Inaccessible
Destination database cannot be accessed.

Incomplete
Destination sequence is not fully qualified. Specify a valid destination qualifier, or specify a fully qualified sequence name.

Pending
When creating a new Table Map, the destination sequences are initially listed, but you must specify a valid destination qualifier.

Unknown
Destination sequence does not exist.

Unsupported
Defaults not supported in the database.

Unused
Destination sequence is not specified.

User Defined Types Tab

Select the **User Defined Types** tab to review mapped source and destination user defined types (UDTs).

Source UDT
List of source UDTs. You cannot modify these values.

Destination Rule
List of destination UDTs. When you create a new Table Map, these names are initially the same as those listed for the source. You can modify these values.

Status Automatically provides information about the destination UDT:

Defined
UDT exists in the destination database.

Inaccessible
Destination database cannot be accessed.

Incomplete
Destination UDT is not fully qualified. Specify a valid destination qualifier, or specify a fully qualified UDT name.

Pending
When creating a new Table Map, the destination UDTs are initially listed, but you must specify a valid destination Qualifier.

Unknown
Destination UDT does not exist.

Unsupported
UDTs not supported in the database.

Unused
Destination UDT is not specified.

Views Tab

Select the **Views** tab to review mapped source and destination database views. The source and the destination must be a view. (Note that on the **Tables** tab, the source must be a table, and the destination can be a table or a view.)

Source View
List of source views. You cannot modify these values.

Destination View
List of destination views. When you create a new Table Map, these names are initially the same as those listed for the source. You can modify these values.

Status Automatically provides information about the destination view:

Defined
View exists in the destination database.

Inaccessible
Destination database cannot be accessed.

Incomplete
Destination view is not fully qualified. Specify a valid destination qualifier, or specify a fully qualified view name.

Pending
When creating a new Table Map, the destination views are initially listed, but you must specify a valid destination qualifier.

Unknown
Destination view does not exist.

Unsupported
Views not supported in the database.

Unused
Destination view is not specified.

Assemblies Tab

Select the **Assemblies** tab to review mapped source and destination database assemblies.

Source Assemblies
List of source assemblies. You cannot modify these values.

Destination Assemblies
List of destination assemblies. When you create a new Table Map, these names are initially the same as those listed for the source. You can modify these values.

Status Automatically provides information about the destination assemblies:

Defined
Assembly exists in the destination database.

Inaccessible
Destination database cannot be accessed.

Incomplete
Destination assembly is not fully qualified. Specify a valid destination qualifier, or specify a fully qualified view name.

Pending
When creating a new Table Map, the destination assemblies are initially listed, but you must specify a valid destination qualifier.

Unknown
Destination assembly does not exist.

Unsupported
Assemblies not supported in the database.

Unused
Destination assembly is not specified.

Partition Functions Tab

Select the **Partition Functions** tab to review mapped source and destination database partition functions.

Source Partition Functions
List of source partition functions. You cannot modify these values.

Destination Partition Functions
List of destination partition functions. When you create a new Table Map, these names are initially the same as those listed for the source. You can modify these values.

Status Automatically provides information about the destination partition functions:

Defined
Partition function exists in the destination database.

Inaccessible
Destination database cannot be accessed.

Incomplete
Destination partition function is not fully qualified. Specify a valid destination qualifier, or specify a fully qualified view name.

Pending
When creating a new Table Map, the destination partition functions are initially listed, but you must specify a valid destination qualifier.

Unknown
Partition function does not exist.

Unsupported
Partition functions not supported in the database.

Unused
Partition function is not specified.

Partition Schemes Tab

Select the **Partition Schemes** tab to review mapped source and destination database partition schemes.

Source Partition Schemes
List of source partition schemes. You cannot modify these values.

Destination Partition Schemes
List of destination partition schemes. When you create a new Table Map, these names are initially the same as those listed for the source. You can modify these values.

Status Automatically provides information about the destination partition schemes:

Defined
Partition scheme exists in the destination database.

Inaccessible
Destination database cannot be accessed.

Incomplete
Destination partition scheme is not fully qualified. Specify a valid destination qualifier, or specify a fully qualified view name.

Pending
When creating a new Table Map, the destination partition schemes are initially listed, but you must specify a valid destination qualifier.

Unknown
Partition scheme does not exist.

Unsupported
Partition schemes not supported in the database.

Unused
Partition scheme is not specified.

Specify Destination Objects in a Table Map

The Table Map Editor displays pairs of corresponding source and destination objects that are mapped on each tab. Initially, the destination (or source 2) object names are populated with the same names as the source (or source 1) object names. You can edit the destination object names or select destination object names from each corresponding object list.

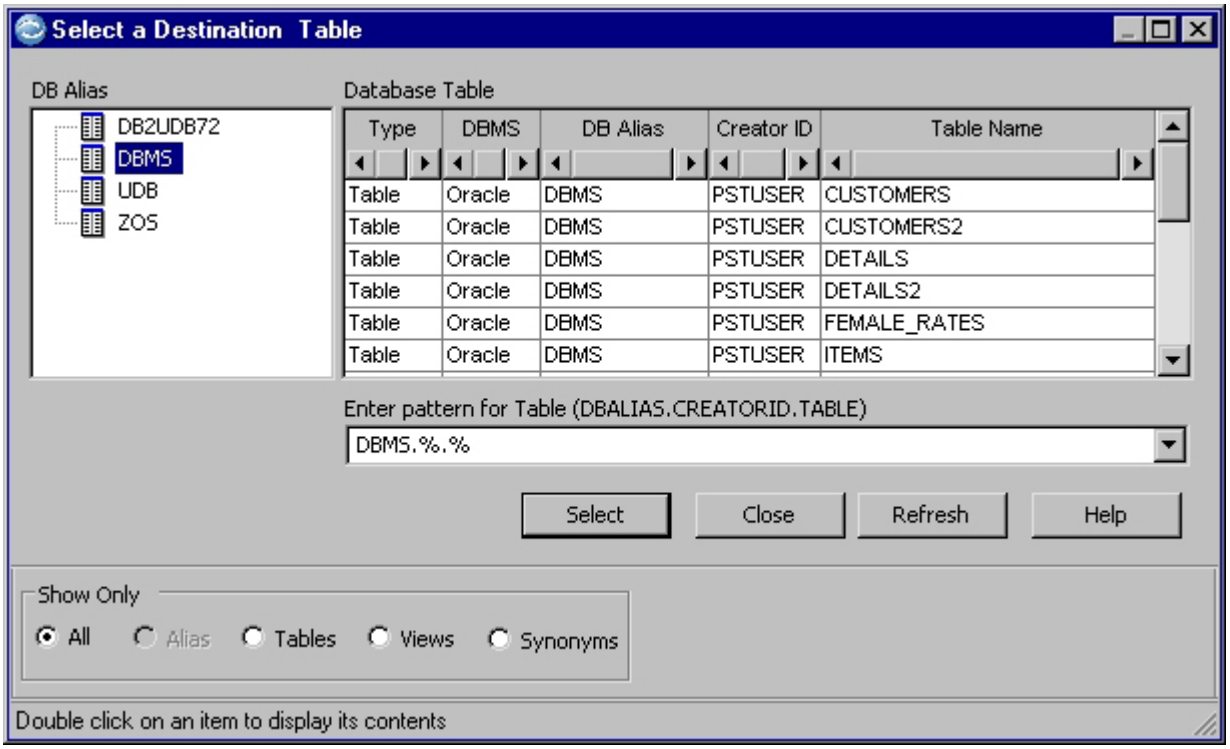
Edit Destination Names

You can edit destination names as follows:

- Overtype a destination name directly.
- Delete a destination name to ignore the corresponding source during a process.
- Change the default qualifier for all unqualified names.
- Right-click the Destination Table Column and select **Populate**.
  - Select **Add** to insert the source object names in any blank destinations.
  - Select **Clear** to remove all destination object names and insert the source object names.
  - Select **Empty** to remove all destination object names and leave each cell blank.
  - Select **Copy This Entry** to copy the source object name to the current destination object name.
- Right-click the Destination Table Column and select **List** to open the Select a Destination dialog.

Select Destination Object Names from a List

The Select a Destination dialog provides a list of DB Aliases on the left and the corresponding list of objects on the right.



DB Alias

The DB Aliases are listed on the left and the selected DB Alias is shaded. To change the selection, double-click a DB Alias or overtype the DB Alias in the **Pattern** box. (If you change this selection in either place, it changes automatically in the other.)

Objects

The type of object (table, view, alias, synonym) and fully qualified table names appear in the right list box. The list is sorted in alphabetical order by Creator ID and table name.

Pattern

You can specify a **Pattern** to limit the list of database objects in the Select a Destination dialog. After you specify a pattern, click **Refresh** to redisplay the list based on your criteria. See “Use a Pattern” on page 27 for more information.

Show Only

Select the specific type of object (all, alias, tables, views, synonyms) to display.

Specify Source 2 Tables in a Table Map

The Table Map grid displays the names of Source 1 tables to map to Source 2 tables. The Source 2 Table column is initially populated with the same table names as the Source 1 Table column.

You can change Source 2 table names:

- Select Source 2 table names by clicking the Source 2 Table column and selecting from a drop-down list of the tables available.

Note: Right-click and select **Show All Tables** to list all available Source 2 tables (unless you selected Database Tables for Source 2).

- Right-click and select **Populate** and then choose one of the following from the submenu:
  - **Add** to insert the Source 1 table names in any blank Source 2 grid cells.
  - **Clear** to remove all Source 2 table names and insert source table names.
  - **Empty** to remove all Source 2 table names and leave each cell blank.
  - **Copy This Entry** to copy the Source 1 table name to the current Source 2 table name (unless you selected File or Access Definition for Source 2).
- Delete a Source 2 table name to exclude the corresponding Source 1 table from the comparison.
- If you selected Database Tables for Source 2, you can right-click the Source 2 Table column and select **List** from the shortcut menu to select tables from the Select a Table dialog.

Merge Table Maps

Use the **Merge** command to overlay all or part of the Table Map in the Table Map Editor with the specifications from another Table Map. If the source table names in the two Table Maps match, **Merge** copies the associated destination table names and Column Maps from the merge Table Map to the Table Map in the Table Map Editor.

To merge Table Maps:

1. In the Table Map Editor, open a Table Map.
2. Select **Merge** from the **Tools** menu to open the Merge a Table Map dialog.
3. Select the Table Map you want to merge.
4. Specify a Target: click **Tables**, **Column Map**, or **Both**.
5. Select a Merge Type: click **Replace**, **Clear**, or **Add**.
6. Click **Open** to merge Table Maps and return to the Table Map Editor.

It is useful to merge Table Maps when you want to create:

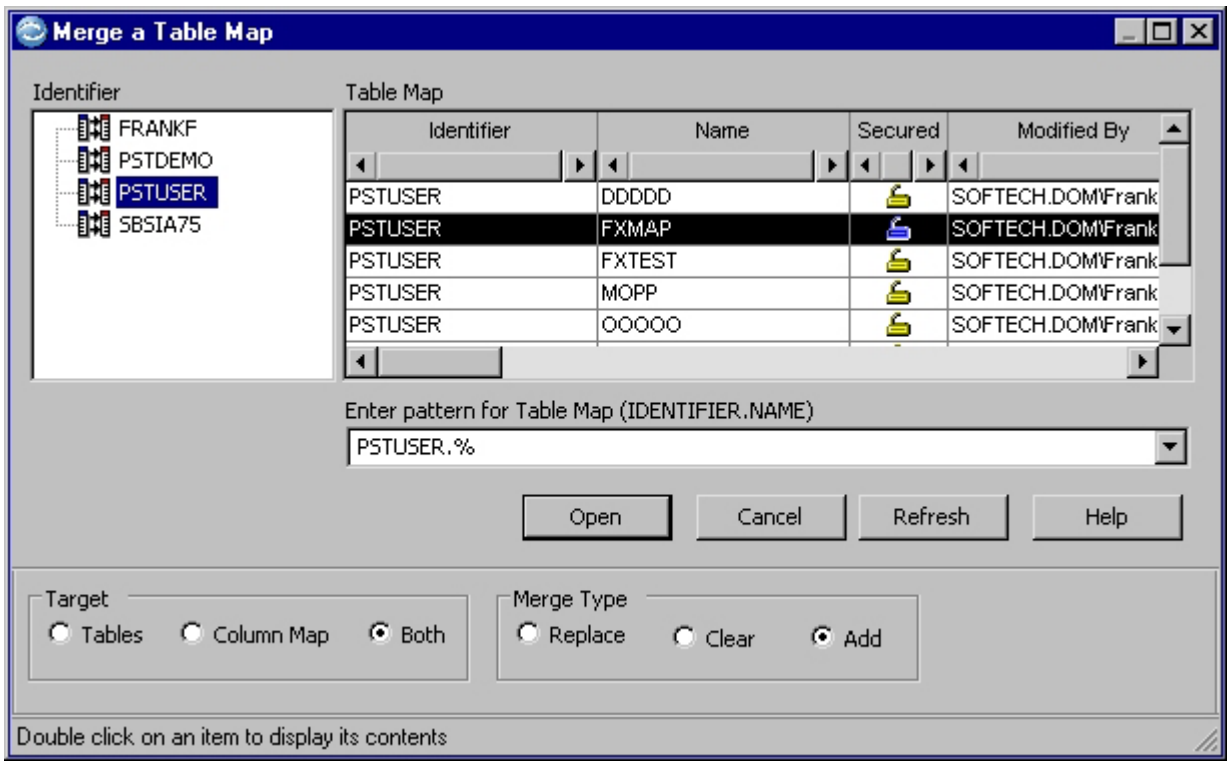
- A composite Table Map from a set of existing Table Maps.
- A Table Map that is a subset of a Table Map.

For example, suppose that you use a common pool of tables for many different process requests. To make processing easier, you create a master Table Map that defines the default mapping for these tables. With a master Table Map in place, you can merge the master Table Map into a specific Table Map for any process request.

Even though two process requests involve two different sets of tables, which may or may not overlap, you can merge the same master Table Map to define Table Maps for both. This feature ensures consistent mapping with minimum effort. After the merge, you can edit the Table Map in the Table Map Editor.

Merge a Table Map Dialog

The Merge a Table Map dialog is divided into two areas. The object identifiers are listed on the left and the associated Table Maps appear on the right. At the bottom of the dialog, you can select options to merge tables, columns, or both, and select the method used to merge the details.



Target

Tables Merge the names of the destination tables in a stored Table Map with those in the Table Map in the Table Map Editor. The Creator IDs are included.

Column Maps
Merge the names of the Column Maps in a stored Table Map with those in the Table Map in the Table Map Editor. The default Column Map ID is included.

Both Merge the names of the destination tables and the names of the Column Maps in a stored Table Map with those in the Table Map in the Table Map Editor. The Creator IDs for the tables and the default Column Map ID are included.

Merge Type

Replace
Replace the destination tables in the Table Map in the Table Map Editor when the source tables in the stored Table Map match those in the active Table Map.

If the source tables in the Table Map in the Table Map Editor and those in the stored Table Map do not match, then the destination tables in the Table Map in the Table Map Editor are retained.

Clear Remove the names of all destination tables in the Table Map in the Table Map Editor before merging.

If the source tables in the stored Table Map match those in the Table Map in the Table Map Editor, then insert the names of the destination tables from the stored Table Map into the Table Map in the Table Map Editor.

Add Insert the destination tables from the stored Table Map into the Table Map in the Table Map Editor when the source tables in the stored Table Map match those in the Table Map in the Table Map Editor and a destination table is not specified in the Table Map.

Create or Modify Archive Actions in a Table Map

Archive Actions are supplemental SQL statements (Actions) to be executed at selected phases of an Archive, Delete, or Restore Process. You create Archive Actions in an Access Definition. Archive Actions created in the Access Definition are used by default for each process.

From the Table Map Editor, you can create, modify, or delete one or more of the default Archive Actions pertaining to the action phases of a Restore Process.

Note: If one or more variables are defined in the original Access Definition specified by the Archive File, you can include them in the SQL statement of the Archive Action.

Right-click the Source Table column in a Table Map to select from the following Archive Actions commands:

Archive Actions
Opens the Archive Actions dialog.

Note: See "Archive Actions Tab" on page 85 for details about **Archive Actions**.

Default Archive Actions
Resets Archive Actions to the Access Definition defaults for the table name upon which you right-clicked.

Default All Archive Actions
Resets Archive Actions to the Access Definition defaults for all tables in the Table Map. You can also select this command from the **Tools** menu in the Table Map Editor.

Specify Column Maps in a Table Map

You can include Column Maps in a Table Map to map the columns in source and destination tables or to compare the columns in Source 1 and Source 2 tables. By default, columns that have the same names and matching attributes are mapped to each other.

Use Column Maps to accomplish the following:

- Direct or compare data between columns with different names or with compatible, but not exactly matching, data types.
- Manipulate the source data to be inserted at the destination.
- Exclude columns from a process.

Specify the name of the Column Map in a Table Map Editor in the following ways:

- Create a new Local or Named Column Map.
- Specify the name of an existing Column Map.
- Select a Column Map from a list.
- Populate Column Maps automatically.
- Merge Table Maps and Column Maps, or both.

Edit a Column Map

When you specify the name of an existing Column Map in the Column Map or "LOCAL" grid column, you can use the Column Map as it is or you can edit the Column Map. If changes were made to the database tables since the last time the Column Map was used, the specifications may no longer be valid.

To view or edit the Column Map, right-click and select **Open Column Map** from the shortcut menu to display the Column Map Editor. For complete details on how to create and edit Column Maps, see Chapter 5, "Column Maps," on page 121.

Create a New Column Map

Use the **Column Map or "LOCAL"** grid column to specify a new Local or Named Column Map in a Table Map:

Local Local Column Maps are saved as part of the Table Map definition and can be used by that Table Map only. Type the word LOCAL in the Column Map or "LOCAL" grid column. Right-click and select **Open Column Map** from the shortcut menu to open the Column Map Editor.

If you want to use a Local Column Map with other Table Maps, select **Save Copy As** from the **File** menu. For complete details about using the Column Map Editor, see Chapter 5, "Column Maps," on page 121.

Named

Named Column Maps are saved in the Optim Directory and can be reused with any other Table Map. Type the fully qualified name of a new Column Map in the Column Map or "LOCAL" grid column. Right-click and select **Open Column Map** from the shortcut menu to open the Column Map Editor. You can edit and save the Column Map.

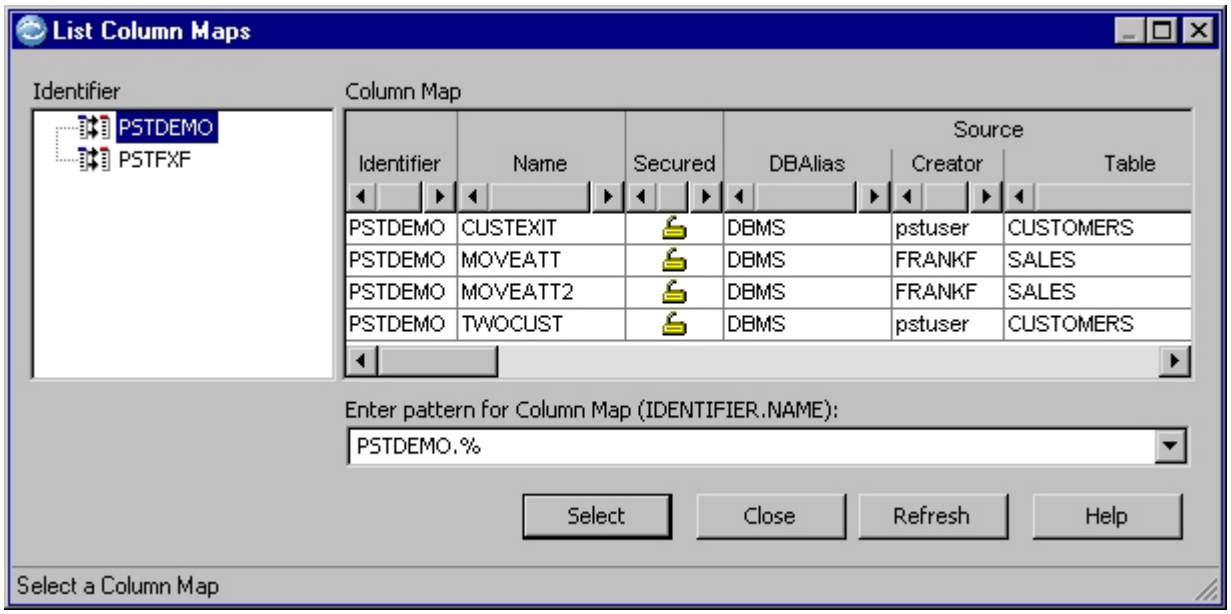
The Column Map Status changes to Unknown if you specify local or if you specify a Column Map name that does not exist. A message is displayed at the bottom of the editor to indicate that you must define the Column Map before you can save the Table Map.

Select Column Maps from a List

You can select an existing Column Map from a list. To display a list of Column Map names, right-click and select **List Column Maps** from the shortcut menu.

List Column Maps Dialog

The List Column Maps dialog shows all available Column Maps, whether or not the table names match names in the Table Map Editor. (You can use a Column Map defined for tables with different names as long as one or more column names match.)



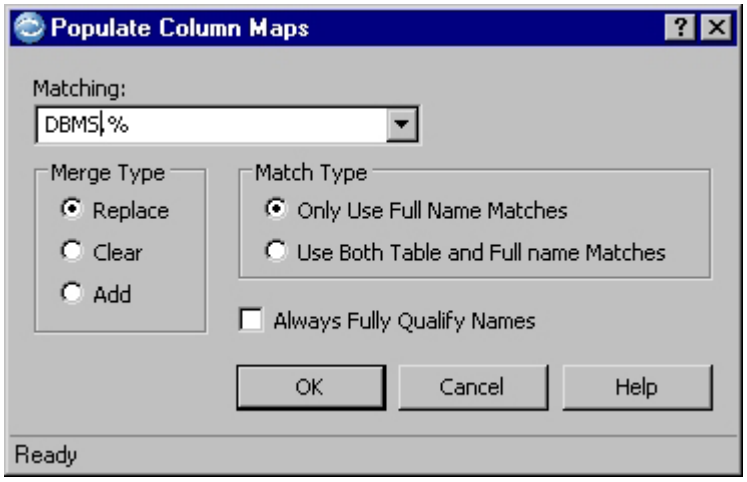
Pattern

Use a Pattern to limit the list of Column Maps in the List Column Maps dialog. After you specify a pattern, click **Refresh** to redisplay the list based on your criteria. See "Use a Pattern" on page 27 for more information.

Populate Column Maps in a Table Map

Populate automatically inserts the names of Column Maps into the Column Map or "LOCAL" grid column in the Table Map Editor. You can specify criteria for the Column Map names.

In the Table Map Editor, select **Populate...** from the **Tools** menu (or right-click and select **Populate...** from the shortcut menu) to open the Populate Column Maps dialog. Specify the criteria for selecting Column Maps.



Matching

A two-part qualifier for the Column Map names. You can use SQL LIKE syntax to specify the qualifier and limit the candidate list of Column Maps. Initially, %.% (the default) to search all Column Maps is displayed.

Merge Type

The method Populate uses to insert Column Map names in the Column Map or "LOCAL" grid cells in the Table Map Editor. Select one of the following:

Replace
Replace any Column Map name, LOCAL, or blank cell with a Column Map name that satisfies the Matching and Match Type criteria. If a Column Map that meets the criteria is not found, no change is made. If several Column Maps that satisfy the criteria exist for a set of tables, a list displays from which you can select one.

Clear Clear all Column Map or "LOCAL" grid cells and insert a Column Map name in each cell for which a Column Map exists that satisfies the Matching and Match Type criteria. If a Column Map that meets the criteria is not found, the cell remains blank. If several Column Maps satisfy the criteria, you can select from a list.

Add Insert Column Map names in blank **Column Map or "LOCAL"** grid cells that satisfy Matching and Match Type criteria. If a Column Map that meets the criteria is not found, the cell remains blank. If several Column Maps satisfy the criteria, you can select from a list.

Match Type

Indicate how Populate matches Column Maps to the Table Map. Select one of the following:

Use Only Full Name Matches
Select this option to display a list of Column Maps for which the fully qualified names of the source and destination tables in the Column Map match those in the Table Map.

Use Table and Full Name Matches

Select this option to display a list of Column Maps where:

- Fully qualified names of the tables referenced in the Column Map match those in the Table Map, and
- Fully qualified names of one or both tables referenced in the Column Map have Creator IDs different from those in the Table Map.

Always Fully Qualify Names

Select this check box to ensure that Populate inserts fully qualified names of Column Maps into the Table Map. This option is useful when you do not specify a default Column Map ID in the Table Map Editor or when the default does not contain the appropriate Creator ID.

Note: If you specify several Column Map names that differ only by the Identifier, and the Column Map ID you specified in the Table Map Editor is inappropriate or inaccurate, Populate may generate unexpected results.

Processing Sequence

Populate scans each pair of source and destination tables in the Table Map Editor to generate a list of Column Maps that match your criteria.

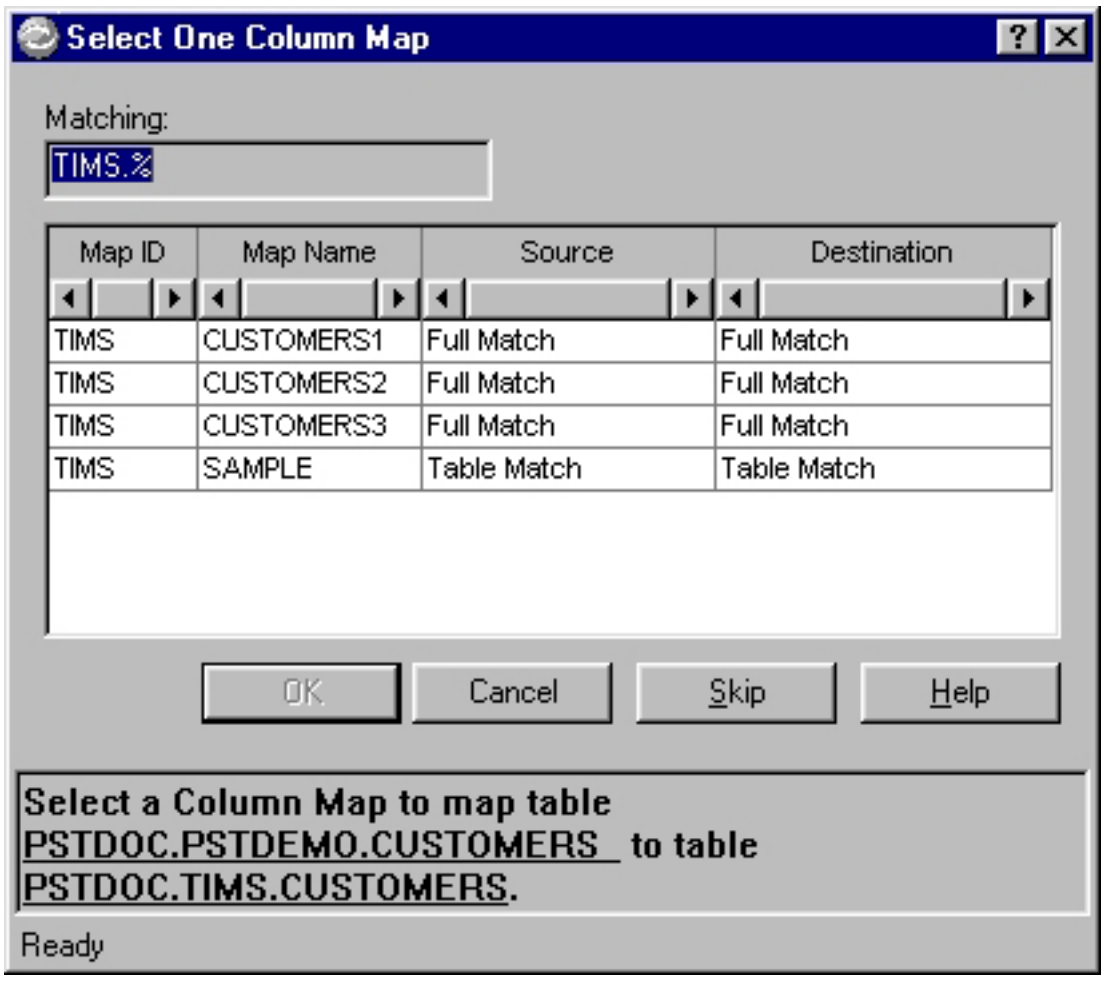
- If a single Column Map satisfies your criteria, Populate replaces, inserts, or adds the Column Map name in the Table Map automatically.
- If several Column Maps satisfy your criteria, a list displays from which you can select a Column Map or bypass the match.
- If no matches are found, Populate skips the pair of tables and processes the next pair.

The method for populating the **Column Map or "LOCAL"** grid column is determined by your **Merge Type** selection. The extent of the search is determined by your **Match Type** selection.

- If you select **Only Use Full Name Matches**, the search ends when the routine finds all full matches.
- If you select **Use Both Table and Full Name Matches**, the routine search includes table matches.

Select One Column Map

When Populate finds several matches for a pair of tables, the Select One Column Map dialog lists the candidate Column Maps. The full matches are listed first, followed by table matches. The names of the two tables that are being processed are displayed in the message box.



- To select a Column Map, click **OK**.
- To bypass a map selection for a pair of tables and continue processing with the next pair, click **Skip**. When no pairs of tables remain, the Select One Column Map dialog closes.

The Column Map or "LOCAL" grid column of the Table Map Editor lists the selected Column Maps and any inserted automatically.

Matching	SQL LIKE syntax you specified as the **Matching** criteria in the Populate Column Maps dialog.	
Map ID	Qualifier of the Column Map name that matches your criteria.	
Map Name	Name of the Column Map that matches your criteria.	
Source	Type of match between the source table in the Column Map and the source table in the Table Map.	
	Full Match	Names and Creator Ids of the source tables match.
	Table Match	Names of the source tables match, but the Creator IDs differ.
Destination	Type of match between the destination table in the Column Map and the destination table in the Table Map.	
	Full Match	Names and Creator IDs of the destination tables match.
	Table Match	Names of the destination tables match, but the Creator IDs differ.

Save a Table Map

The save commands are available from the **File** menu in the Table Map Editor. When you save a Table Map, you must supply a two-part name: *identifier.name*.

Before you can save a Table Map, you must provide a valid qualifier under **Destination** or **Source 2** in the Table Map Editor, ensure that all Column Maps are defined, and correct any other errors.

Chapter 11. Restart/Retry

Use the Restart/Retry Utility to restart a process that terminated abnormally or to retry a process when all rows are not successfully processed. These incomplete processes are referred to as "pending" processes. You can use this utility to restart or retry a Delete, Insert, or Insert/Update Process.

Restart

If a process ends abnormally, either by user request or because of system limitations, you can restart the process from the last commit point. A process can end abnormally when:

- Insufficient computer resources are allocated for the process.
- You chose to end the process before it completes normally.
- Processing exceeds specified limits (for example, when the process reaches the limit set for the maximum number of discarded rows).

Retry

If one or more rows did not process successfully, you can use information in the Control File to correct the problem and retry the process for those rows that did not process successfully. For example, a row may not process successfully when a request conflicts with referential integrity rules (such as when a parent row cannot be deleted because child rows exist for that row).

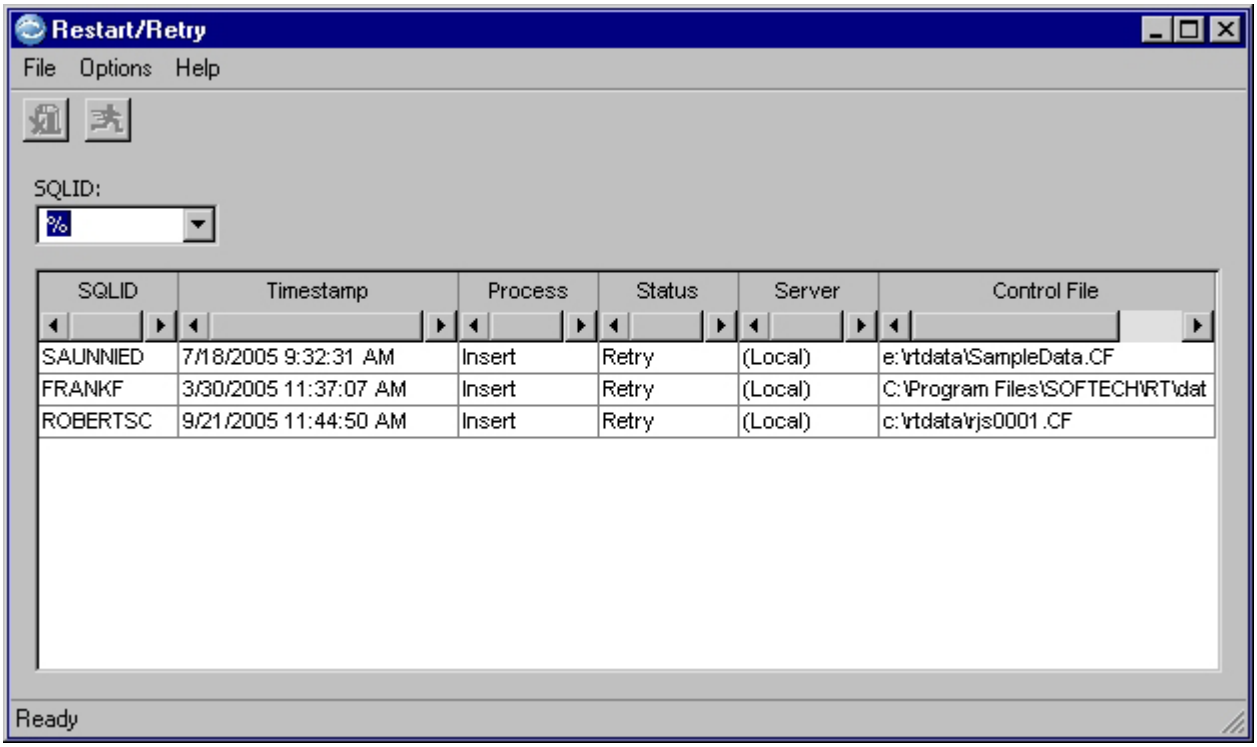
Contents

This section explains how to use the Restart/Retry Utility to:

- Review a list of pending processes.
- View details for a particular pending process.
- Retry or Restart a pending process.

Open Restart/Retry

On the **Utilities** menu, select **Restart/Retry** to open the Restart/Retry dialog.



The Restart/Retry dialog displays identification and processing information about pending processes.

SQLID

Enter the SQLID identifier to limit the list of pending processes. This identifier limits the list to a specific user or project, depending on site naming conventions. To select from a list of recent SQLID values, click the down arrow. To display all pending processes, use the wildcard character (%).

Grid Details

SQLID
Identifier assigned to a process.

Timestamp
Date and time the process was run.

Process
Type of process that was run.

Status Current status of each pending process:

Retry One or more rows were not processed successfully.

Restart
Process ended abnormally.

***Invalid**
The path to the Archive, Extract, or Control File associated with the process is invalid. You cannot retry or restart a process that is assigned *Invalid status without correcting the path. Delete the process to remove it from the list.

Server Name of server where Control File is located, or "Local" if Control File is located on the workstation.

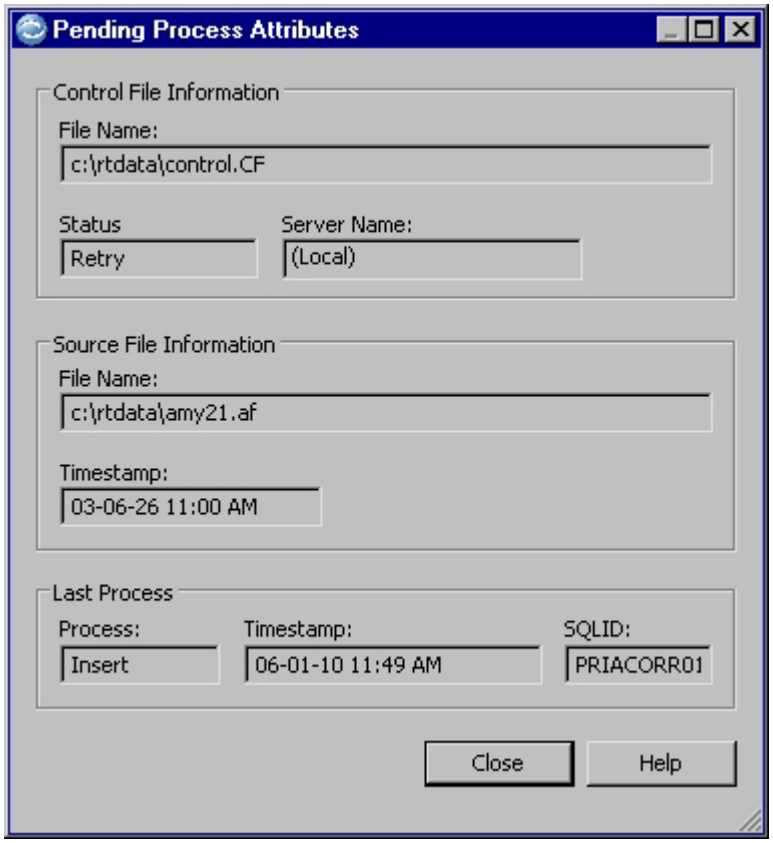
Control File
Directory path and name of the Control File specified for the process request.

Using Restart/Retry

Select an individual pending process to review details of the process, restart the process from the last commit point, or retry unprocessed rows.

Review a Pending Process

Select the pending process to review from the list. On the **File** menu, select **Display** or right-click and select **Display** from the shortcut menu. The Pending Process Attributes dialog is displayed.



Details

This dialog contains read only information and cannot be modified.

Control File Information	
File Name	Fully qualified name of Control File associated with the pending process.
Status	Status of the pending process.
Server	Name of server where Control File is located, or "Local" if Control File is located on workstation.
Source File Information	
File Name	Fully qualified name of Source File associated with the pending process.

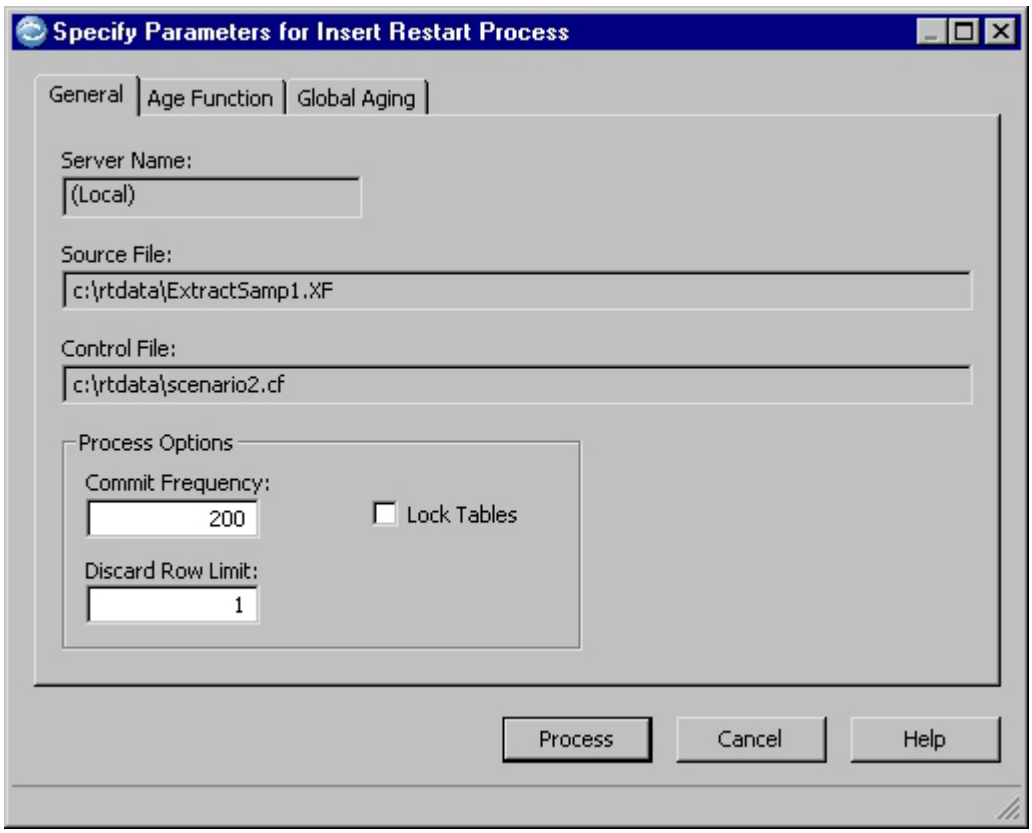
Timestamp	Date and time Source File was created.
SQLID	Identifier assigned to the process. **Note:** The SQLID for the Source File is displayed only if the Extract File or Archive File was created using Optim.
Last Process	
Process	Type of process that terminated abnormally or that contains unsuccessfully processed rows.
Timestamp	Date and time the process was run.
SQLID	Identifier assigned to the process that terminated abnormally or that contains unsuccessfully processed rows.

Restart or Retry a Pending Process

In the grid list, select the entry for the pending process you want to Restart or Retry, and select **Run** from the **File** menu or the shortcut menu.

Note: If you select a process that was run on a different machine, the files may be inaccessible from your machine. One or more files on your machine with the same names as those for the pending process, but generated by a different process, may cause unexpected results.

The Specify Parameters dialog is displayed. This dialog contains three tabs for Insert and Update/Insert processes – **General**, **Age Function**, and **Global Aging**. For the Delete Process, only the **General** tab is displayed.



General

The **General** tab displays the Server name and location and names of the Source File and Control File for the selected pending process. Process Options for the new processing attempt can be specified, as follows:

Process Options

Commit Frequency
Number of rows to process before committing the changes to the database. The Commit Frequency is a number from 1 to the maximum limit set as the default in Product Options (refer to the *Installation and Configuration Guide*). To use the default, clear the entry.

Note: Frequent commits keep page locks to a minimum, but may slow the process.

Discard Row Limit
Maximum number of rows (0 to 999999) that can be marked as discarded when you restart or retry a process. The process stops when the specified number of rows is discarded. To end the process if a single row is discarded, specify 1 as the maximum. To set no limit to the number of rows that can be discarded, specify 0 (zero).

Lock Tables
Select to lock database tables when you restart/retry a process. You can lock tables to ensure that other database activity does not interfere with the process attempt and prevent other users from accessing tables involved in the process. The default for locking tables is set by using Product Options (see the *Installation and Configuration Guide*).

Age Function

The date aging options allow you to adjust dates in specified columns and to verify application results on critical dates. See the *Move User Manual* for information on how to specify parameters on the **Age Function** tab.

Global Aging

You can specify options for Global Aging to adjust any columns that have a date data type. See the *Move User Manual* for information on how to specify parameters on the **Global Aging** tab.

Chapter 12. Calendars

Use the Calendar Utility to customize handling of dates for aging data and scheduling process requests. The Calendar Editor allows you to create and maintain Calendars that contain general specifications, special types of dates (for example, holidays, workdays, and weekends), and business rules to apply to those special types of dates.

Calendars are stored in the Optim Directory. The dates and rules in a Calendar facilitate the semantic aging of data and adjust the day a scheduled process request is executed, as required. For details on scheduling process requests, see Chapter 14, "Schedule," on page 283.

Semantic Aging

Optim uses the principles of Semantic Aging to advance dates relative to one another and in accordance with business rules. Semantic Aging adjusts dates to occur on valid business days. For example, you can increment all dates by three months. For many dates, this is sufficient. However, incrementing a date may result in a weekend day or a holiday. (Aging September 25th by three months yields December 25th, a standard holiday.) These weekend and holiday results must be adjusted to valid business days. This adjustment, known as semantic aging, is critical. Without it, the result is linear aging which cannot provide accurate test data in all cases.

To provide semantic aging, use the Calendar Editor to define special Calendar dates and rules to apply to the data you want to age. Once defined, you can use a Calendar with the Age Function in a Column Map or the global specifications in a process request (Convert, Insert or Load) to perform semantic aging.

Naming Conventions

The fully qualified name of a Calendar consists of a 1 to 12 character string. When you create Calendars, it is helpful to use a logical set of naming conventions to identify the use for each and to organize Calendars for easy access.

Contents

This section explains how to create and maintain Calendars including how to:

- Specify general default options and set up a Calendar year.
- Define special Calendar dates (e.g., holidays).
- Define business rules to adjust aged dates when necessary.

For details on using the Age Function in a Column Map, see Chapter 5, "Column Maps," on page 121. For details on how to specify date aging for a process request, refer to *Move User Manual* and *Archive User Manual* .

Open the Calendar Editor

Open the Calendar Editor to create a new Calendar or edit an existing Calendar.

Create a New Calendar

To create a new Calendar:

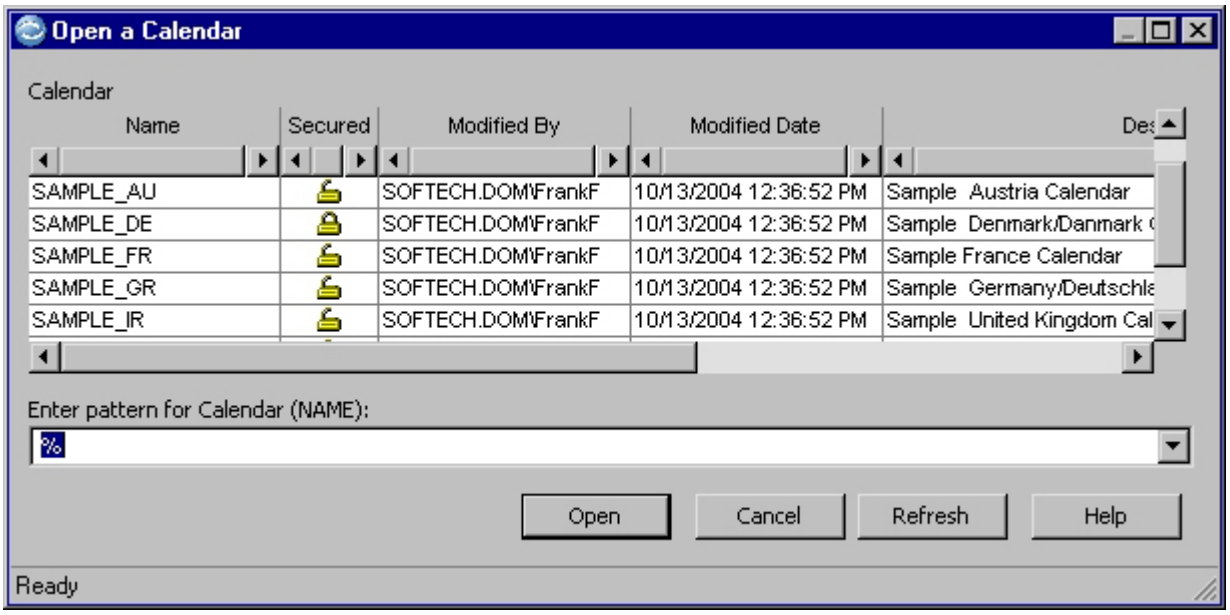
1. Select **Calendar** from the **Utilities** menu to open the Calendar Editor.
2. Select **New** from the **File** menu.
3. Specify the details on the **General** tab, **Dates** tab, and **Rules** tab.

4. Type a brief description of the Calendar (optional).
5. Select **Save** from the **File** menu.

These steps are the minimum required to create a Calendar. Because the options to create and edit a Calendar are similar, see "Using the Editor" on page 261 for complete details.

Open a Calendar Dialog

The Open a Calendar dialog lists the names of available Calendars and their details.



Grid Details

Name Name assigned to the Calendar.

Secured
Indicates whether the Calendar is secured (i.e., locked) or unsecured (i.e., unlocked).

Modified By
User ID under which the Calendar was created or last modified.

Modified Date
Date when the Calendar was created or last modified.

Description
Text that describes or explains the purpose of the Calendar.

Pattern

You can use a **Pattern** to limit the list of requests in the Open dialog. A Calendar name has one part: *name*. You can use the % (percent) wildcard to represent one or more characters or use the _ (underscore) wildcard to represent a single character in a Calendar name. (The underscore must be selected as the SQL LIKE character on the **General** tab in Personal Options.)

Note: After you specify a **Pattern**, select **Refresh** to redisplay the list based on your criteria.

Select a Calendar to Edit

To open a Calendar:

1. Select **Calendar** from the **Utilities** menu to open the Calendar Editor.
2. Select **Open** from the **File** menu to display the Open a Calendar dialog.
3. Double-click to select a Calendar and return to the Calendar Editor.

Using the Editor

When you open the Calendar Editor, the last Calendar you created or edited is displayed.

Tabs

Tabs on the Calendar Editor allow you to set up the Calendar; each tab serves a unique purpose:

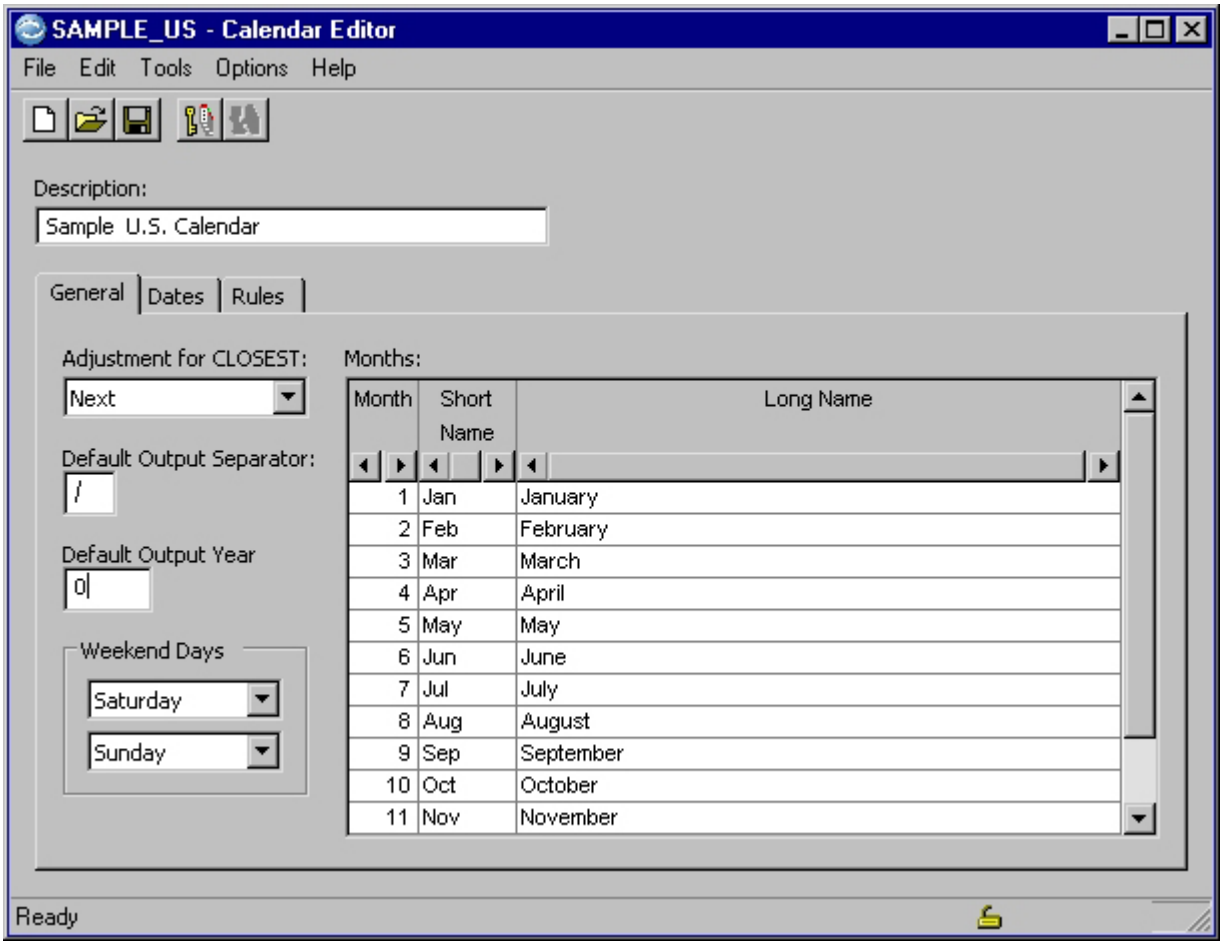
General
: Global information and default settings such as the names of the months, the days of the week that represent a weekend, the default output separator, and default output year. The **General** tab is always displayed first.

Dates
: Dates that are significant at your site.

Rules
: Rules available in the current Calendar.

General Tab

Use the **General** tab to specify the following default settings for date adjustments and formatting, as well as the months in the Calendar year:



Description

Text to describe the purpose of the Calendar (up to 40 characters).

Adjustment for Closest

Direction (**Next** or **Previous)** in which to adjust dates. This adjustment applies when you specify **Closest** in a Calendar rule, and the number of days on either side of the calculated date are equal.

Default Output Separator

Date separator used to format dates. This default applies when the calculated date requires separators, and the source column does not include separators, or when the date format for the destination column is not specified.

Default Output Year

Four-digit year, in the format YYYY (for example, 1999), to use as the default for formatting dates. If you specify zero (0), the current year is used as the default. This default applies when a calculated date requires a year and the source column does not specify one, or when the format for the destination column does not include a year.

Weekend Days

Day or days of the week that represent weekends in the Calendar year. You can specify one or two days for a weekend. The days do not have to be consecutive. Any day of the week that is not defined as a weekend is considered a workday.

Grid Details

Month
Number to identify a month in the Calendar year.

Short Name
Abbreviated name for a month in the Calendar year.

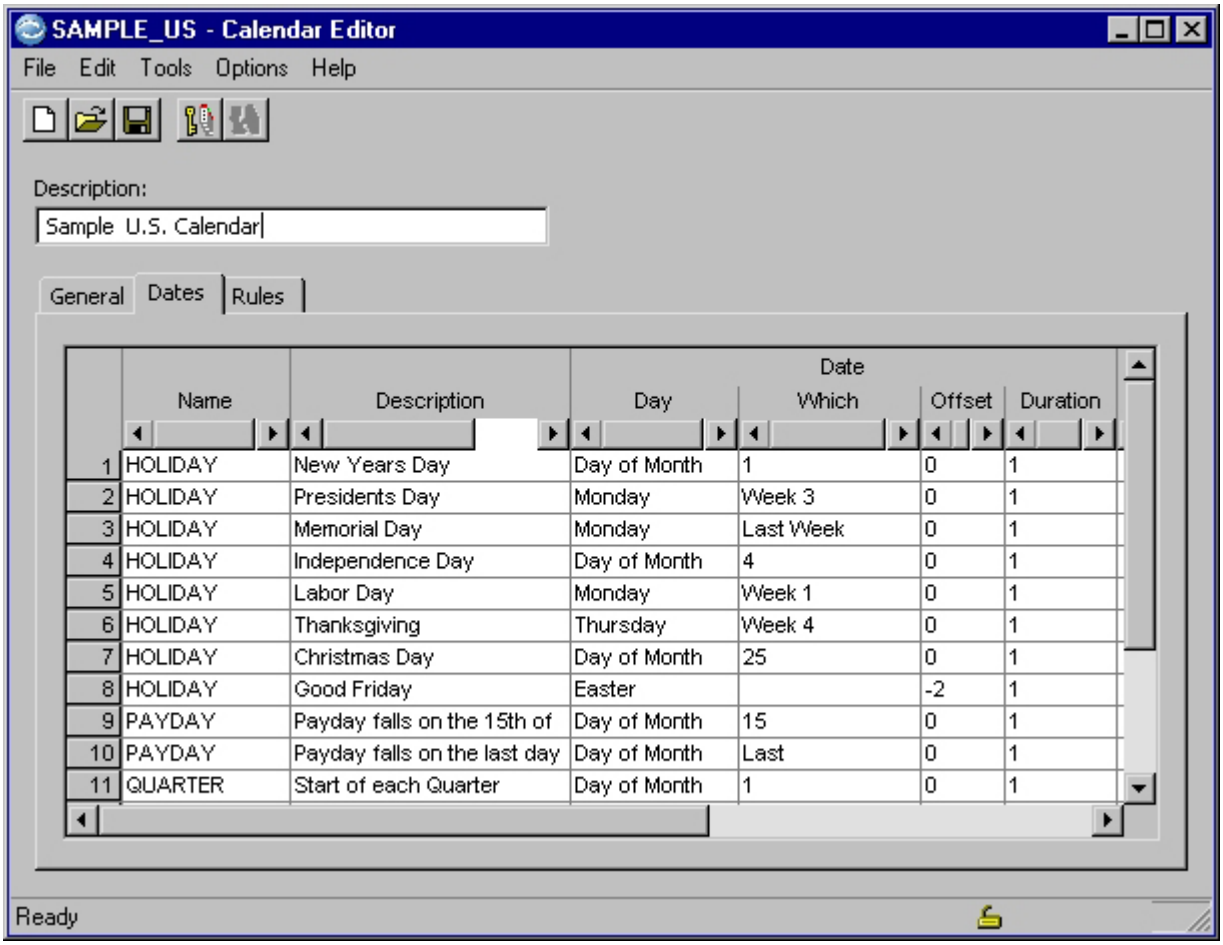
Long Name
Full name of a month in the Calendar year.

Dates Tab

You can define and name up to 14 unique categories to describe special Calendar dates, such as holiday, payday, quarter, etc. Each name may be specified as many times as required to identify the special day. (For example, in the distributed US Calendar, there are 8 days predefined as HOLIDAY.)

Select the **Dates** tab on the Calendar Editor to:

- Define Calendar dates that have significant meaning at your site, for example, holiday, workday, payday or other.
- Specify the exact day, number of offset days, and duration of each Calendar date.
- Indicate how to resolve differences when an aged date or scheduled process request occurs on the defined Calendar date.
- Specify how often each Calendar date can reoccur during the year.



Grid Details

Name A category to which the Calendar date applies (1 to 8 characters).

- You can specify up to 14 unique names.
- You can use the same name, with different characteristics, any number of times.

Description

Text to describe or explain the Calendar date (up to 40 characters).

Date/Day

A name to identify the day for the Calendar date. Click the down arrow to select from a list of the following:

- Day of the week
- Easter
- Workday
- Day of the Month

Date/Which

The frequency of the **Date/Day** you enter. To select from a list of valid entries, click to display a down arrow, then click the down arrow. For example:

If you enter Date/Day:

Specify:

Day of the week

Every Week, Week 1, Week 2, Week 3, Week 4, or Last Week

Workday
Every, an explicit day by number 1 through 25, or Last

Day of the Month
Every, an explicit day by number 1 through 31, or Last

Easter No entry required.

Note:

- Every indicates every instance of the specific day, for recurring dates.
- Last indicates the last instance of the specific day.
- Workday indicates any day not specifically defined as a Weekend or Holiday.
- Easter calculates the specific day based on Easter Sunday, using a plus (+) or minus (-) offset

.

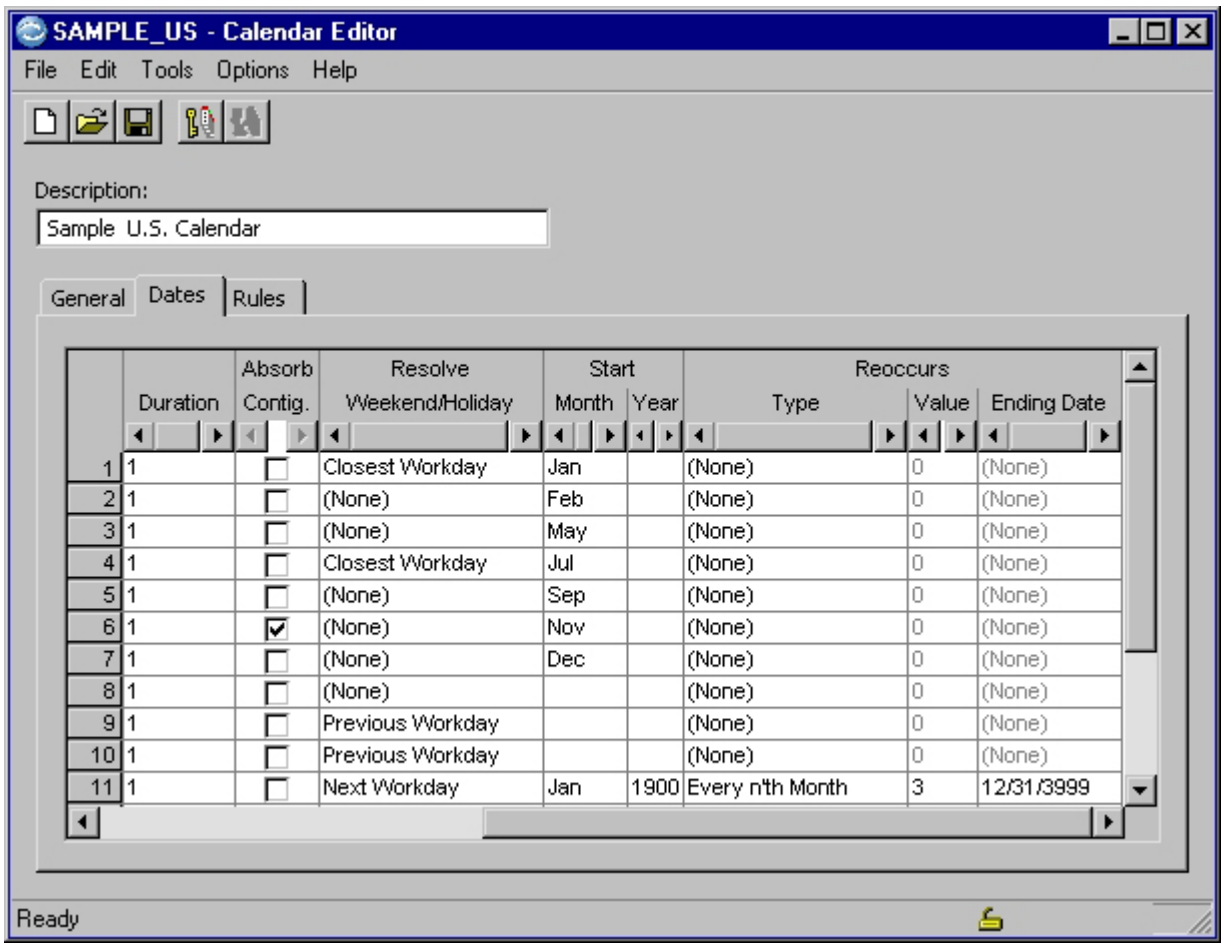
Date/Offset
The number of days (1 to 366) to adjust the Calendar date. Use a minus (-) sign to decrement the date, or a plus (+) sign to increment the date.

Note: The Date/Offset value entered must not cause the adjusted date to occur in a different year. The value you specify must be less than or equal to the number of days remaining in the year (if incremental), or the number of days since the beginning of the year (if decremented).

Date/Duration
The duration of the Calendar date, in number of days (1 to 366).

Note: The Date/Duration value entered must not cause any portion of the Calendar date to occur in a different year. The value you specify must be less than or equal to the number of days remaining in the year (if incremental), or the number of days since the beginning of the year (if decremented).

Scroll the grid horizontally to the right to display the following details:



Absorb Contig

Select this check box to include Monday when the specified Calendar date occurs on a Tuesday, or to include Friday when the specified Calendar date occurs on a Thursday. The check box is cleared by default.

Resolve Weekend/Holiday

The adjustment parameter to use when the Calendar date occurs on a weekend or holiday. Specify None if no adjustment is required. Click the down arrow to select from the following list:

- None
- Closest Workday
- Previous Workday
- Next Workday

Start/Month

The month to start using the Reoccurs (frequency) settings for the Calendar date.

Start/Year

The year to start using the Reoccurs (frequency) settings for the Calendar date.

Reoccurs/Type

The frequency of the Calendar date. (For example, payday may occur every two weeks, regardless of the date.) Click the down arrow to select from a list, where **n** is the number for each unit of time. "(None)" indicates the Calendar date does not reoccur.

Reoccurs/ Value

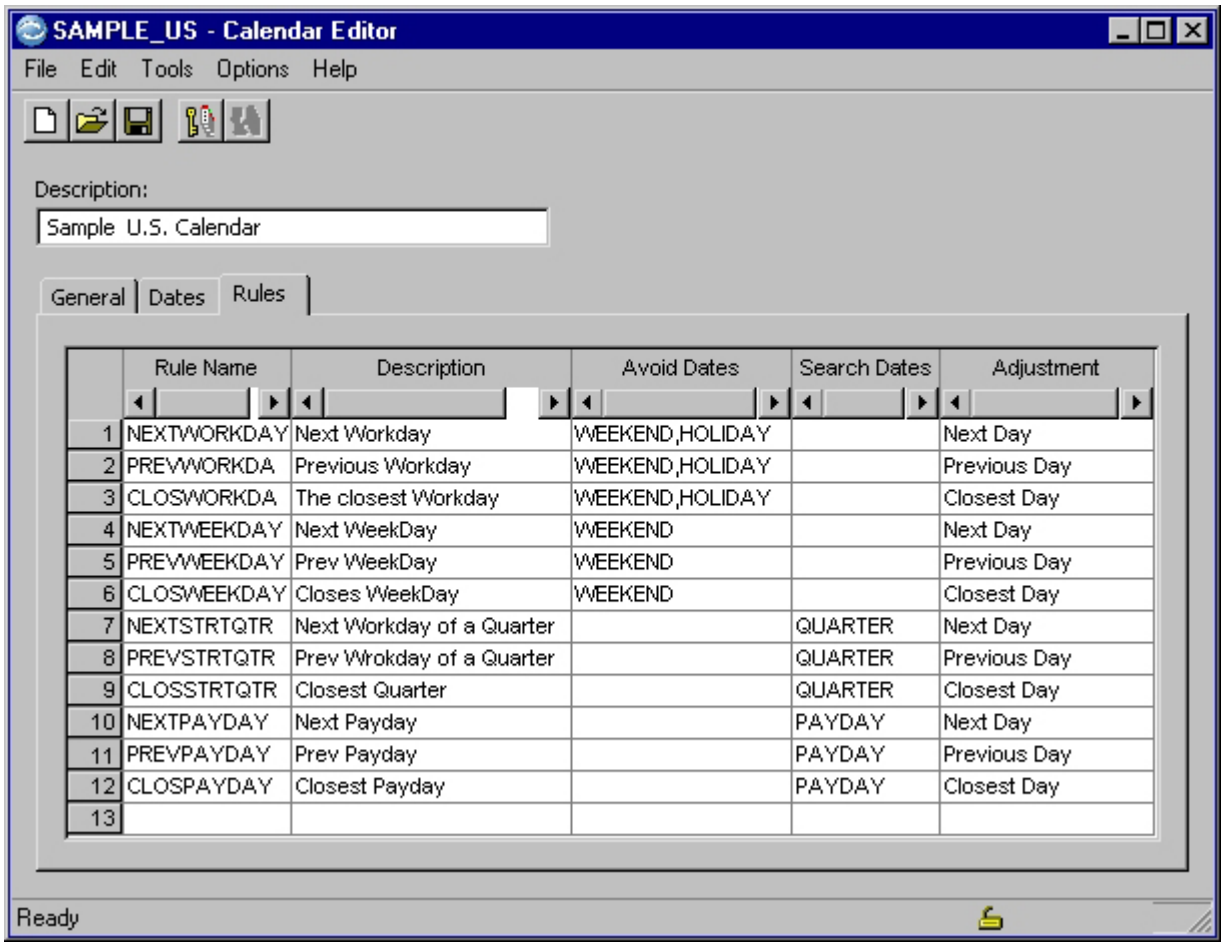
The number for the **nth** unit of time specified in the Type of frequency. You can specify a value only when Type is not "(None)".

Reoccurs/ Ending Date
The date that the frequency specifications are to be discontinued. You can specify a value only when Type is not "(None)". The default ending date is 12/31/3999.

For a list of sample Calendar dates provided with Optim, see "Sample Calendar Dates" on page 268.

Rules Tab

Use the **Rules** tab to define rules used to adjust dates when aging data or scheduling process requests. Dates in the Avoid Dates column signify that other dates are acceptable. Dates in the Search column signify that only those specified dates are acceptable, and all other dates are unacceptable.



Grid Details

Rule Name
Name of the Calendar rule you want to define (1 to 11 characters).

Note: Optim includes several sample rules that you can use. There is no limit to the number of rules you can define.

Description
Text to describe the Calendar rule you want to define (up to 40 characters).

Avoid Dates
Dates to avoid when making an adjustment. Specify any date type defined on the **Dates** tab or specify one of the sample date types provided.

For example, if the rule is NEXTWEEKDAY, the Avoid column contains WEEKEND, and the Adjustment column specifies NEXTDAY, the calculated date is adjusted to the next day that is not defined as a weekend day.

Note: For each rule, you can specify an Avoid Date, a Search Date, or both. If you specify both, the entries must be different Calendar dates.

Search Dates

Dates to search for when making an adjustment. Specify any date type defined on the **Dates** tab or specify one of the sample date types provided.

For example, if the rule is NEXTPAYDAY, the Search column contains PAYDAY, and the Adjustment column specifies NEXTDAY, the date is adjusted to the next defined payday. (If the calculated date is March 5th and payday is defined as the 1st and 15th of each month, the date is adjusted to March 15th.)

Note: For each rule, you can specify an Avoid Date, a Search Date, or both. If you specify both, the entries must be different Calendar dates.

Adjustment

Specify the adjustment parameter to use to adjust the date for the named rule. The adjusted date must satisfy the Avoid and Search specifications.

- Closest Day
- Previous Day
- Next Day

For a list of sample Calendar rules provided with Optim, see "Sample Calendar Rules" on page 270.

Sample Calendar Dates

The Calendar Utility provides a sample Calendar, SAMPLE_US, containing the Calendar dates for standard United States holiday, workday, and quarter dates.

Sample Holidays

The following sample holiday dates are defined in the sample Calendar:

Calendar Date	Specify	Details
Presidents Day	Description	'Presidents Day'
	Day	Monday
	Which	Week 3
	Month	February
Memorial Day	Description	'Memorial Day'
	Day	Monday
	Which	Last
	Month	May
Independence Day	Description	'Independence Day'
	Day	Day of Month
	Which	4
	Month	July
	Resolve	Closest Workday
Labor Day	Description	'Labor Day'

Calendar Date	Specify	Details
	Day	Monday
	Which	Week 1
	Month	September
Thanksgiving Day	Description	'Thanksgiving Day'
	Day	Thursday
	Which	Week 4
	Month	November
	Duration	2
Christmas Day	Description	'Christmas Day'
	Day	Day of Month
	Which	25
	Month	December
Good Friday	Description	'Good Friday'
	Day	Easter
	Offset	2

Sample Business Days

The following sample business dates are defined in the sample Calendar:

Calendar Date	Specify	Details
Payday on the 15th	Description	'Payday on the 15th of the Month'
	Day	Day of Month
	Which	15
	Resolve	Previous Workday
Payday on Last Workday of a Month	Description	'Payday on the Last Workday of the Month'
	Day	Day of Month
	Which	Last
	Resolve	Previous Workday
First Quarter	Description	'First Quarter'
	Day	Day of Month
	Which	1
	Month	January
	Resolve	Next Workday
Second Quarter	Description	'Second Quarter'
	Day	Day of Month
	Which	1
	Month	April
	Resolve	Next Workday
Third Quarter	Description	'Third Quarter'
	Day	Day of Month

Calendar Date	Specify	Details
	Which	1
	Month	July
	Resolve	Next Workday
Fourth Quarter	Description	'Fourth Quarter'
	Day	Day of Month
	Which	1
	Month	October
	Resolve	Next Workday

Sample Calendar Rules

The Calendar Utility provides a sample Calendar, SAMPLE_US, that contains the following Calendar rules for standard business adjustments, based on Saturday and Sunday defined as weekend days.

NEXTWORKDAY

When a date falls on a weekend day or a holiday, adjust the date to the next workday.

Description
'Next Workday'

Avoid WEEKEND,HOLIDAY

Adjustment
Next Day

For example, when an aged date falls on a Saturday or Sunday, the date is adjusted to the following Monday, but if Monday is a holiday, the date is adjusted to Tuesday (provided Tuesday is not a holiday and so on).

PREVWORKDAY

When a date falls on a weekend day or a holiday, adjust the date to the previous workday.

Description
'Previous Workday'

Avoid WEEKEND,HOLIDAY

Adjustment
Previous Day

For example, when an aged date falls on a Saturday or Sunday, the date is adjusted to the previous Friday, but if Friday is a holiday, the date is adjusted to Thursday (provided Thursday is not a holiday and so on).

CLOSWORKDAY

When a date falls on a weekend or holiday, adjust the date to the closest workday.

Description
'Closest Workday'

Avoid WEEKEND,HOLIDAY

Adjustment
Closest Day

For example, when an aged date falls on a Saturday, the date is adjusted to the previous Friday. If the date falls on a Sunday, the date is adjusted to the following Monday. If the date falls on a holiday, the date is adjusted to the closest weekday.

NEXTWEEKDAY

When a date falls on a weekend day, adjust the date to the next weekday.

Description
'Next Weekday'

Avoid WEEKEND

Adjustment
Next Day

For example, when an aged date falls on a Saturday or Sunday, the date is adjusted to the following Monday.

PREVWEEKDAY

When a date falls on a weekend day, adjust the date to the previous weekday.

Description
'Previous Weekday'

Avoid WEEKEND

Adjustment
Previous Day

For example, when an aged date falls on a Saturday or Sunday, the date is adjusted to the previous Friday.

CLOSWEEKDAY

When a date falls on a weekend day, adjust the date to the closest weekday.

Description
'Closest Weekday'

Avoid WEEKEND

Adjustment
Closest Day

For example, when an aged date falls on a Saturday, the date is adjusted to the previous Friday. If the date falls on Sunday, the date is adjusted to Monday.

NEXTSTRTQTR

When a date does not fall on a date defined as the start-of-quarter day, adjust the date to the next quarter.

Description
'Next Workday after Quarter'

Avoid WEEKEND, HOLIDAY

Search
QUARTER

Adjustment
Next Day

For example, when a calculated date is March 5th, the date is adjusted to April 1st (the start of the next quarter). If April 1st is a weekend or holiday, the date is adjusted again to the next following weekday.

PREVSTRTQTR

When a date does not fall on a date defined as the start-of-quarter day, adjust the date to the previous quarter.

Description
'Previous Workday before Quarter'

Avoid WEEKEND, HOLIDAY

Search
QUARTER

Adjustment
Previous Day

For example, when a calculated date is March 5th, the date is adjusted to January 1st (the start of the previous quarter). If January 1st is a weekend or holiday, the date is adjusted again to the next following weekday.

CLOSSTRTQTR

When a date does not fall on a date defined as the start-of-quarter day, adjust the date to the closest quarter.

Description
'Closest Workday after Quarter'

Avoid WEEKEND, HOLIDAY

Search
QUARTER

Adjustment
Closest Day

For example, when a calculated date is March 5th, the date is adjusted to April 1st. If the date is January 15th, the date is adjusted to January 1st. If the first day of the quarter is not a workday, the date is adjusted again to the closest weekday.

NEXTPAYDAY

When a date does not fall on a date defined as a payday, adjust the date to the next payday.

Description
'Next Payday'

Search
PAYDAY

Adjustment
Next Day

For example, when a calculated date is March 5th and payday is defined as the 1st and the 15th day of each month, the date is adjusted to March 15th.

PREVPAYDAY

When a date does not fall on a date defined as a payday, adjust the date to the previous payday.

Description
'Previous Payday'

Search
PAYDAY

Adjustment
Previous Day

For example, when the date is March 5th and payday is defined as the 1st and the 15th day of each month, the date is adjusted to March 1st.

CLOSPAYDAY

When a date does not fall on a date defined as a payday, adjust the date to the closest payday.

Description
'Closest Payday'

Search
PAYDAY

Adjustment
Closest Day

For example, when a calculated date is March 5th and payday is defined as the 1st and the 15th day of each month, the date is adjusted to March 1st, the closest payday.

Chapter 13. Currency

Use the Currency Utility to create, browse, edit and delete Currency Definitions. Currency Definitions provide currency conversion parameters specified in a Column Map, including conversion rates and conversion method (direct conversion or triangulation via the euro dollar).

The Currency Editor displays a Currency Definition in a three-tabbed dialog to allow you to specify general conversion information parameters, conversion rates and currency types. You can create date⌂specific rate tables to allow transaction dates to be specified as part of the currency conversion parameters (the current date is used by default). You can also define Types Tables within a Currency Definition to match currency types defined in a database with currency types defined in the rate tables. Each time a currency transaction is performed, the source and destination currency types must be specified explicitly or defined in a specified Types Table.

Naming Conventions

The fully qualified name of a Currency Definition consists of a 1 to 12 character string. When you create Currency Definitions, it is helpful to use a logical set of naming conventions to identify the use for each and to organize Currency Definitions for easy access.

Contents

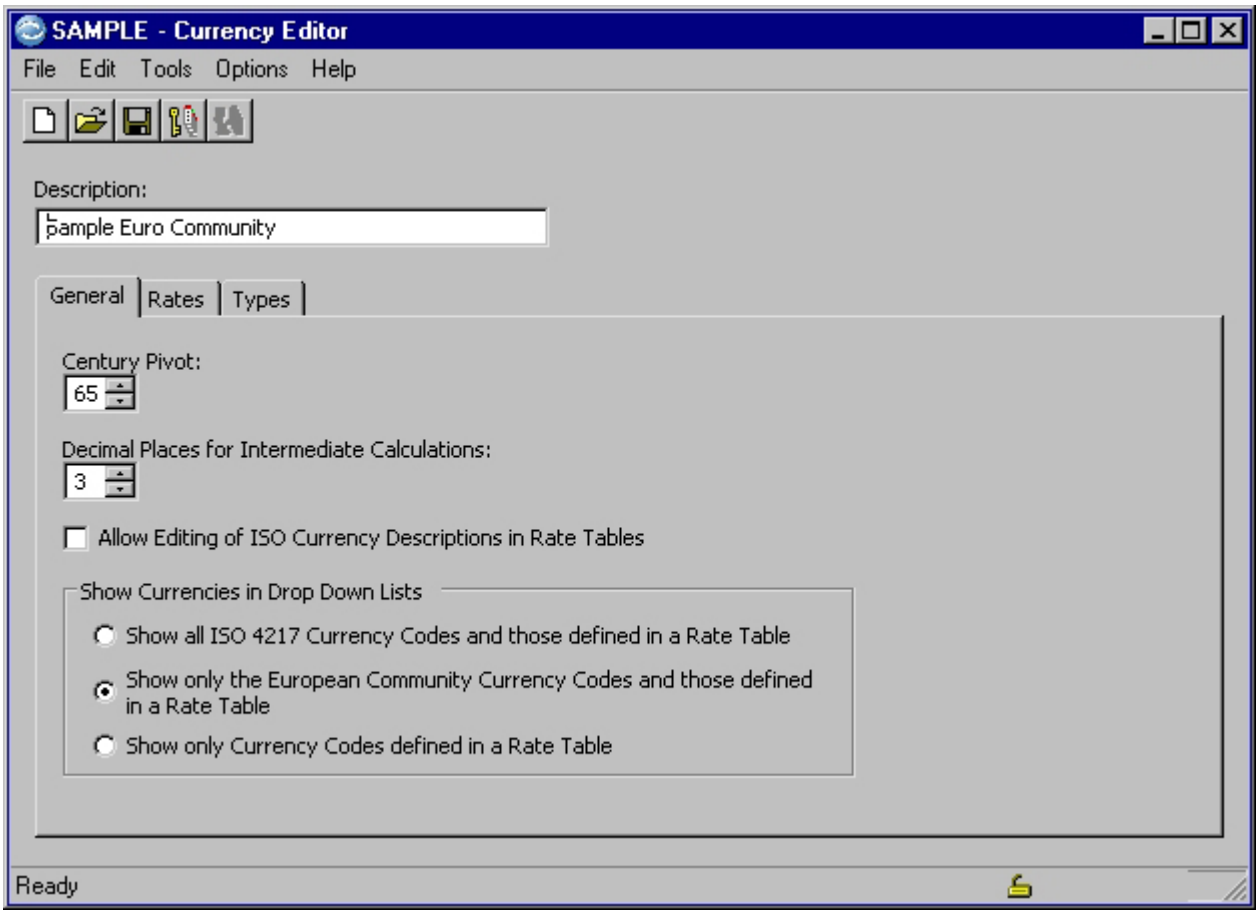
This section explains how to create and maintain Currency Definitions, including how to:

- Specify general default options and set up a Currency Definition.
- Define currency rates.
- Define date ranges for conversion rates, as required.
- Define codes for conditional determination of currency types.
- Define the method of conversion.

For details on specifying currency conversion parameters in a Column Map, see Chapter 5, “Column Maps,” on page 121.

Using the Currency Editor

Select **Currency** from the **Utilities** menu to open the Currency Editor. The **General** tab is always displayed first.



Description

Text to describe the Currency parameters specified for the Currency Definition to distinguish it from other Currency Definitions (up to 40 characters).

Tabs

The Currency Editor displays three tabs. Each tab serves a unique purpose:

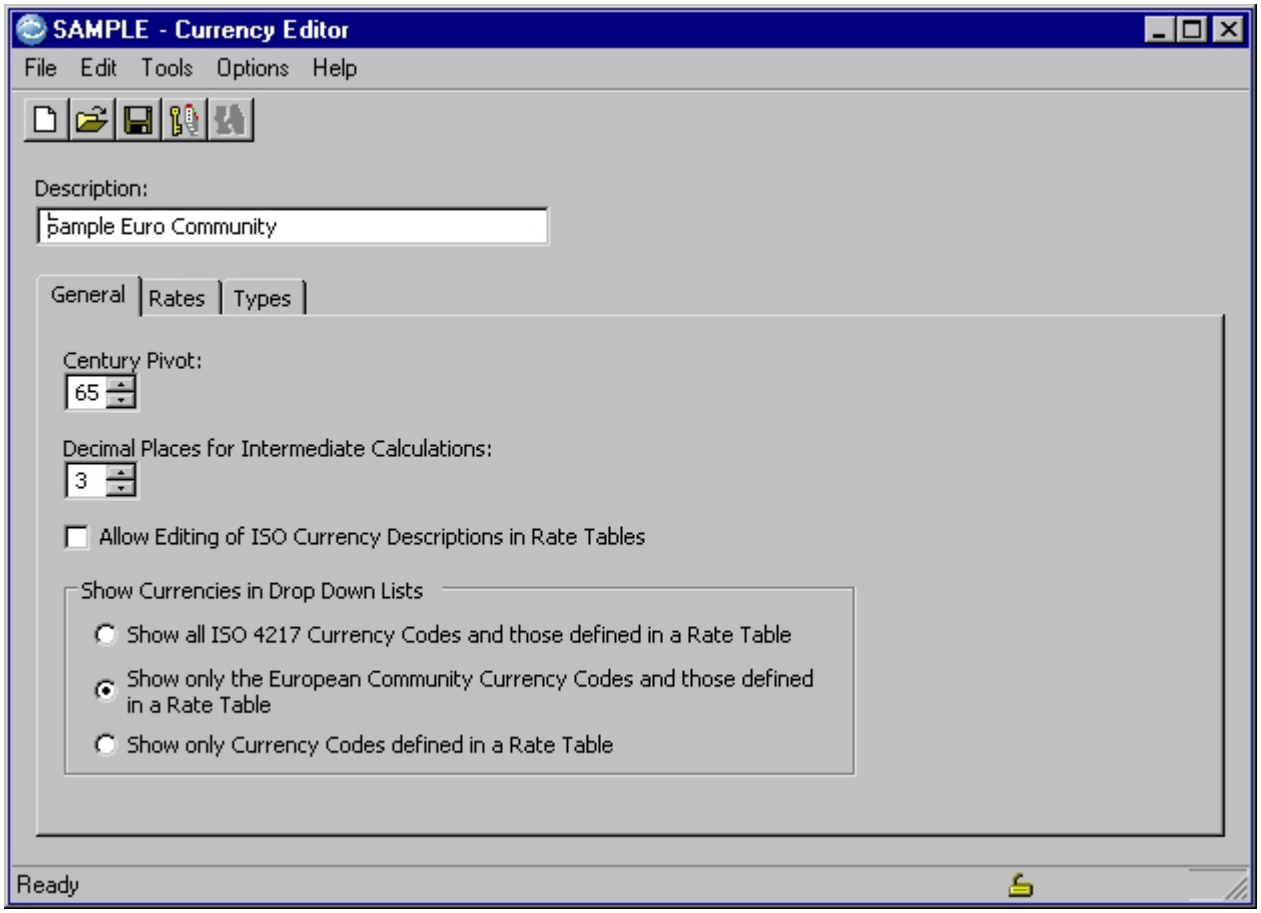
General
The Century Pivot specification, the specification for the number of decimal places to be retained, and editing and display parameters.

Rates The rates and effective time frame to be used for currency conversion.

Types Types Tables define codes used in a database to represent types of currency. You can create a maximum of four tables.

General Tab

Use the **General** tab to establish the Century Pivot specification, the specification for the number of decimal places to be retained, and editing and display parameters.



Century Pivot

The value used to determine the appropriate century when the Currency function in a Column Map includes a transaction date with a two-digit year. If you do not specify a Century Pivot, 65 is the default. For example, if you specify 55 as the default Century Pivot:

- All two-digit years equal to or greater than 55 are assumed to be in the 20th century.
- All two-digit years less than 55 are assumed to be in the 21st century.

Decimal Places...

The number of decimal places to be retained for the intermediate value when triangulation is performed.

Allow Editing of ISO...

Select this check box to allow editing of the ISO currency descriptions on the **Rates** tab.

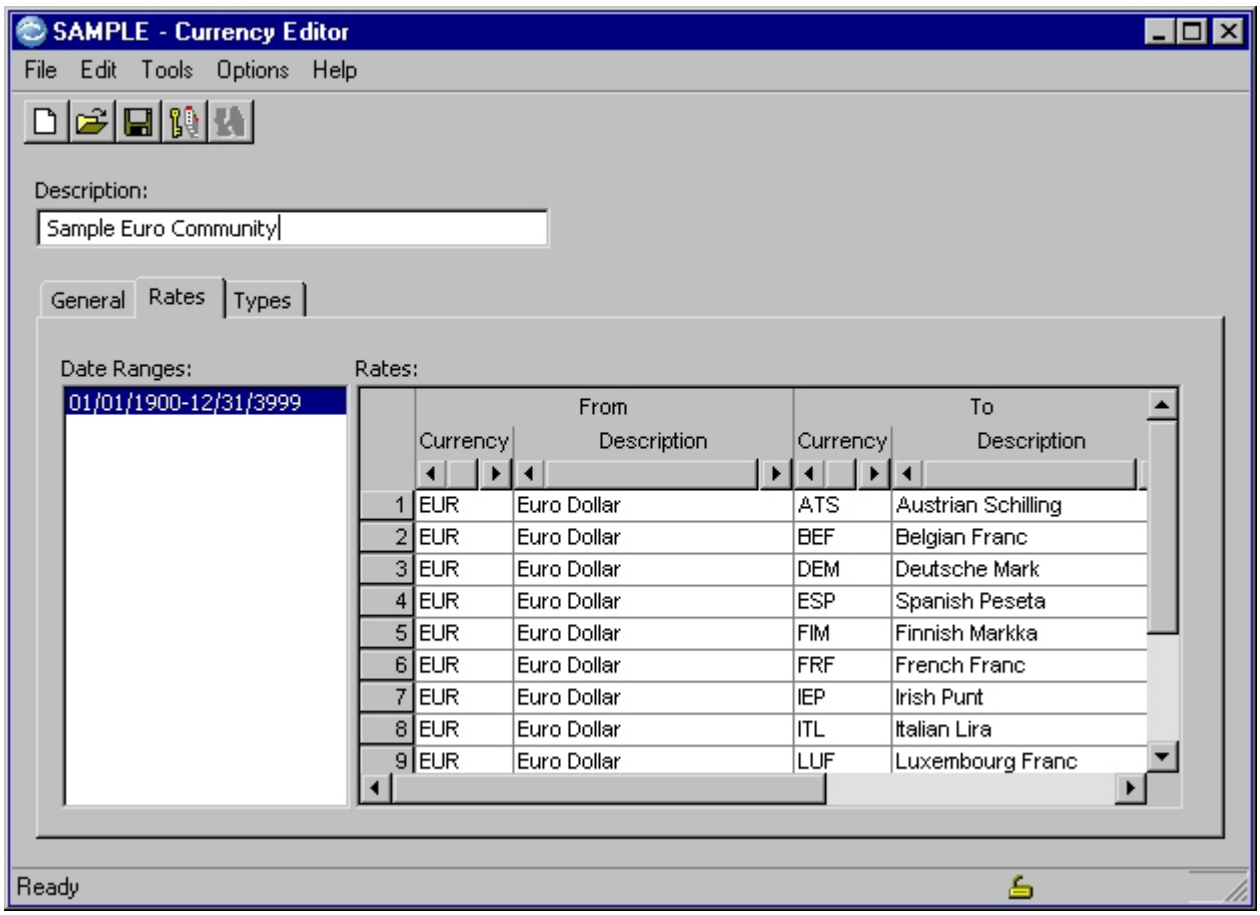
Show Currency...

Display options. Select:

- Show all ISO 4217 Currency Codes and Currency Codes in the Rate Tables.
- Show European Community Currency Codes and Currency Codes in the Rate Tables only.
- Show Currency Codes defined in the Rate Tables only.

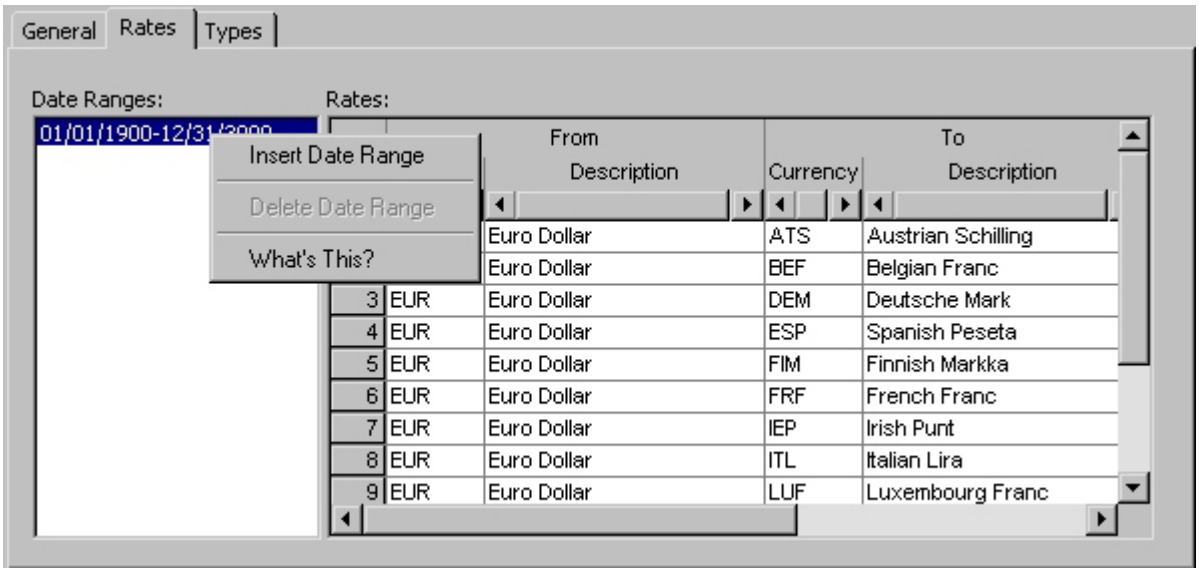
Rates Tab

Use the **Rates** tab to enter the rates used to convert currencies, and the effective time frames for the rates.



Date Ranges

Enter the effective date range for the specified currency conversion rates. To modify the date ranges, right-click to select from the shortcut menu. Assign date ranges to accommodate periodic fluctuations of currency rates. Date ranges must not overlap.



Rates

From

Currency
The source currency for the conversion rate specification. Enter a currency code or click the down arrow to select from a list of currencies.

Description
The description of the currency code.

To

Currency
The destination currency for the conversion rate specification. Enter a currency code or click the down arrow to select from a list of currencies.

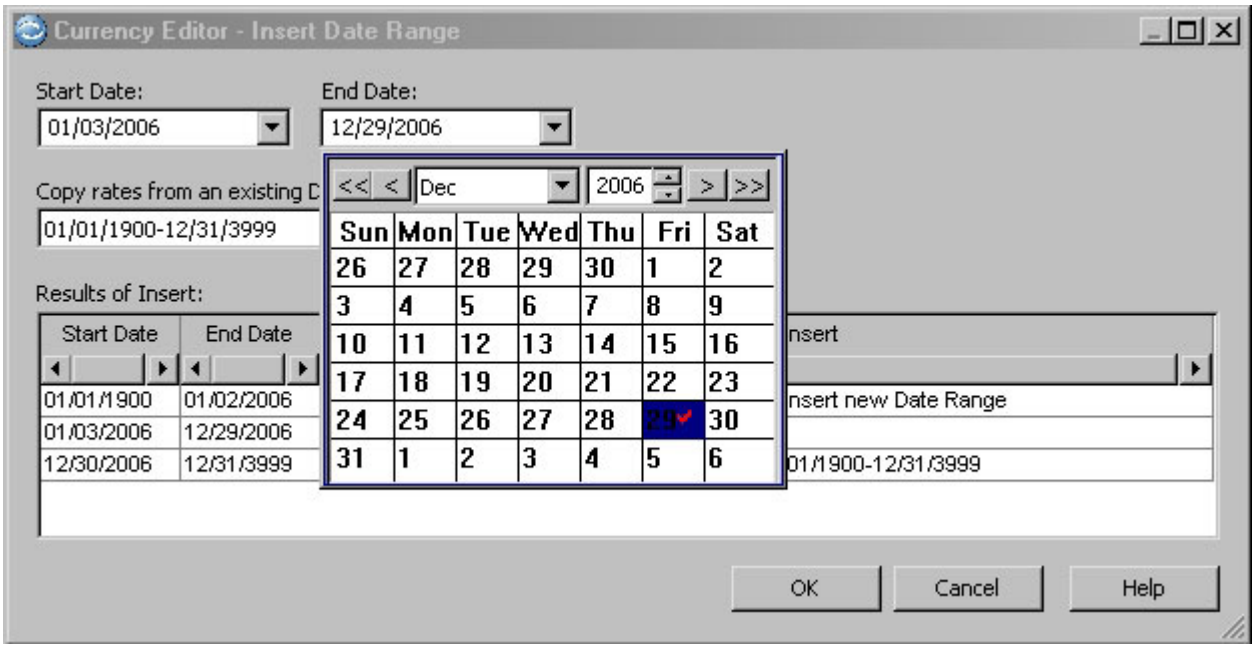
Description
The description of the currency code is displayed.

Rate A multiplier used to convert from the source currency to the destination currency, in each date range.

Note: You can specify zero (0) to cause a deliberate runtime error. Use a zero rate to purposely prevent processing currency conversions for a particular From/To currency within a specific date range.

Insert Date Range

Right-click in the Date Ranges group box and select **Insert Date Range** from the shortcut menu to display the Insert Date Range dialog. Enter the Start Date and End Date for the new date range. The existing date ranges are automatically adjusted to include the new range.

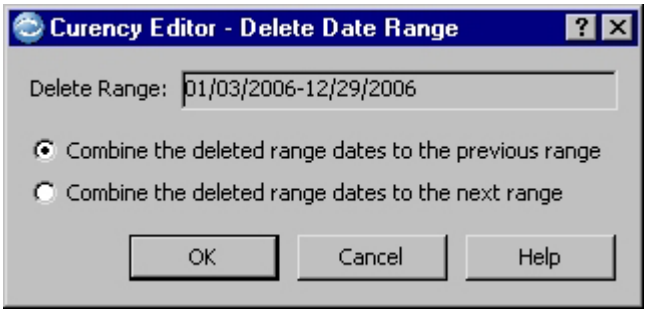


Start Date
Click the arrow to select a Start Date for the new date range.

End Date
Click the arrow to select an End Date for the new date range.

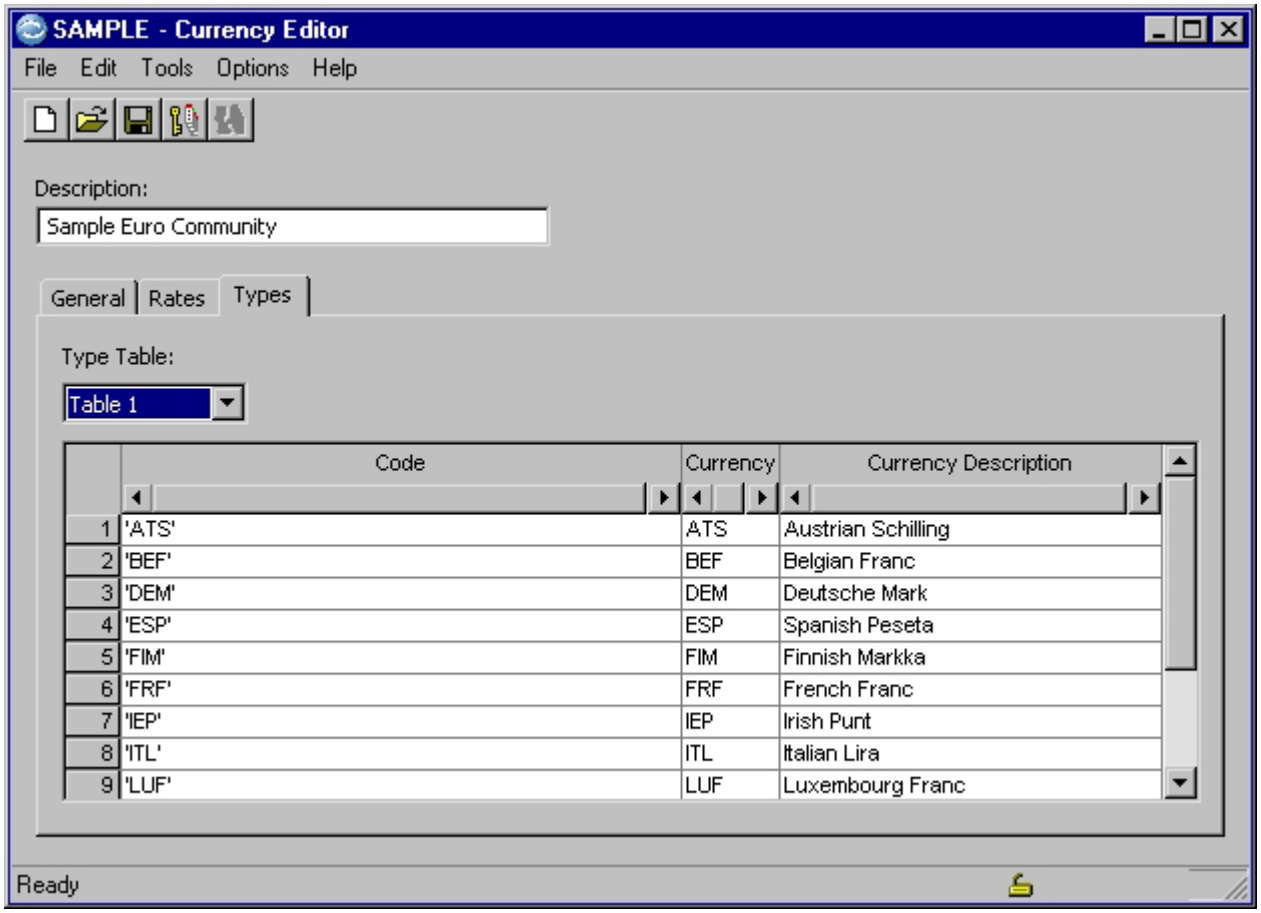
Copy rates from an existing ...
Click the arrow to select a previously-defined date range.

Delete Date Range
Right-click the date range and select **Delete Date Range** from the shortcut menu. Select an option in the Delete Date Range dialog to combine the deleted range with the previous range or the next range.



Types Tab

Use the **Types** tab to create and maintain as many as four Types Tables. Types Tables match currency types or codes in database tables with currencies on the **Rates** tab. Four empty tables, labeled 1 through 4, are distributed with Optim.



Type Table

Click the arrow to select one of four Types Tables provided.

Grid Details

Code The code used in a database table to represent a currency type. Define codes to correspond to currencies on the **Rates** tab of the Currency Definition. Enter the code in one of the following formats:

- String literal, in single quotes: *'alphanumeric'*
- Hexadecimal literal: X*'1234567890ABCDEF'*
- Hexadecimal literal: 0X*1234567890ABCDEF*
- Numeral: *12345*

Currency
The ISO 4217 Currency Code or user-defined Currency Code from the **Rates** tab that corresponds to the code used in the database table to represent a currency type.

Currency Description
The description of the Currency Code as shown on the **Rates** tab is displayed.

Sample Rate Table

The following sample currency rate table is defined with several rate entries (the rates are fictitious):

From	To	Rate
EUR	BEF	2.217436
EUR	FIM	4.323233
EUR	FRF	7.261892
GBP	USD	1.563321
USD	CHF	0.813
CHF	USD	1.252

Note that this table provides conversion rates between the euro dollar and three national currencies. When specifying a conversion rate involving the euro dollar, it is only permissible to specify the rate in terms of the quantity of currency for a nation equals a single euro dollar. This rate is then used as a multiplier or divisor (as appropriate) to perform all calculations.

As the two entries between USD (US Dollars) and CHF (Swiss Francs) demonstrates, individual rates may be specified when dealing directly with currencies other than the euro dollar. If a rate is provided for converting in a single direction only, the rate will be used as a multiplier or divisor (as appropriate) to perform all calculations.

Triangulation is used as a method for converting currencies via the euro dollar only. The table above permits conversion from Belgian Francs (BEF) to French Francs via triangulation with the euro dollar. Conversely, conversion from British Pounds (GBP) to Swiss Francs (CHF) via triangulation with the US Dollar (USD) is not supported.

Chapter 14. Schedule

Use the Scheduler to schedule process requests, edit the schedule, and monitor processing for Optim. The Scheduler allows you to run process requests at a later time, periodically, or on a regular schedule.

You can schedule an individual process request or several process requests together in one processing job. For example, you can automatically refresh test data on a regular basis by creating a scheduled job that includes an Extract Request and an Insert Request or you can archive data on a monthly or quarterly basis by creating an Archive Request that uses macros to name the files that are generated each time and selects data for archiving on the basis of generic criteria that can be used without change.

The Scheduler has three major components:

Job Details Dialog
When you create a process request, you can run the request immediately, or you can save and schedule the request for processing at a later time. Select **Schedule** from the process request editor **File** menu to display the Job Details dialog. Use the Job Details dialog to specify parameters for scheduling one or more process requests to run at a specified future time.

Scheduling Editor
After you schedule a process request, you can edit the schedule by selecting **Schedule** from the **Utilities** menu in the main window. When you select a job from the list, you can review and edit general details, edit specifications for regular or periodic processing, and add, modify, or delete the number of steps (process requests) included in each job.

Scheduling Monitor
After you schedule a process request, you can start the Scheduler to monitor processing. The Scheduling Monitor allows you to review a list of the scheduled jobs, monitor active jobs as they are being processed, and review the results of processing as it occurs. When you select a job from the list, you can review and edit general details, edit specifications for regular and periodic processing, and modify the number of steps (process requests) included in each job.

Contents

This section explains how to schedule a process request, including how to:

- Schedule a process request from any process request editor.
- Arrange for more than one process request to be scheduled to run as a multi-step scheduled job.
- Edit the processing schedule.
- Monitor scheduled processing.

After processing completes, you can review and print the results from the Process Report dialog.

Schedule a Process Request

From the Archive, Restore, Insert, Extract, Convert and Load Process Request Editors, you can run a request immediately or schedule a process request to run at a later time. To schedule a request to run at a later time, select **Schedule** from the **File** menu of the process request editor to display the Job Details dialog.

To schedule a process request:

1. Select a command from the **Actions** menu to create and save a process request.
2. Select **Schedule** from the **File** menu on the process request editor to open the Job Details dialog.

3. On the **General** tab, specify a start date, start time, and latest start time.
4. Click **OK** to return to the process request editor.
5. Select **Close** from the **File** menu on the process request editor.

These steps are the minimum required to schedule a process request. In addition, you may specify details on the **Repeats** tab and the **Steps** tab of the Job Details dialog. Each of these tabs is described in the following section.

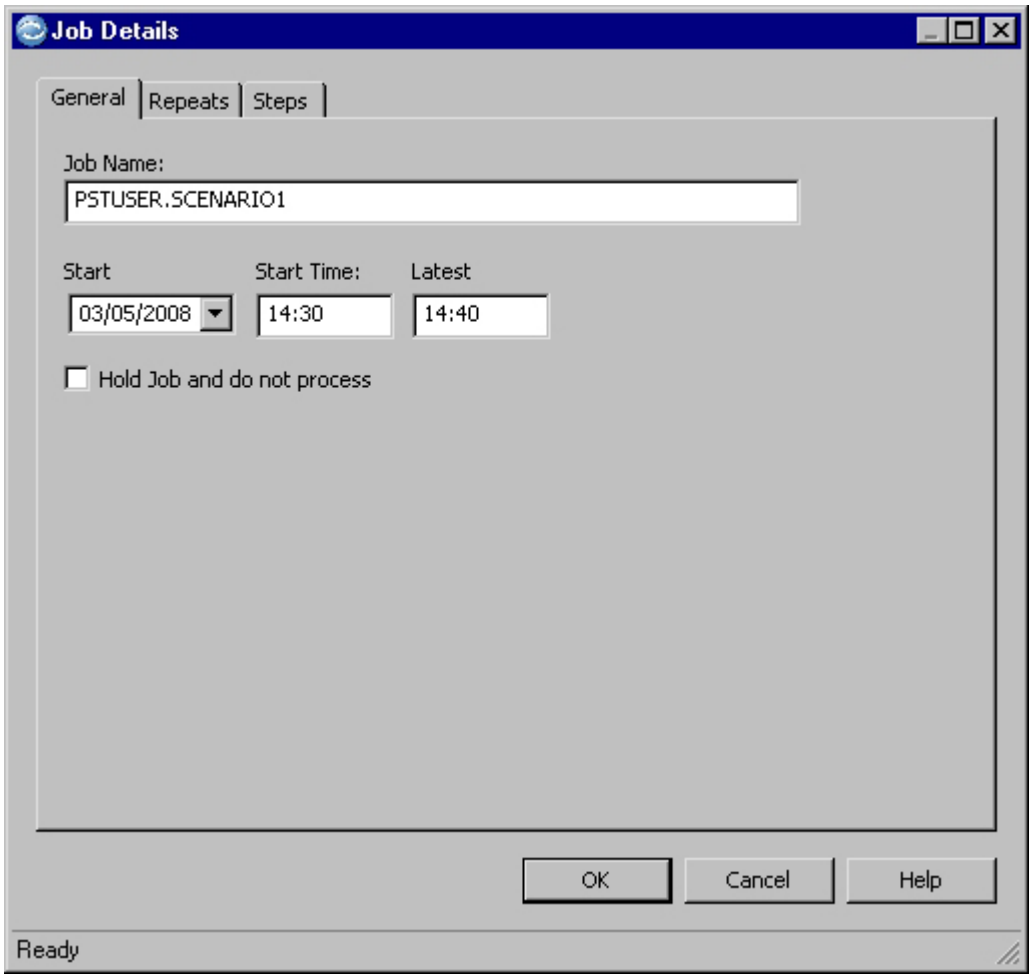
Note: The **General**, **Repeats**, and **Steps** tabs of the Job Details dialog are available from the process request editors, the Scheduling Editor, and the Scheduling Monitor.

Job Details Dialog

The Job Details dialog contains three tabs. Specify the parameters required to schedule one or more process requests to run as a scheduled job.

General

Use the **General** tab in the Job Details dialog to provide a description of the scheduled job and specify when you want processing to begin.



Details

Job Name
Enter a descriptive name to identify the scheduled job. (1 to 30 characters).

Start Enter the date the scheduled job is to begin. To open a calendar to select the start date, click the down arrow.

Note: The calendar is similar to that on the **Repeats** tab; see "Repeats" for details.

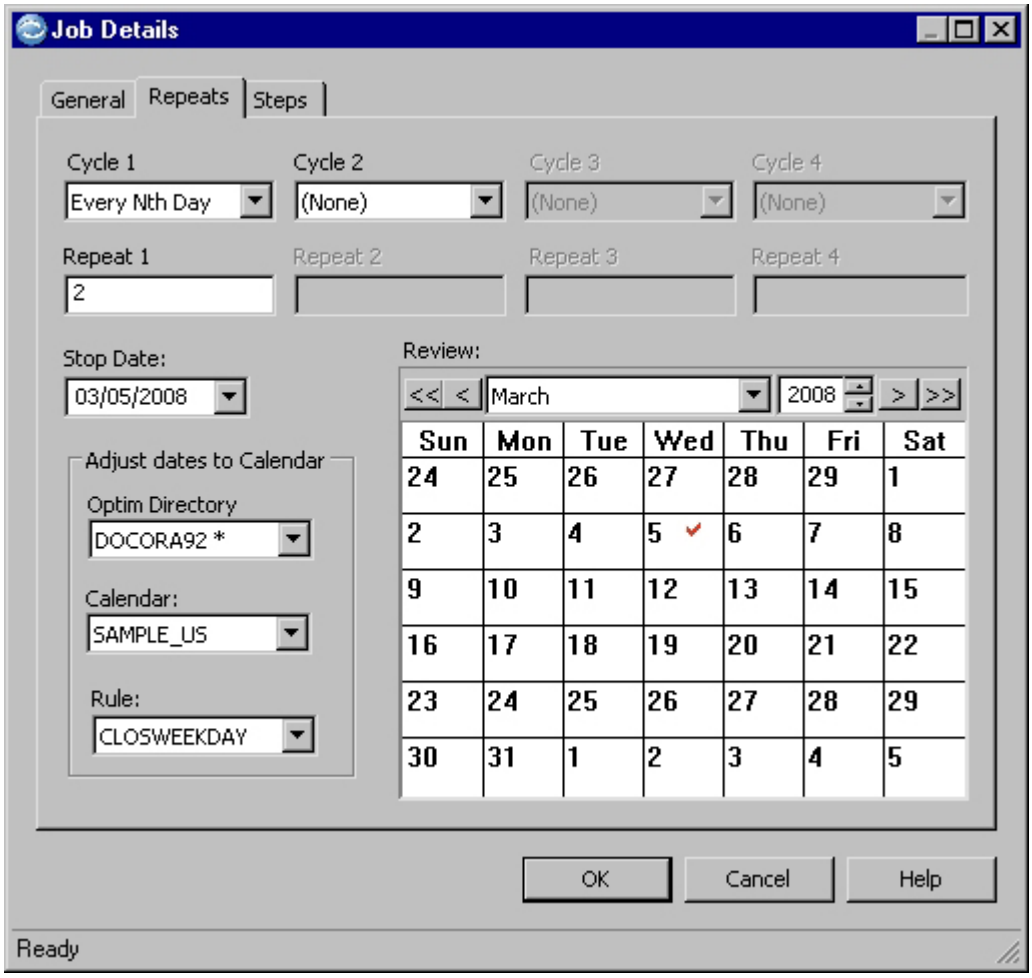
Start Time
Enter the time the processing is to start (in 24-hour time). For example, 4:00 PM is shown as 16:00.

Latest Enter the latest start time for processing to begin (in 24-hour time). The process must begin on or before this time or the job is not run.

Hold Job and do not process
Select this check box to place a job on hold. The job does not run until you clear this check box.

Repeats

Use the **Repeats** tab in the Job Details dialog to specify parameters for a job to run on a regular schedule.



When you specify scheduled jobs to run on a regular or periodic cycle, you can select a calendar and a rule to adjust dates to comply with specific business requirements. For example, you can schedule a job

to run every *n*th day, and select a calendar and rule to specify that the job only runs on weekdays, not weekends. For details on creating Calendars and selecting rules, see Chapter 12, “Calendars,” on page 259.

Details

Cycles (1-4)

Specify how often to run a scheduled job. You can specify up to four different levels of frequency, or cycles, for scheduled jobs. The available choices for each cycle depend on your choice in the previous cycle. **None** is the default. For example:

- If you select **Every Nth Year** for Cycle 1, you must specify **Month** for Cycle 2, and can choose **Day or Week** for Cycle 3.
- If you select **Day** for Cycle 3, you can choose **None** or **Every Nth HH:MM** for Cycle 4.
- If you select **Week** for Cycle 3, you can choose **Day of Week** for Cycle 4.

Note: When you choose a cycle, you can specify the number (N) of repeats within that cycle.

Repeat (1-4)

Specify the number of times a job is repeated within each cycle. The repeat interval must be appropriate for the cycle.

- If you select **Every Nth Year** (Month, Week, or Day) for Cycle 1, you can specify the Repeat as an integer.
- If you select **Every Nth Month** (Week or Day) for Cycle 2, 3, or 4, you can specify the Repeat as a specific month (January to December), within a range of 1 to 4 weeks, 1 to 31 days, and 00:00 to 23:50 hours and minutes.

Stop Date

Specify the last date the scheduled job is to run. To open a calendar to select the date, click the down arrow. To schedule the job to run periodically without a specific end date, click **Never** at the bottom of the calendar.

Adjust Dates to Calendar

Select a calendar for adjusting the dates a scheduled job is to run according to the set of parameters specified for the calendar. SeeChapter 12, “Calendars,” on page 259 for additional information.

Optim Directory

Enter the name of the Optim Directory that contains the calendar to use. If you have access to more than one Optim Directory, click the down arrow to select from a list.

Calendar

Enter the name of the calendar. To select from a list, click the down arrow.

Rule Select the name of a business rule defined to the calendar to use for job scheduling. To select from a list, click the down arrow.

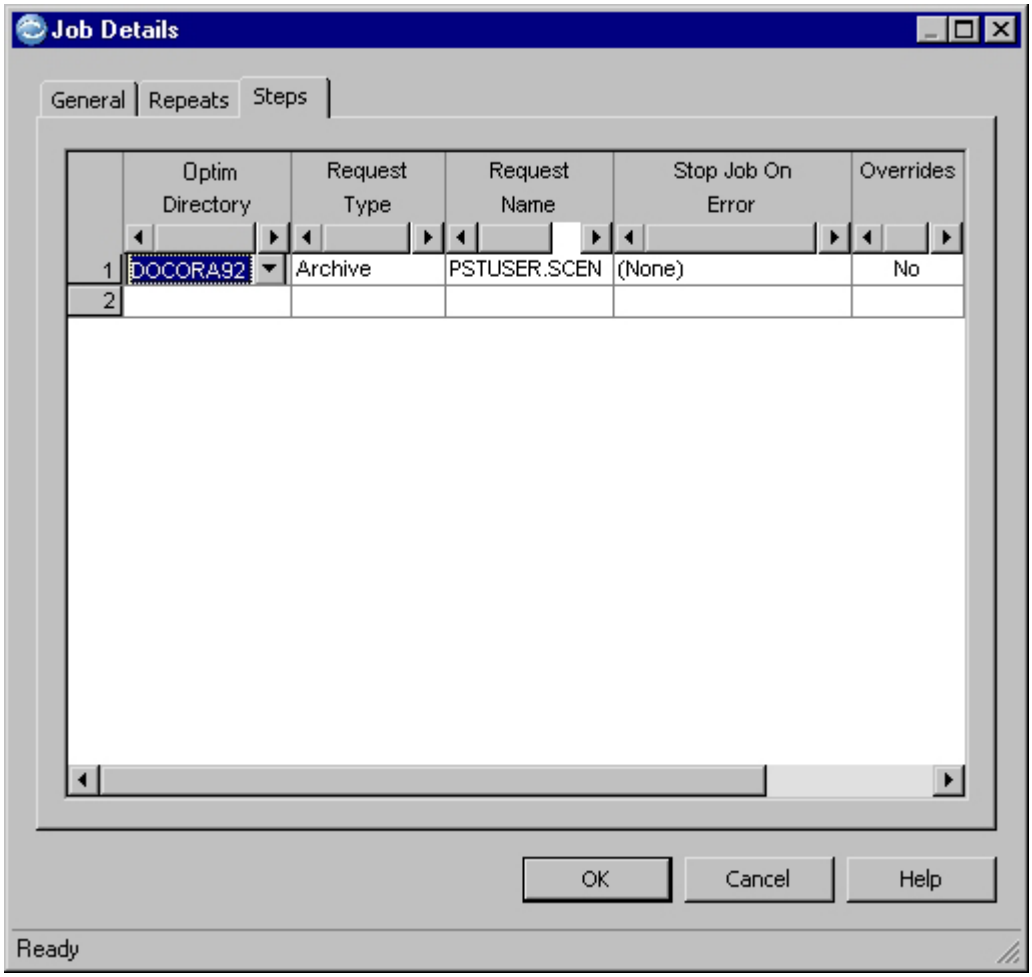
Review

Displays a calendar identifying the days scheduled jobs are to run. A red check mark on the calendar day indicates the day the scheduled job is to run.

- To select a different calendar month, click the down arrow.
- To select a different year, click the spin buttons.
- To move forward or backward one month at a time, click the single arrow.
- To move forward or backward one year at a time, click the double arrow.

Steps

A scheduled job can include several process requests. Use the **Steps** tab on the Job Details dialog to review and modify the process requests that comprise a particular job and indicate whether or not to stop processing if an error occurs. List requests in the order or sequence in which to process.



Details

Optim Directory

Enter the name of the Optim Directory where the process request is stored. To select from a list, click in the grid cell.

Note: If you select a different Optim Directory, the Directory is checked to verify that the request is saved in that Directory.

Request Type

Specify the type of processing request (Archive, Compare, Convert, Delete, Extract, Insert, Load, Report or Restore). To select from a list, click in the grid cell.

Request Name

Enter the name of the process request to include in the scheduled job. To display a browse button, click in the grid cell. Click the browse button to display a list of the specified request type.

Stop On Error
Select the type of error on which to stop processing subsequent steps of a scheduled job. This is applicable to scheduled jobs with multiple steps only. To select from the list of valid entries, click in the grid cell.

None Select to continue processing all steps regardless of errors.

Informational
Select to stop the processing of subsequent steps if an informational, warning, or fatal error occurs.

Warning
Select to stop the processing of subsequent steps if a warning or fatal error occurs.

Fatal Select to stop the processing of subsequent steps if a fatal error occurs.

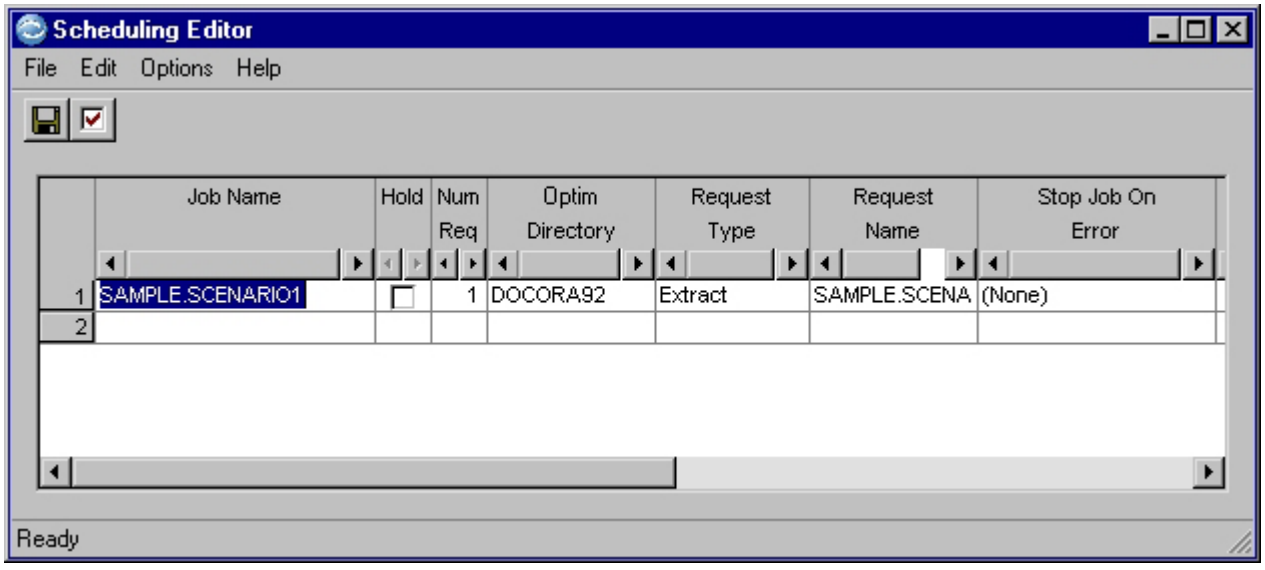
Overrides
Indicates with a Yes or No whether request parameters have been modified using the Overrides dialog. Click the grid cell to display a browse box, then click the Browse box to display the Overrides dialog. See "Overrides Dialog" on page 290 for more information on that dialog.

Scheduling Editor

Use the Scheduling Editor to review and edit details of scheduled jobs. To open the editor, select **Schedule** from the **Utilities** menu on the main window.

To edit the details of a scheduled job, type directly in the grid or right⌂click the job name to display a shortcut menu. On the shortcut menu, select a tab to open the Job Details dialog to the corresponding tab or select **Delete Job** to delete a scheduled job.

The Scheduling Editor lists the jobs scheduled and displays the following details.



Details

Job Name
A 1 to 30 character name used to identify a scheduled job. You can modify the name or enter a new descriptive name.

Hold Select the check box to hold the job. The job cannot run until you clear this check box.

Num Req
The number of steps in the scheduled job. You can list several process requests as steps in a scheduled job. See "Steps" on page 287 for more information.

Optim Directory
The name of the Optim Directory where the process request is stored. If there is more than one step in the scheduled job, the grid cell displays the name of the Optim Directory where the process request for the first step is stored.

Request Type
The type of processing request (Archive, Compare, Convert, Delete, Extract, Insert, Load, Report or Restore). You can modify the type of processing request. If there is more than one step in the scheduled job, the grid cell displays the type of request for the first step.

Request Name
The name of the processing request. You can modify the name. If there is more than one step in the scheduled job, the grid cell displays the name of the process request for the first step.

To display a browse button, click in the grid cell. Click the browse button to display a list of the specified request type.

Stop Job on Error
Processing upon an error or warning condition. This option applies to scheduled jobs with multiple steps only. To select from the list of valid entries, click the grid cell.

None Continue processing all steps regardless of errors or warnings.

Informational
Stop the processing of subsequent steps if an informational, warning, or fatal error occurs.

Warning
Stop the processing of subsequent steps if a warning or fatal error occurs.

Fatal Stop the processing of subsequent steps if a fatal error occurs.

Overrides
Indicates with a Yes or No whether request parameters have been modified using the Overrides dialog. Click the grid cell to display a browse box, then click the Browse box to display the Overrides dialog. See "Overrides Dialog" on page 290 for more information on that dialog.

Rpts A check indicates that repeat processing is specified for a particular job. You cannot edit this check box. To specify repeat processing, double-click the check box to open the Job Details dialog.

Start Date
The date the scheduled job is to begin. To open a calendar to select the start date, click the down arrow.

Note: The calendar is similar to the calendar on the **Repeats** tab. See "Repeats" on page 285 for more information.

Start Time
The time the processing is to start (in 24-hour time). For example, 4:00 PM is shown as 16:00.

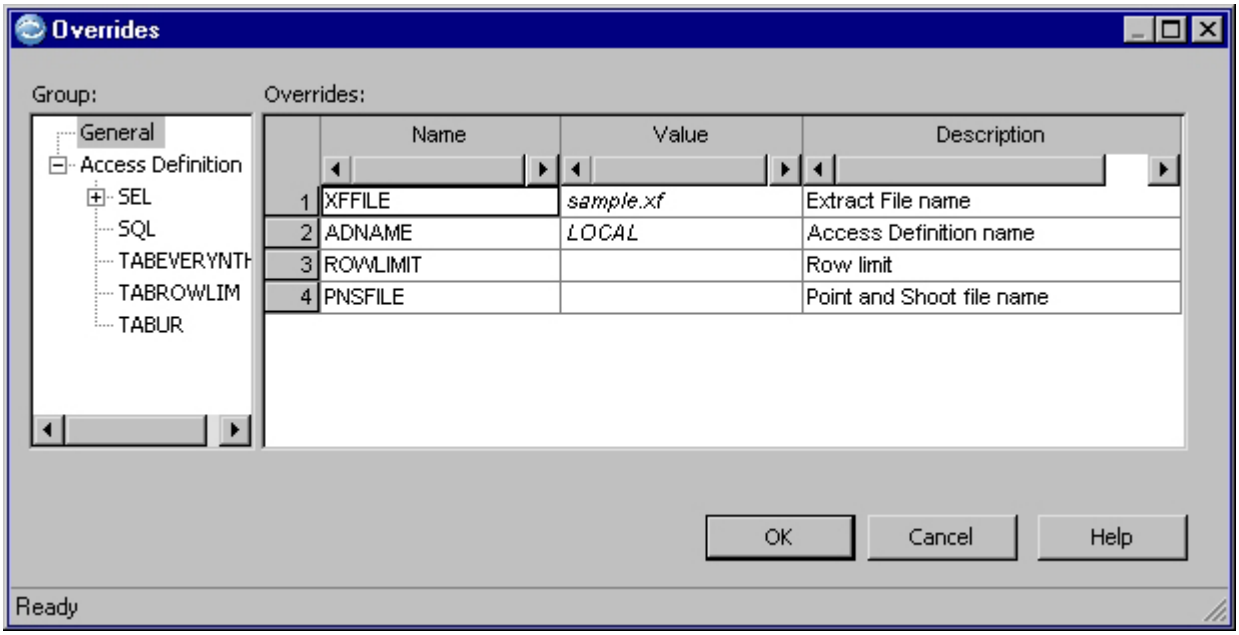
Latest Time
The latest start time for processing to begin (in 24-hour time). The process must begin on or before this time or the job is not run.

Next Start
The next date and time the scheduled job is to begin. You cannot edit this entry.

Overrides Dialog

Use the Overrides dialog to modify parameters of a scheduled process request. In the Scheduling Editor, click the **Overrides** column of a process request to display a browse button in the column. Click the browse button to display the Overrides dialog.

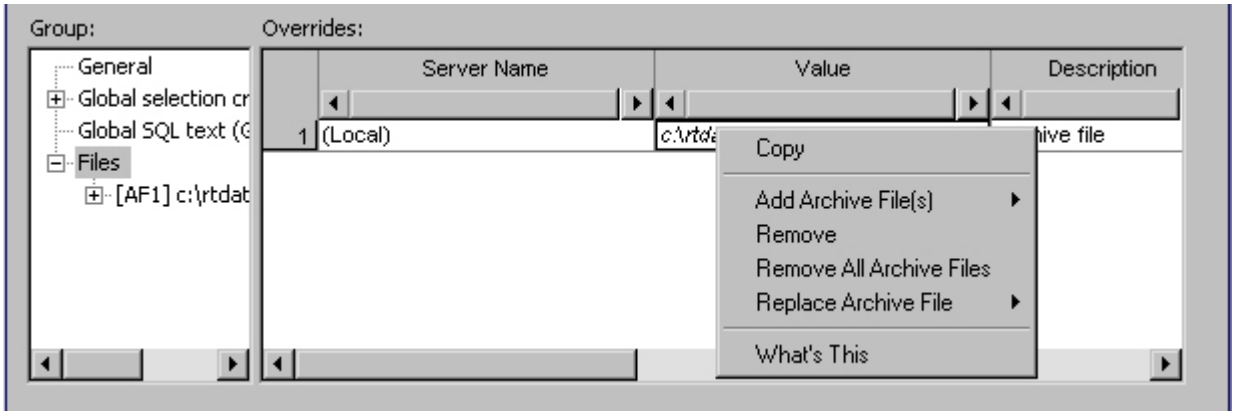
The Overrides dialog is divided into two areas. The Group area identifies the parameters that can be modified in the process request. The Overrides area contains the specific names, values, and descriptions for the selected parameter.



You can modify any named override by changing the corresponding Value column, as required. Current and default values are displayed in italics. To reset a value to the default or current value specified in the request, right-click in the grid cell and select **Insert Default**.

For additional information about specific named overrides for a particular process request, refer to the Command Line Interface section of the corresponding user manual.

Note: When specifying overrides for a scheduled Restore Request, you can select shortcut menu options to add, delete, or replace Archive Files in the request. To display the shortcut menu, select **Files** in the Group area and right-click the list of Archive Files in the Values column.



For complete information about adding, deleting or replacing Archive Files in a Restore Request, refer to the *Archive User Manual* .

Click **OK** to return to the **Scheduling Editor**.

Scheduling Monitor

Use the Scheduling Monitor dialog to review and edit the jobs you have scheduled for processing. The Scheduling Monitor dialog displays when you open the Scheduler. This component installs automatically when you install Optim.

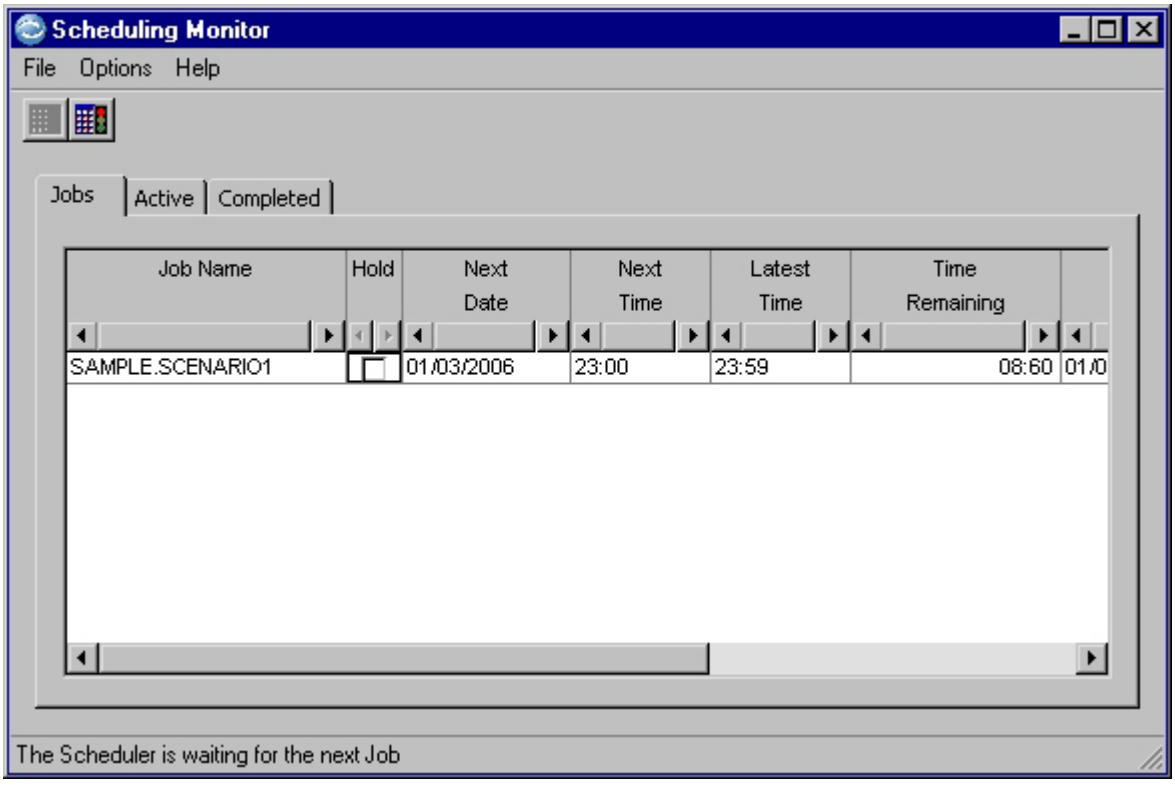
Note: You can set Personal Options to start the Scheduling Monitor dialog when the monitor starts. For details, see Chapter 18, "Personal Options," on page 427.

The Scheduling Monitor dialog has three tabs. You can review the list of pending jobs, monitor active jobs, and review the results of completed jobs.

Jobs

Use the **Jobs** tab in the Scheduling Monitor dialog to review the specifications for processing each job in the queue.

- To edit a scheduled job from the Scheduling Monitor dialog, display the Job Details dialog by double-clicking the job name, or right-click and select **Edit** from the shortcut menu.
- To run a scheduled job immediately, right-click and select **Run** from the shortcut menu.



Details

The **Jobs** tab on the Scheduling Monitor dialog includes the following:

Job Name
Name or description of the job scheduled for processing.

Hold Select to hold the job. To cancel the hold, clear this check box.

Next Date
Next date the job is scheduled to run, or (none) if a date is not specified.

Next Time
Next time the job is scheduled to start, or (none) if a time is not specified. Time is shown in 24-hour time format. For example, 4:00 P.M. is shown as 16:00.

Latest Time
Latest start time for processing to begin (in 24-hour time format), or (none) if a time is not specified.

Time Remaining
Time (in days, hours and minutes) remaining before the job is scheduled to start, or (none) if no time remains.

Stop Date
Date to discontinue processing a regularly scheduled job, or (none) if a date is not specified.

Reason Not Scheduled to Start
One of the following messages indicates the current status of a scheduled job or the reason why a scheduled job did not run:

- Scheduler is not running.
- Job is on hold.
- Job has not reached Start Date.
- Scheduler is processing job.
- Job already processed and no repeats.
- Job is past Stop Date
- Scheduler was not running before Stop Date.

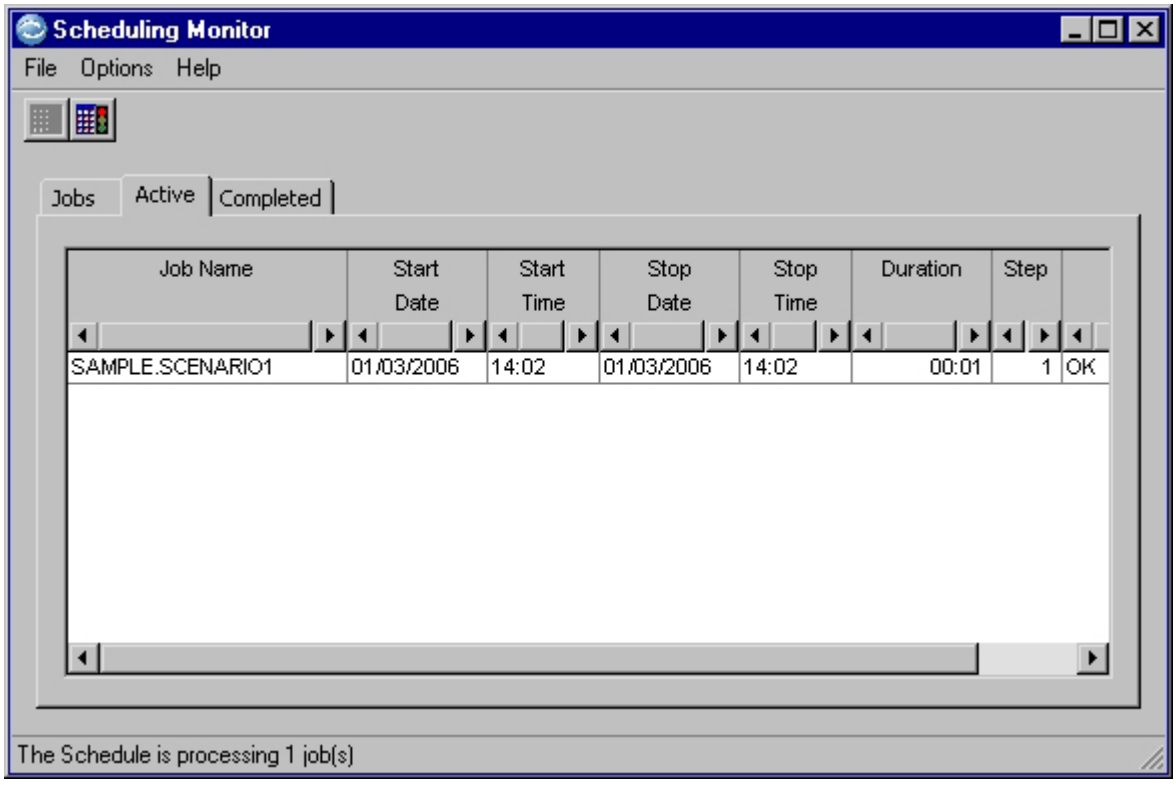
Shortcut Menu

Right-click a job name to select one of the following options from the shortcut menu:

- To process a scheduled request immediately, right-click and select **Process Now**.
- To edit the scheduled request, select **Edit** to display the Job Details dialog. (Refer to "Job Details Dialog" on page 284.)

Active

Use the **Active** tab in the Scheduling Monitor dialog to review the specifications for jobs that are currently being processed.



Details

The **Active** tab on the Scheduling Monitor dialog includes the following:

Job Name
Name or description of the job scheduled for processing.

Start Date
Date the job is scheduled to run, or (none) if a date is not specified.

Start Time
Time the job is scheduled to start, or (none) if a time is not specified. Time is shown in 24-hour time format. For example, 4:00 P.M. is shown as 16:00.

Stop Date
Date processing completed.

Stop Time
Time processing completed.

Duration
Elapsed processing time. Check the process report for warnings or errors if processing time is significantly shorter or longer than anticipated.

Step Step currently being processed in the job.

Status A status indicator for the scheduled job or step being processed:

Waiting to Start
The scheduled job has not started.

Executing
The scheduled job is processing.

Timed Out
Resources available to process the scheduled job in the allotted time are Insufficient.

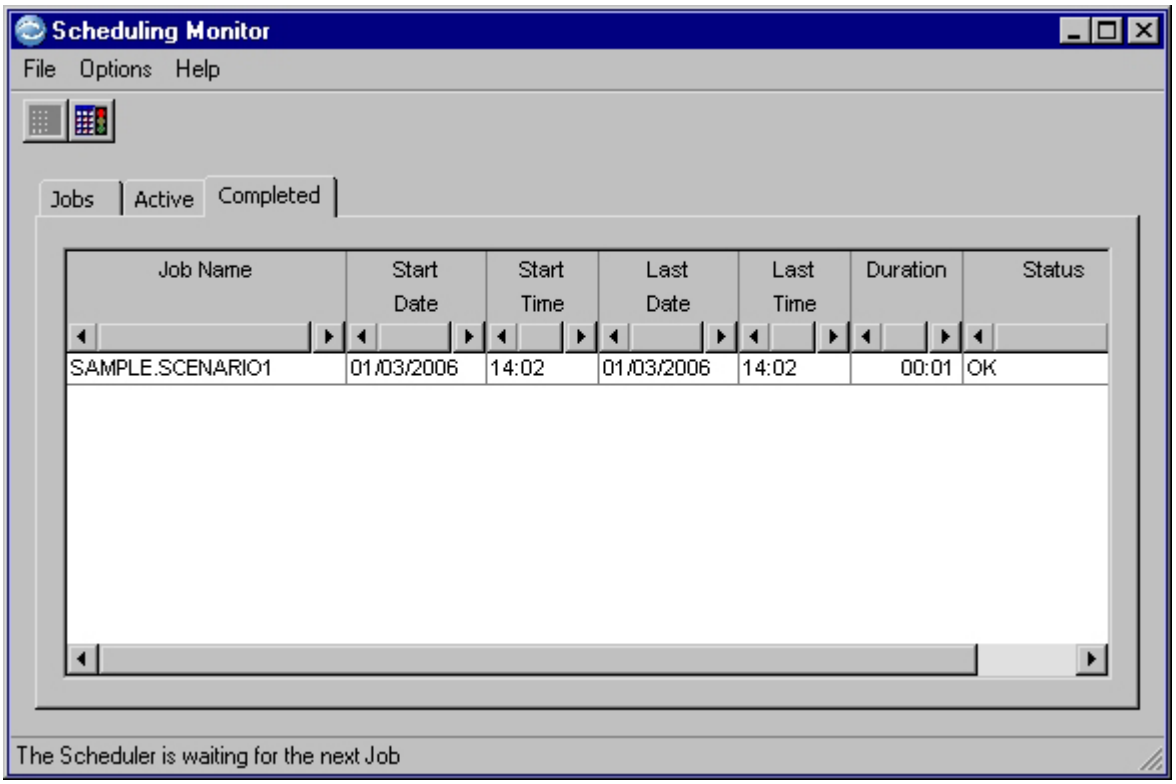
Shortcut Menu

Right-click a job name to select one of the following options from the shortcut menu:

- To show or hide the process progress dialog, right-click and select **Show Monitor** or **Hide Monitor**.
- To display completed steps of a scheduled job that contains more than one step, select **Display Completed Steps**.

Completed

Use the **Completed** tab in the Scheduling Monitor dialog to review the details and status of completed jobs.



Details

The **Completed** tab on the Scheduling Monitor dialog displays the following:

Job Name
Name or description of the scheduled job.

Start Date
Date the scheduled job ran.

Start Time
Time the scheduled job started. Time is shown in 24⌂hour time format. For example, 4:00 P.M. is shown as 16:00.

Last Date
Last date the scheduled job ran.

Last Time
Last time the scheduled job ran.

Duration
Time the scheduled job took to process.

Status A status indicator for the scheduled job. (Use the shortcut menu to determine the status of each step of a scheduled job.)

OK Scheduled job completed successfully.

Informational
Scheduled job has stopped due to an Informational error.

Warning
Scheduled job has stopped due to a Warning error.

Fatal Scheduled job has stopped due to a Fatal error.

Timed Out
Scheduled job did not successfully complete prior to the Stop Time.

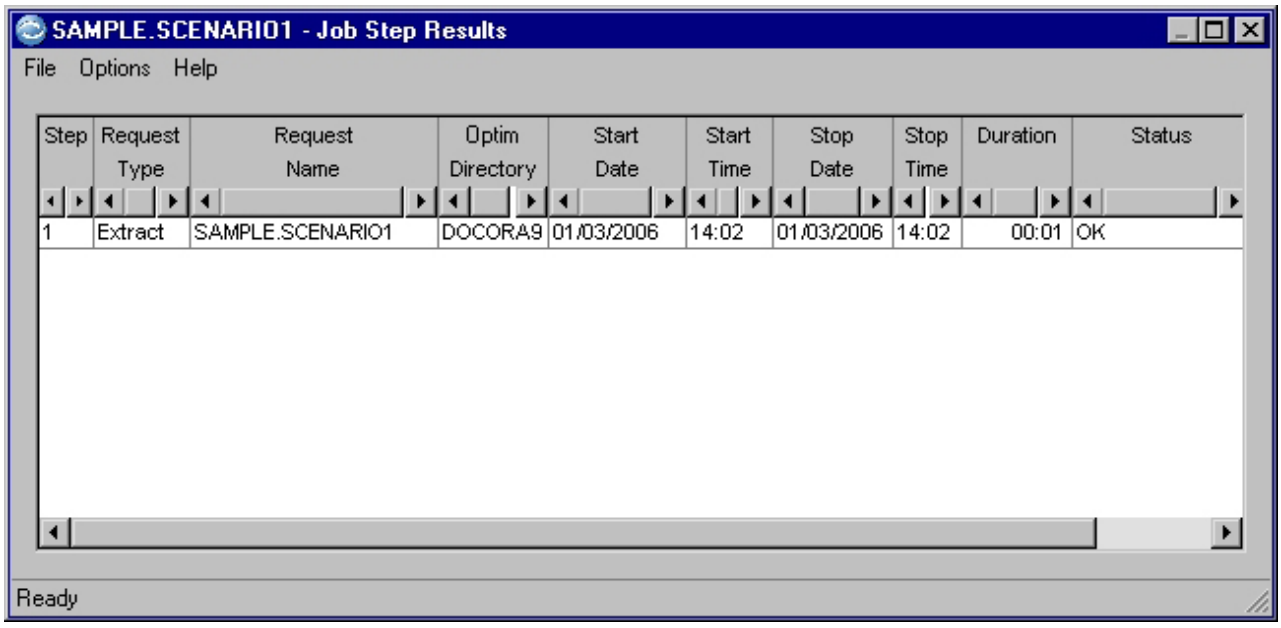
Shortcut Menu

Right-click the name of a scheduled job to select an option from the shortcut menu:

- To display the results of each step in the scheduled job, right-click and select **Display Steps** from the shortcut menu.
- To delete a job from the list, right-click and select **Delete** from the shortcut menu.

View Job Step Results

When you select **Display Steps** from the shortcut menu you can view details of each step in the completed job.



Details

The Job Results dialog displays the following:

Step Sequence number for each step in the scheduled job.

Request Type
Type of processing request for each step.

Request Name
Name of each process request.

Optim Directory
Name of the Optim Directory where each process request is stored.

Start Date
Date each step began.

Start Time
Time each step began.

Stop Date
Date each step stopped or completed.

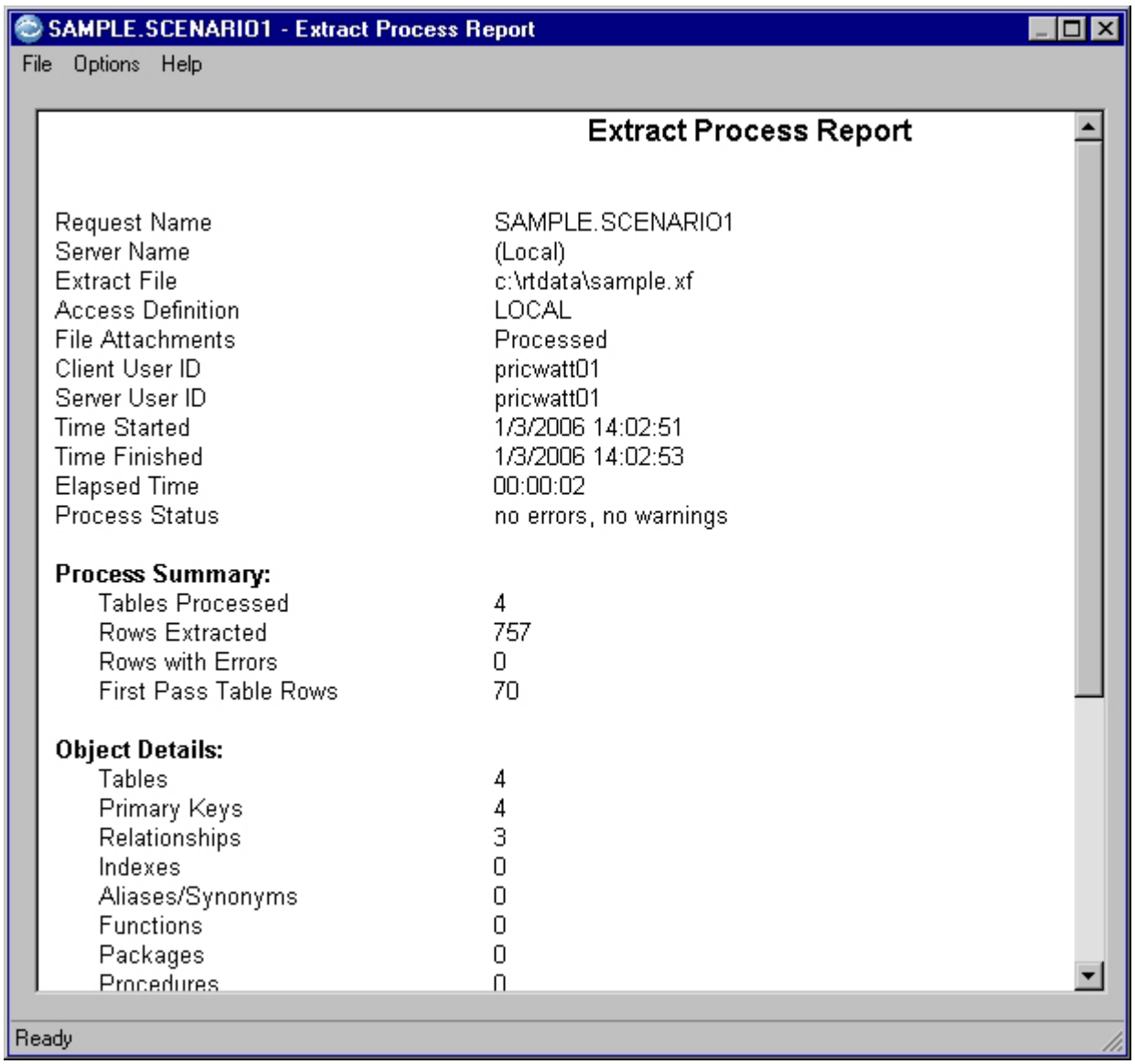
Stop Time
Time each step stopped or completed.

Duration
Actual elapsed processing time for each step.

Status Status of each step in the job. See the **Completed** tab for the list of possible status indicators.

View the Process Report

To view or print the process report for any step in a completed job, double-click the job name to open the Process Report dialog. The Process Report contains detailed information about the completed step.



To print or save the report, select **Print** or **Save As** from the **File** menu in the Process Report dialog.

Chapter 15. Browse

Use the Browse Utility to review the contents of an Archive, Compare, Extract, or Control File. You can browse the contents of a file to obtain information or to determine that the data is as expected.

For example, you can browse the contents of an Archive File to review data without having to restore it to the database. This may be an easy solution when handling a customer inquiry or resolving a problem. Browse the contents of a Control File to review rows discarded during an Insert, Convert, Load or Delete Process, and the reasons they were discarded. The Control File identifies each row that was processed with a code for the reason for each failure.

Default extensions for the files are as follows:

Archive File	.af	Extract File	.xf
Compare File	.cmp	Control File	.cf

Note: See the *Compare User Manual* for detailed information about browsing data in a Compare File

The Browse Utility allows you to select a table in a file and browse the data in the table. Table data is displayed in a dialog that allows you to display character data in hexadecimal, exclude selected rows from the display, and navigate the displayed data. When browsing an Archive, Control, or Extract File, you can join to related rows in other tables.

Long Object Names (LONs)

When browsing an Archive file migrated from the mainframe version of Optim (that is, the Optim z/OS® Solution), any object names that exceed the maximum length for an Optim client/server object of the same type are truncated. When this happens, the truncated name is suffixed with the code "__TRUNC__". A 128-character Creator ID, for example, might be truncated in Optim as follows:

PSTASLG_1234567890_xx__TRUNC__

Contents

This section explains how to browse data in these files, manipulate the display, and perform the following tasks:

- Use the components of the Browse dialog.
- Browse data and change the display using grid facilities.
- Join tables to browse related data from other tables in an Archive, Control, or Extract File.

Open a File to Browse

You can browse an Archive, Compare, Extract, or Control File:

From the Main Menu

To open a file to browse:

1. In the **Utilities** menu, select **Browse** to open the **Browse** dialog.
2. In the **File** menu:
   - Select **Open** to display the Open dialog to locate and open a file, OR

- Select **Last Created** [**Archive**, **Compare**, **Extract**, or **Control**] **File** to open the most recently created file of the type you select.

From a Request Editor or list of files

To open a file to browse:

- Right-click the file name and select **Browse** from the shortcut menu.

From Windows Explorer

To open a file to browse:

- In Windows Explorer, double-click a file name or drag the file name to the Optim icon on the workstation desktop to open the selected file.

The Browse dialog displays the name of the selected Archive, Compare, or Extract File in the title bar. When you browse a Control File, the name of the Control File appears briefly in the status bar before the name of the associated Archive or Extract File appears in the title bar.

Note: Each Control File is associated with an Archive or Extract File. To open the Control File, the associated file must be available on the drive and in the directory specified when the Control File was created. If the associated file cannot be found, the Control File will not open. Instead, an error message displays the fully qualified name of the associated file, as saved in the Control File.

While browsing an Archive, Control, or Extract File, use the **Actions** menu commands to open the Convert, Delete, Insert, or Load Request Editor, with the selected file as the Source File. If a menu command is grayed out, it is not available for the type of file you are browsing.

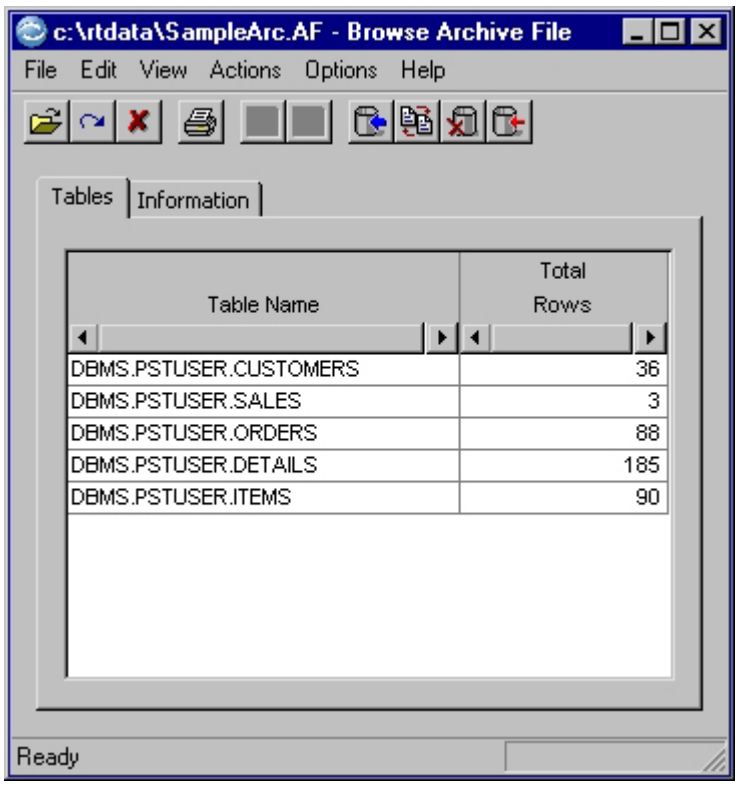
Use **View** menu commands when browsing a Control File, as follows:

Hide Row Status
Select to omit the status and action information columns in the grid.

Only Show Rows in Error
Select to include only discarded rows in the grid.

Tables Tab

Use the **Tables** tab to view, but not edit, information about the tables in the selected file.



Details

The following details are displayed:

Table Name
Names of tables in the file.

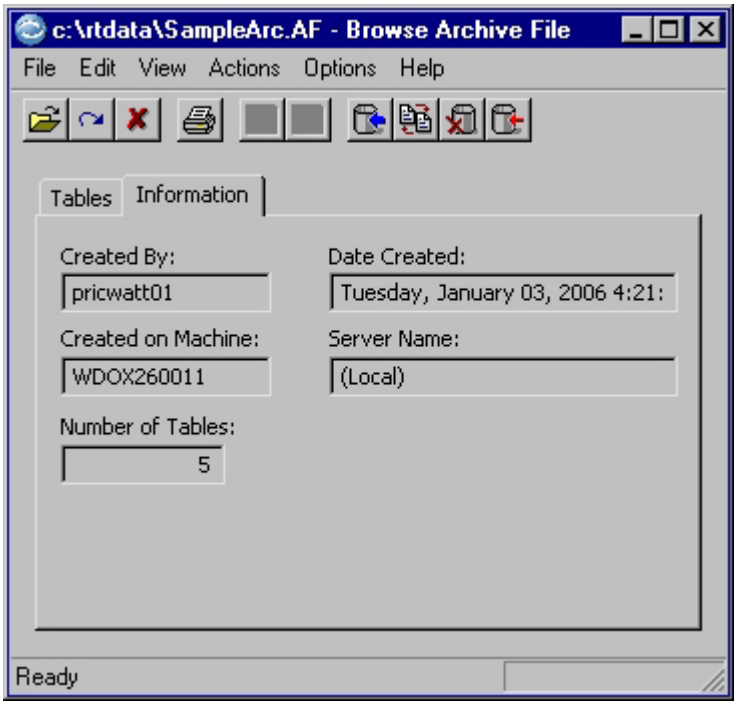
Total Rows
Number of rows in each table.

Rows in Error
Number of rows not processed (Control Files only).

Use the **Find** option, available on the grid heading shortcut menu, to locate specific information in a large display. (See “Using Find and Replace” on page 23 for more information.)

Information Tab

Use the **Information** tab to review system information about the selected file.



Details

The following details are displayed:

Created By
User ID of the person that ran the process that created the file.

Date Created
The file creation date in Long Date and Time format, as set on the local machine.

Created on Machine
Identifier for the machine from which the file was created.

Server Name
Name of the Optim Server from which the file is accessed or "Local" if the file is accessed from the workstation.

Number of Tables
Number of tables in the file.

Control File Name
Control File name is displayed only when browsing a Control File. The associated file name is displayed in the dialog header.

SQL ID
Identifier assigned to a process that originated in the DB2 MVS environment. SQL ID is displayed when browsing a DB2 MVS Extract File.

Subsystem
Subsystem for a file that originated in the DB2 MVS environment. Subsystem is displayed when browsing a DB2 MVS Extract File.

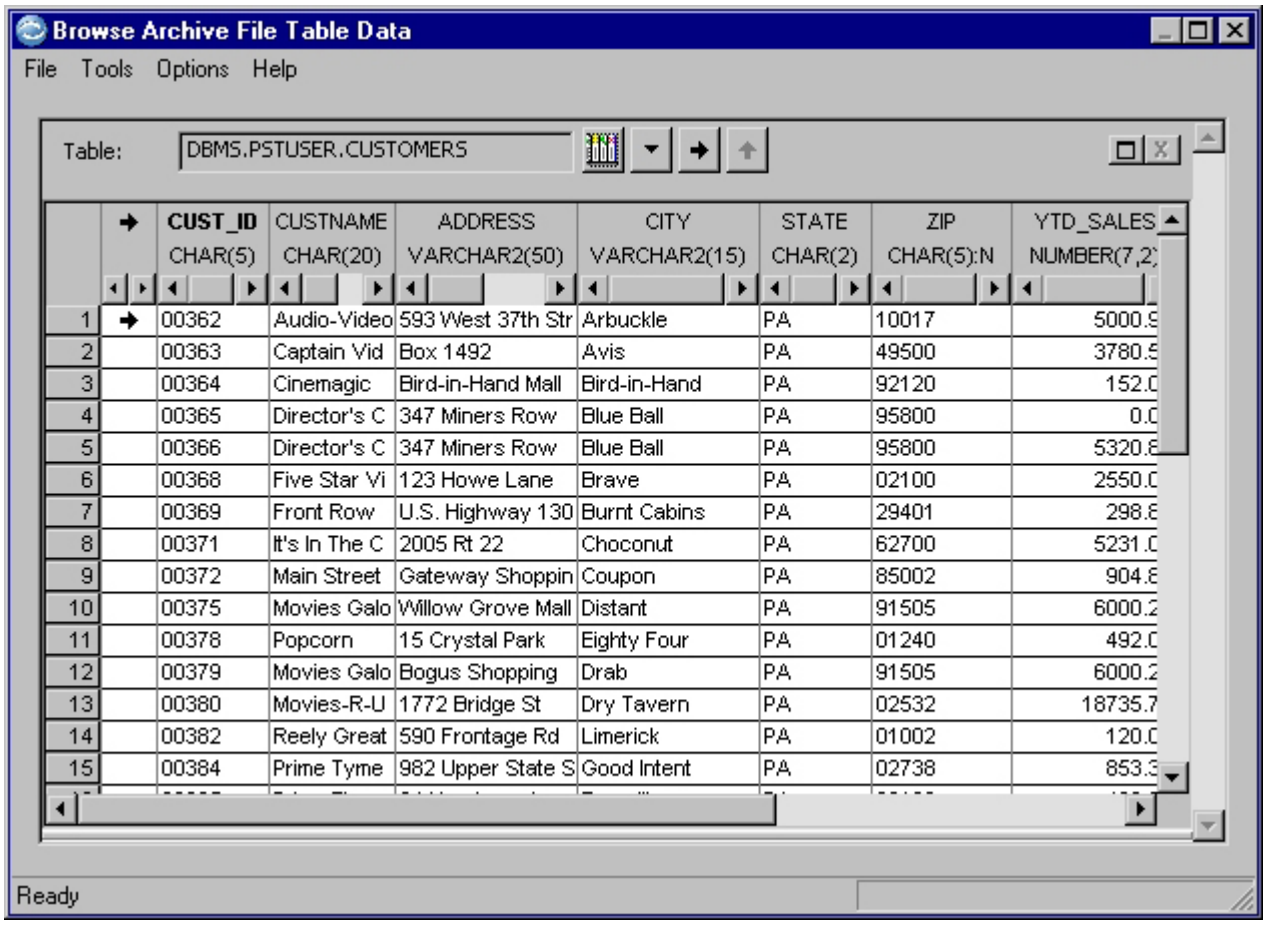
Display Table Data

To display the rows from a particular table or tables:

- Double-click the table name in the grid.
- Select names of tables in the grid, then right-click and select **Display Rows** from the shortcut menu, or select **Display Rows** from the **File** menu. You can use ctrl or shift to select more than one table name, or use the **Select All** and **Invert Selection** options on the **Edit** menu or the grid shortcut menu.

A Browse Table Data dialog displays data from the selected table.

Note: If you make more than one selection, several Browse Table Data dialogs open. Move a Browse Table Data dialog to see other dialogs beneath.



	CUST_ID CHAR(5)	CUSTNAME CHAR(20)	ADDRESS VARCHAR2(50)	CITY VARCHAR2(15)	STATE CHAR(2)	ZIP CHAR(5):N	YTD_SALES NUMBER(7,2)
1	00362	Audio-Video	593 West 37th Str	Arbuckle	PA	10017	5000.9
2	00363	Captain Vid	Box 1492	Avis	PA	49500	3780.5
3	00364	Cinemagic	Bird-in-Hand Mall	Bird-in-Hand	PA	92120	152.0
4	00365	Director's C	347 Miners Row	Blue Ball	PA	95800	0.0
5	00366	Director's C	347 Miners Row	Blue Ball	PA	95800	5320.8
6	00368	Five Star Vi	123 Howe Lane	Brave	PA	02100	2550.0
7	00369	Front Row	U.S. Highway 130	Burnt Cabins	PA	29401	298.8
8	00371	It's In The C	2005 Rt 22	Choconut	PA	62700	5231.0
9	00372	Main Street	Gateway Shoppin	Coupon	PA	85002	904.8
10	00375	Movies Galo	Willow Grove Mall	Distant	PA	91505	6000.2
11	00378	Popcorn	15 Crystal Park	Eighty Four	PA	01240	492.0
12	00379	Movies Galo	Bogus Shopping	Drab	PA	91505	6000.2
13	00380	Movies-R-U	1772 Bridge St	Dry Tavern	PA	02532	18735.7
14	00382	Reely Great	590 Frontage Rd	Limerick	PA	01002	120.0
15	00384	Prime Tyme	982 Upper State S	Good Intent	PA	02738	853.3

Note: Binary data is not displayed. Cells containing binary data are shaded. To view binary data, use the hexadecimal display. For more information about viewing binary data, see “Column Data Display” on page 310.

Data is displayed in columnar format (as shown) or in side label format, depending on the default setting in Personal Options. (Refer to Chapter 18, “Personal Options,” on page 427 for additional information about setting display defaults.)

A browse window contains the following components:

Table

Name of the table displays to the left of the toolbar.

Toolbar

The toolbar allows you to select display options and menu choices for the browse window, as follows:

Format

or 

Switch the data display between columnar and side label format. The default format is set in Personal Options. For more information about display formats, see "Display Options" on page 307.

Options

Display the browse window **Options** menu.

Display Attributes
Switch between displaying and hiding column attribute information in the column headings.

Display Source 2
(Available when browsing a Compare File.) Select to display Source 2 column names and data attributes. Clear to display Source 1 column names and data attributes.

Note: The name and label of the table at the top of the dialog changes to reflect your selection.

Show Unmatched Columns
(Available when browsing a Compare File.) Display or hide unmatched columns (columns excluded from Compare processing, using a Column Map).

This option is available in a columnar display only, and is disabled for tables that do not have unmatched columns.

Note: Names of unmatched columns are shown in the column header with a number prefix to indicate the Source for the column.

Show Excluded Rows
Display all previously excluded rows (rows are excluded using the **Exclude** command on the shortcut menu). To display excluded rows individually, right-click a row and select **Show Next** from the shortcut menu.

Access Definition
If the Extract File or Archive File contains Large Objects (LOBs), select **Access Definition** to establish an association between a LOB and an application used to view the LOB, as follows: From the submenu, choose **Create Local** to remove LOB column associations; **Reload Default** to reload the column associations from the Access Definition stored in the file; **Select...** to copy LOB column associations from an existing Access Definition; or **Columns...** to open the Columns dialog to create new LOB column associations.

For more information, see "LOB Column Associations" on page 312.

Join

Join and view data from a related table. Data from the joined table must be in the file. If more than one table is related to the table from which you join, or more than one relationship exists for a pair of tables, you select from a selection list. See "Display Multiple Tables" on page 314 for details.

Unjoin

Remove the table and all subordinate joined tables from the Browse Table Data dialog.

Show/Hide

When browsing a Control File, click the buttons to show or hide the Status and Action columns, and to switch between a display of all rows and a display of rows in error.

Navigation

In side label format, scroll to display the first row, previous row, next row, or last row, respectively.

Grid Details

Data from the table named on the title bar displays in the grid. The display can be navigated and customized using the **Find**, **Exclude**, **Include**, **Hide**, and **Lock** options available on the grid heading shortcut menu. (See Chapter 2, "Main Window, Menus, and Dialogs," on page 5 for additional information.) When you are browsing a Control File, the grid includes status and action information that can be switched on and off with the toolbar buttons.

Current Row Indicator

Position the Current Row Indicator to display related rows in joined table(s). To move to a different row, click the grid cell for the desired row or use the up/down arrows on your keyboard. See "Display Multiple Tables" on page 314 for details.

Note: The Current Row Indicator grid column and the Status grid column are displayed in columnar format only.

Status For a Control File, the status of the row is displayed. If the row processed successfully, the status is **OK**. If not, a brief description of the error is displayed (most errors are self-explanatory):

Unprocessed
The row was not part of the process, the process was aborted, or the process has not been performed.

Postponed
The row could not be processed due to RI rules, but has the potential to be processed successfully later in the cycle. This status indicates that the process terminated before the row could be reprocessed.

DB: *n* The row could not be processed due to a database error. Refer to the DBMS documentation for an explanation of database error codes.

Referential Integrity Error
The row could not be processed because of a violation of RI rules defined to the DBMS. This status is provided after retrying a row in a cycle and the row is not successfully inserted when the process completes. (Note that when a cycle is not involved, a diagnostic error message is displayed.)

Conversion Error
The row was discarded because one or more columns could not be properly converted from the Extract File format to the database format. This can occur when the Extract File column contains NULL and the DBMS column is defined as NOT NULL or when the Extract File column value and the DBMS column data types are not compatible. This row cannot be restarted or retried.

Act For a Control File, the type of process performed, such as **Upd** for update, **Ins** for insert, or **Del** for Delete.

Grid Shortcut Menu Commands

Right-click a grid cell to display the following shortcut menu commands.

Join Join and view data from a related table. Data from the joined table must be in the file. For more information, see "Display Multiple Tables" on page 314.

Display
Open the Column Data Display dialog for viewing character or hexadecimal value of a cell in a column defined as a character, BLOB, or CLOB data type. Select the format of the data, **Character** or **Hex**. For more information, see "Display Options" on page 307.

Character
Display data offset information and the character representation of the data in the cell.

Hex Displays data offset information and the character and hexadecimal representations of the data in the cell.

Side Label Display
Display data in side label format. Available with columnar display only. For more information about display formats, see "Display Options" on page 307.

Columnar Display
Display data in columnar format. Available with side label display only. For more information about display formats, see "Display Options" on page 307.

Exclude
Remove the row from the display.

Show Next
Display the first excluded row after the row you right-clicked.

Show All
Display all excluded rows between the row you right-clicked and the next displayed row.

Note: To display all excluded rows in the grid, select **Show Excluded Rows** from the **Tools** menu.

Access Definition
If the Extract File or Archive File contains Large Objects (LOBs), select **Access Definition** to establish an association between a LOB and an application used to view the LOB, as follows: From the submenu, choose **Create Local** to remove LOB column associations; **Reload Default** to

reload the column associations from the Access Definition stored in the file; **Select...** to copy LOB column associations from an existing Access Definition; or **Columns...** to open the Columns dialog to create new LOB column associations.

For more information, see "LOB Columns" on page 311.

Run Associated Application
For cells containing LOBs, open the selected object with the associated application.

Note: If an object association has not been established, you will be prompted to create one using the Columns dialog. For more information, see "Columns" on page 313.

Export LOB
For cells containing LOBs, export LOB data to a file.

Display Options

The Browse Table Data dialog provides several ways to manipulate the display.

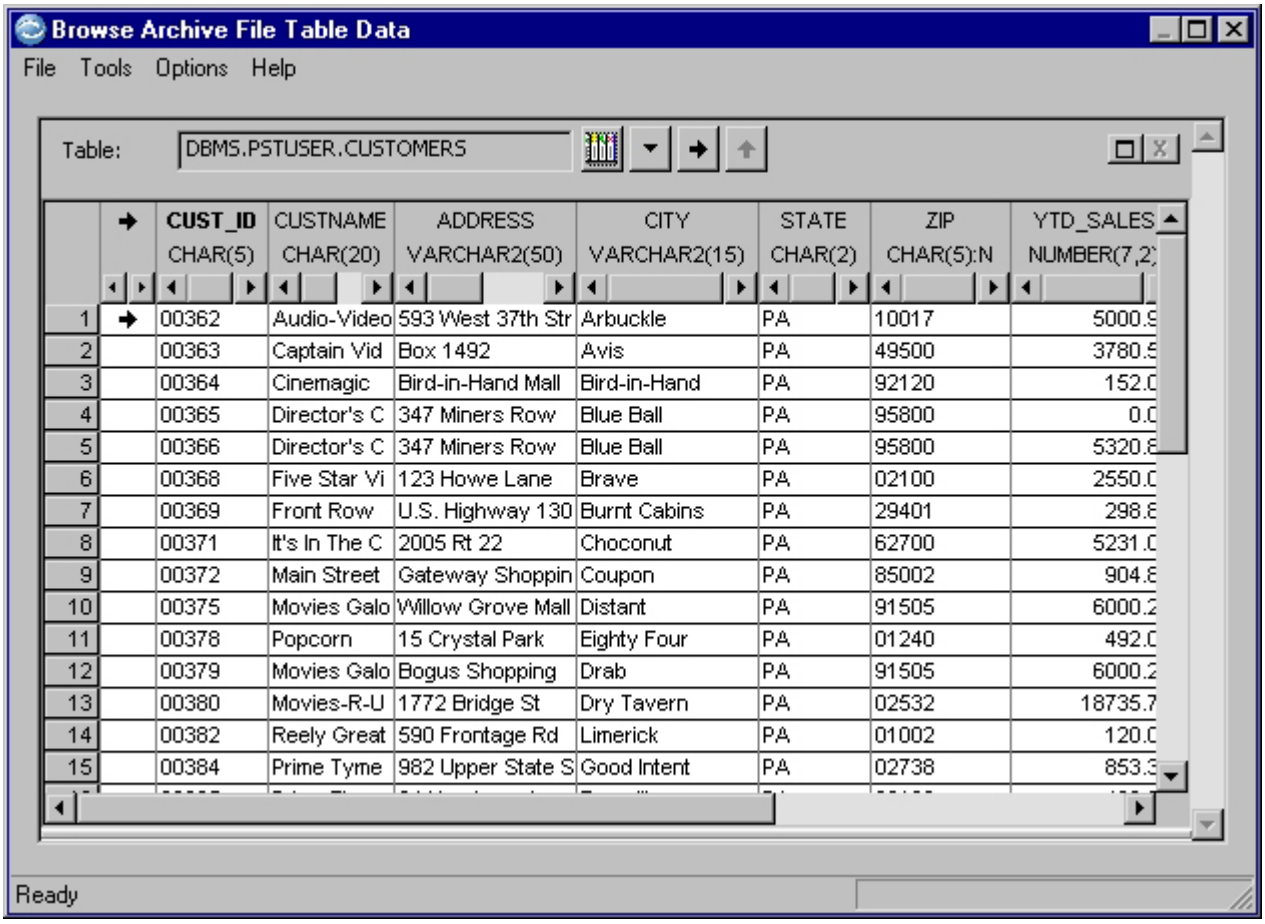
Format Options

A browse window has two format options, columnar and side label. You can switch between the two formats by clicking the  or  button on the browse window toolbar. The default format is columnar. Most of the examples throughout this manual are shown in columnar format.

Columnar

In columnar format, column names are displayed across the top of the browse window and the data is displayed in columns under the headings. Note that the headings for primary key column(s) are in bold type.

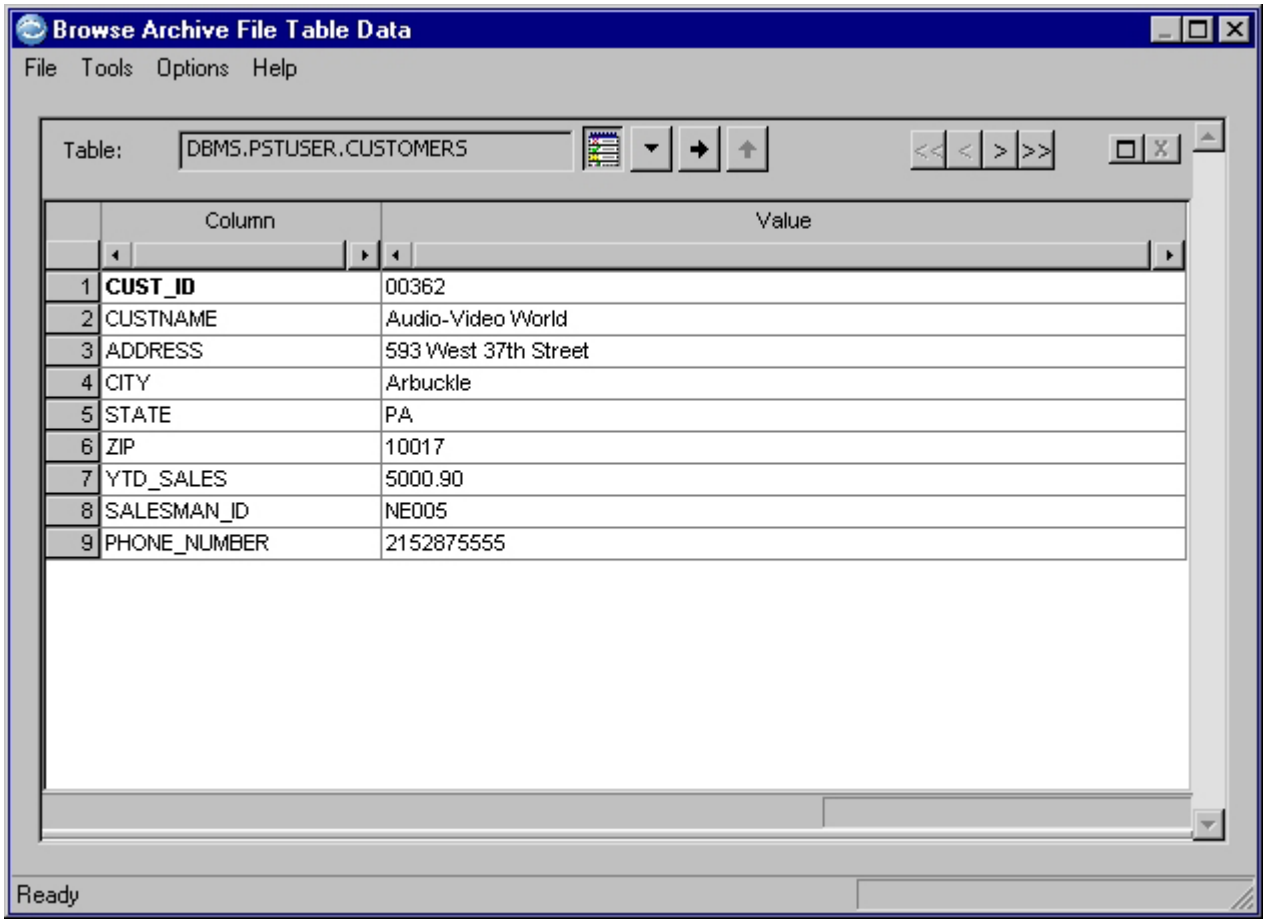
In the following example, the browse window shows several rows of data in columnar format.



Side Label

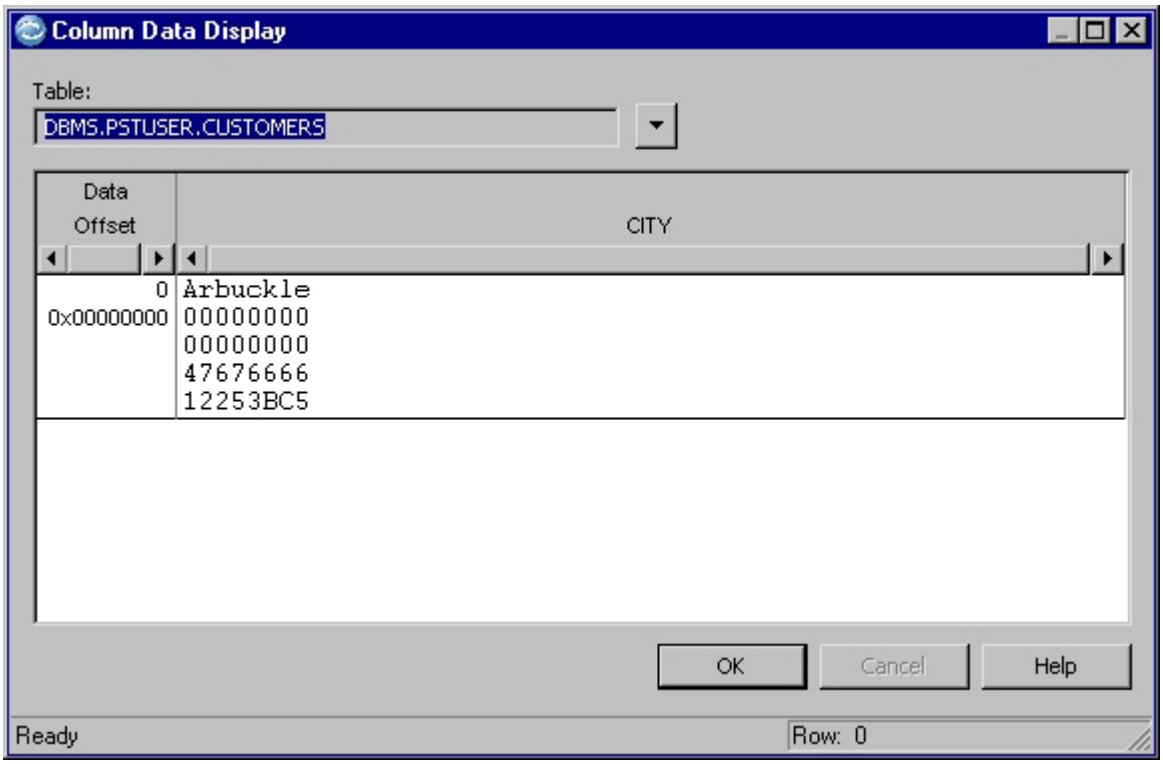
In side label format, the column names are displayed down the left side of the browse window and a row of data is displayed to the right of the headings. The row number is noted in the status bar at the bottom of the browse window. Use the navigation buttons on the browse window toolbar to scroll to another row. Side label format focuses on a single row and can display more columns for the row than the columnar format. Side label format is also useful for browsing data in very wide columns.

In the following example, the browse window shows data in side label format.


Browse Archive File Table Data
File Tools Options Help
Table: DBMS.PSTUSER.CUSTOMERS
Column
Value
1 CUST_ID 00362
2 CUSTNAME Audio-Video World
3 ADDRESS 593 West 37th Street
4 CITY Arbuckle
5 STATE PA
6 ZIP 10017
7 YTD_SALES 5000.90
8 SALESMAN_ID NE005
9 PHONE_NUMBER 2152875555
Ready

Column Data Display

Use the Column Data Display dialog to display a character or hexadecimal representation of data.



Data Offset

Displays the location of data, in bytes, from the beginning of the column or file. For a hexadecimal display of data in UTF-8 or multi-byte format, the number of bytes per line is displayed in parentheses, and if the number of characters displayed is greater or less than the number of characters displayed per row (as determined by the **Characters per Row** option), the offset and bytes per line are displayed in italic type.

Display Character

For character data, right-click a cell and select **Display, Character** from the shortcut menu to display the character representation of the data. For a CLOB, click the Aa icon.

Display Hexadecimal

Right-click a cell and select **Display, Hex** from the shortcut menu to display the character and hexadecimal representations of the column data. For a LOB, click the 01 icon.

The digits that make up the hexadecimal representation of each character are displayed on the lines below that character. For binary columns, the hexadecimal representation is displayed on two lines (the character line contains no data and is shaded). For CLOB columns, the hexadecimal representation includes all bytes, including carriage returns, line feeds, and the byte order mark (BOM).

For the hexadecimal display of character columns, the following applies:

- UTF-16 and Extract File or Archive File data will display the hexadecimal representation on four lines.

- UTF-8 or multi-byte data will display the character over the first byte, and a period will be displayed over any additional bytes. For example, the UTF-8 French character À is displayed as two bytes:
  À.
  C8
  30

  Note:
  - For data in multi-byte format (for example, Oracle JA16SJIS), the character and hexadecimal representations are each displayed in different fonts and may not be aligned.
  - For release 5.3 or earlier Extract Files and Archive Files, the hexadecimal representation is displayed on two lines only.

Options Button

Click the options button to display the following:

Characters per Row
Select the number of characters to display per row: 64, 128, 256, or 512.

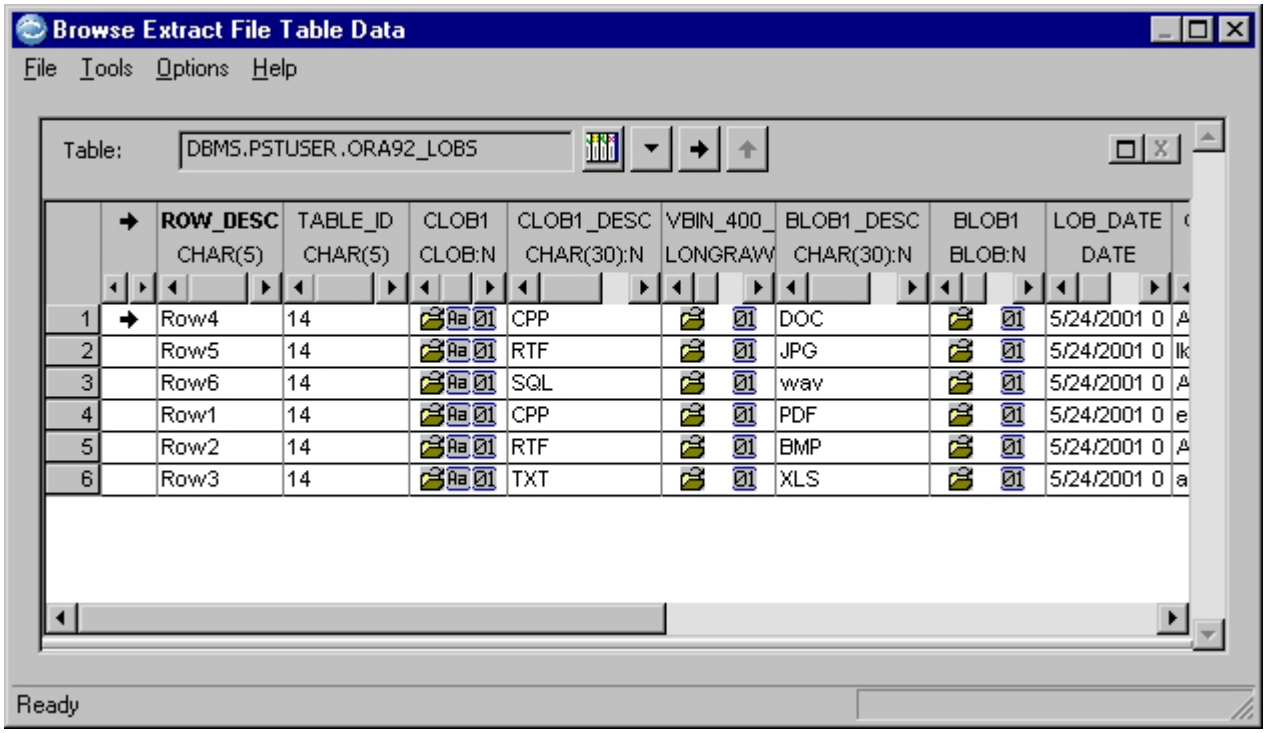
Clear Data
Remove data from the row. Available when text can be modified only.

File Type
For CLOB data only. If the correct encoding scheme for the CLOB file is not displayed, select the encoding scheme, UTF-8 or UTF-16. For UTF-8, the hexadecimal representation is displayed on two lines. For UTF-16, the hexadecimal representation is displayed on four lines.

If a file does not include a byte order mark, the default encoding scheme is based on the data type, CLOB (UTF-8) or NCLOB (UTF-16).

LOB Columns

LOB columns display icons that provide options for browsing:



- Click the  icon to start the application associated with the LOB. If an association has not been established, you are prompted to create one.
- Click the  icon to browse a CLOB in character mode.
- Click the  icon to browse the LOB in hex mode.

The original Access Definition stored with an Archive or Extract File does not include file attachment pseudocolumns. To create column associations for these pseudocolumns, you must create a new Access Definition. For more information, see "LOB Column Associations."

When you browse a Compare File that contains a LOB column, and the Source 1 LOB column is different from the Source 2 LOB column, the background in the LOB column contains a bitmap pattern of dots to indicate that a difference exists.

LOB Column Associations

LOB column associations are stored in the Access Definition embedded in the Archive or Extract File. If the file does not contain an embedded Access Definition, the Browse Utility creates one based on the columns in the file. While browsing a file, you can change the LOB associations in an Access Definition, use the associations in another Access Definition, or create a new Access Definition. If you modify an Access Definition, you are prompted to save it. You cannot reuse an Access Definition that has not been saved.

To associate a LOB column with the application required to view or edit the LOB data (e.g. MS Word, NotePad, Paint, etc.), right-click the column and select **Access Definition** from the shortcut menu to display the following submenus:

Create Local

Click **Create Local** from the submenu to create a new Access Definition based on the columns in the file. The new Access Definition will not contain any file associations.

Reload Default

Click **Reload Default** to reload the original Access Definition stored in the file.

Select

Click **Select** from the submenu to select a named Access Definition. Associations in the named Access Definition you select are used for corresponding columns in the browsed file.

Note: For a Compare File, right-click the Source 1 row or Source 2 row to select an Access Definition for the corresponding Source File. If the rows are equal, the Access Definition you select is for Source 1, by default.

The fully qualified (*dbalias.creatorid.tablename*) table names in the selected Access Definition are matched with names in the browsed file. If no match is found, a two-part (*creatorid.tablename*) table name match is attempted, and finally, a table name only match. If no match is found, a message prompts you to select a different Access Definition.

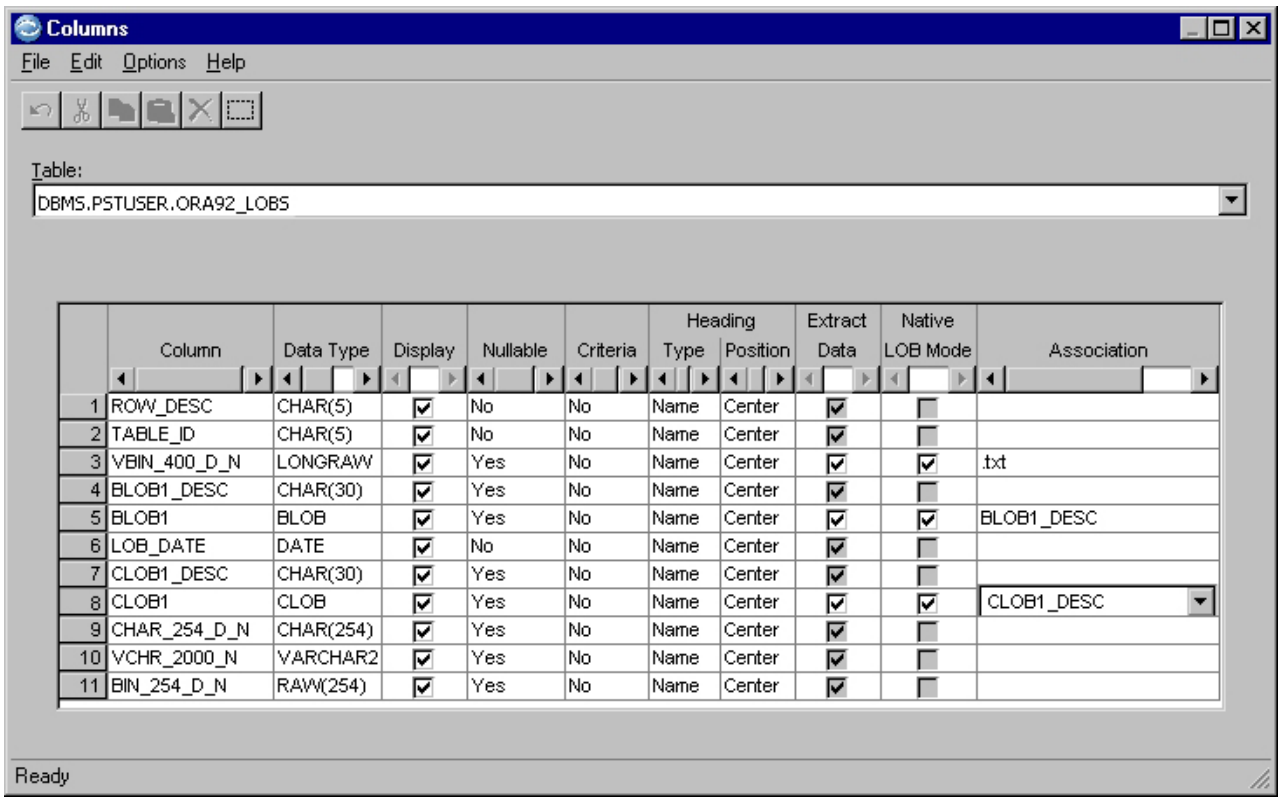
If you attempt to browse a LOB column for which an association is not found in the selected Access Definition, a message prompts you to open the Columns dialog to create one.

If you attempt to browse a LOB column that is not in a table referenced in the selected Access Definition, a message prompts you to select a different Access Definition. You can select a different Access Definition as often as required.

Note: When selecting or saving an Access Definition for a Compare File, the dialog will contain "Source 1" or "Source 2" as part of the title, for reference.

Columns

Click **Columns** from the submenu to display the Columns dialog.



The grid on the Columns dialog contains a column labeled **Association**. Use the **Association** column to associate a LOB column with the application required to view or edit the LOB data, in one of two ways:

- Type a file name extension in the **Association** column that corresponds to the type of LOB (for example, type the extension *.doc* to associate a LOB Word document with Microsoft Word).
- OR
- If the table contains a reference column used to identify the LOB data, you can enter the name of the reference column in the **Association** column. The first three characters in the reference column are used as the file name extension for the LOB data in the corresponding row of the LOB data column. The reference column must be a character-type column. Click the **Association** column to display a drop-down list of the names of character-type columns in the table, and select the reference column name.

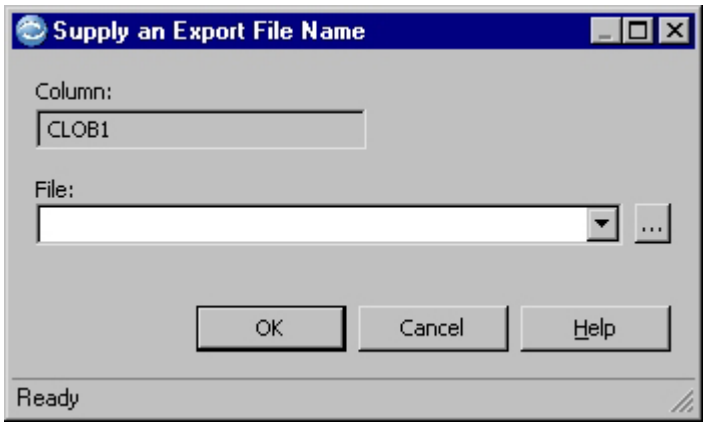
Note: The check box in the **Native LOB Mode** column corresponding to the LOB column must be selected in order to display the LOB in the associated application.

When you attempt to browse a LOB column associated with an application that is inaccessible from the workstation, Windows 2000 displays the Open with... dialog to enable you to select an accessible application. (Older versions of Windows may display an error message. To manually assign an accessible application to use, select **Options** from the Windows **View** menu, then select **File Types**.)

An Archive or Extract File created on the OS/390® mainframe does not contain an embedded Access Definition that is usable by Optim. Therefore, an Access Definition is created dynamically, when necessary, with DEFAULT as the DB Alias. Since DEFAULT is not a valid DB Alias for Optim, the Access Definition can only be used for LOB column associations.

Export LOB Data

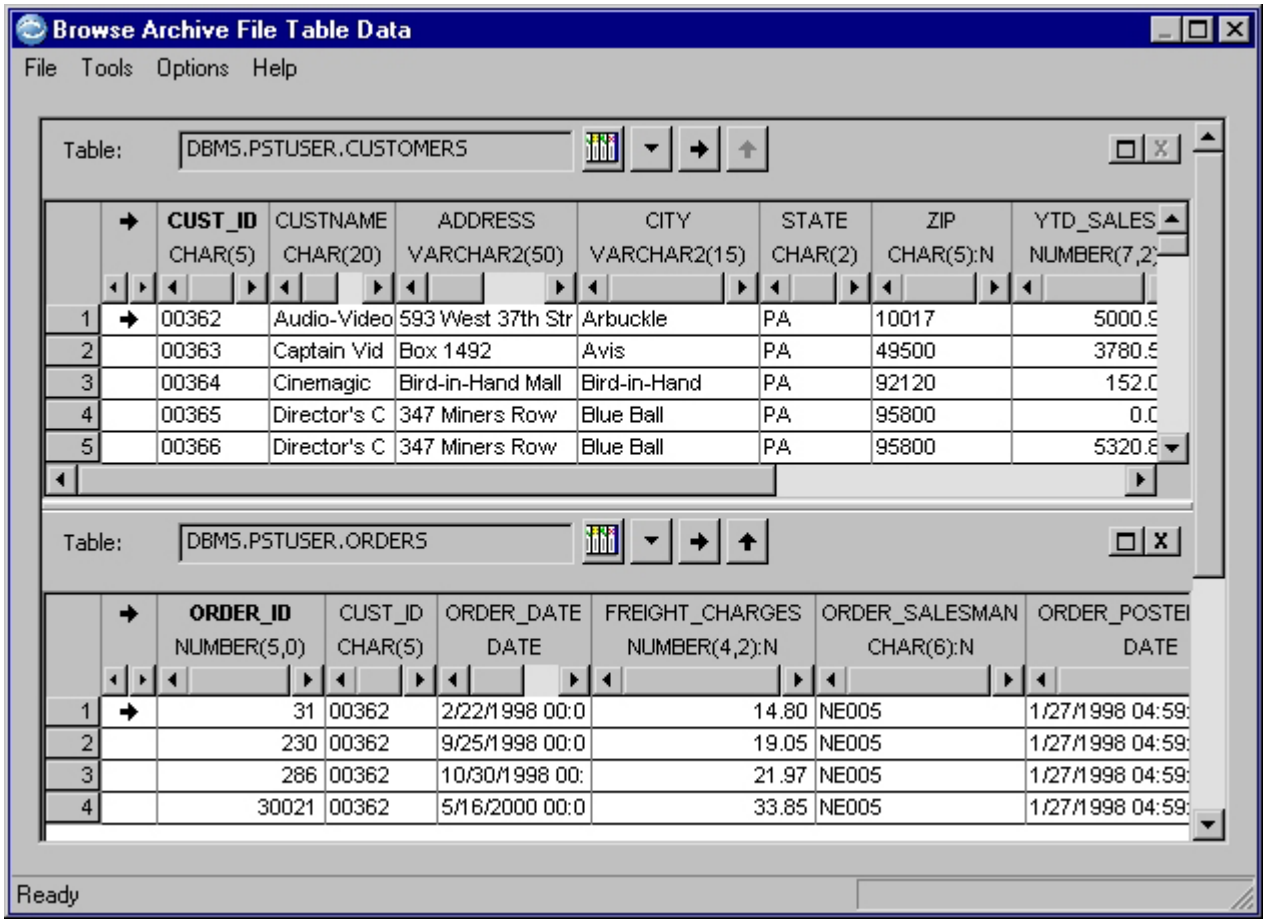
Right-click and select **Export LOB** to export LOB data to a file. Enter a name for the Export File.



Display Multiple Tables

You can join one or more tables to a table displayed in the Browse Table Data dialog. When you first open the Browse Table Data dialog, rows from the selected table are displayed.

Use the Join feature to display related data from other tables in the file. When you join a table, the related data is displayed in an additional browse window in the Browse Table Data dialog.



You can join several related tables to a single table, or join additional tables to a joined table. Each joined table displays in a new browse window in the dialog. In the example shown, the ORDERS table is joined to the CUSTOMERS table. Note that the displayed ORDERS rows are related to the CUSTOMERS row indicated by the Current Row Indicator.

Join Button

Click the Join

button in a browse window toolbar to join a related table. The Select Join Table dialog opens when you click the Join button.

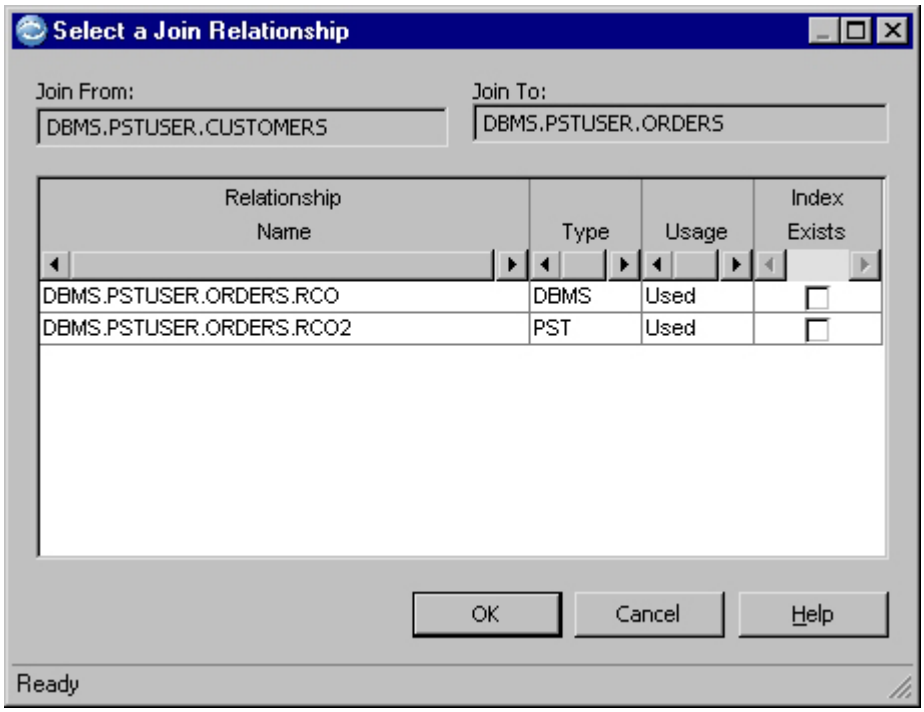
Select Join Table Dialog

The Select Join Table dialog lists tables in the Archive File or Extract File that are related to the table in the Browse Table Data dialog.

Select one or more tables from the list of table names. If a single relationship exists between the tables, the table joins automatically and related rows display in a new edit window. If more than one relationship exists for the table you select to join, the Select a Join Relationship dialog displays.

Specify a Relationship for Joining

If more than one relationship exists for a table you select to join, you must select the relationship to use from the list on the Select a Join Relationship dialog.



Open Relationship Editor

If only one relationship exists between the current table and an eligible join table, you can select **Open Relationship** from the shortcut menu to open the Relationship Editor. If you select a join table with more than one relationship, you can select **Open Relationship** from the shortcut menu on the Select a Join Relationship dialog to open the Relationship Editor.

Stack Tables

If a table is related to more than one table, you can join any or all of the related tables. If several tables are joined to a single table, the joined tables are "stacked" in a single browse window, in the order in which they were joined. By default, data from the most recently joined table is displayed, and other tables in the stack are hidden.

Use the drop-down box in the browse window for stacked tables to display data from one of the other tables in the stack. Click the arrow to display a list of stacked tables, then click a table name in the list to display data from that table in the browse window.

Note: When the stacked table displayed in the browse window does not have a related row, the Auto Switch feature automatically displays the next table in the stack. You can disable the Auto Switch feature on the **Browse** tab in Personal Options.

You can display any table in the stack and join other tables to any table in the stack. In many cases, a table is related to two or more tables, creating different paths for joining and browsing the data.

When a stacked table is displayed, all subordinate joined tables are also displayed. When a stacked table is hidden, all subordinate joined tables are also hidden.

Unjoin Tables

Click the Unjoin button  in the browse window toolbar to unjoin the table in the browse window, and all subordinate joined tables. If the table is part of a stack when you click unjoin, data from the next table in the stack populates the browse window.

Printing Options

When you browse a file, the information you are browsing determines the print options. Different print options are available from the Browse and Browse Table Data dialogs.

Browse Dialog

From this dialog, you can print the list of tables in the file, and the number of rows for each table. Select **Print** from the grid heading shortcut menu to print the list of tables in the file, and the number of rows for each table only.

Browse Table Data Dialog

From this dialog, you can print a list of rows displayed in the browse window. Right-click in the grid heading of a browse window and select **Print** from the shortcut menu to display the Windows Print dialog (for details, refer to Windows Help).. The data is printed in a columnar format, in the order displayed. Each row starts on a new line.

Save File Information as an Output File

Archive, Compare, Extract, and Control Files are stored in a proprietary format; they are not readable when opened with a text editor. However, you can use the Browse dialog to open a file, and then save it as an output file.

Choose to save the file in text (**.txt**) format or comma separated (**.csv**) format. You can save the information from one table, or several tables, each in a separate file. The file, once generated, can be opened in a text editor or imported into another application, such as a spreadsheet or database application. (Comma separated format is especially useful for importing into other applications.)

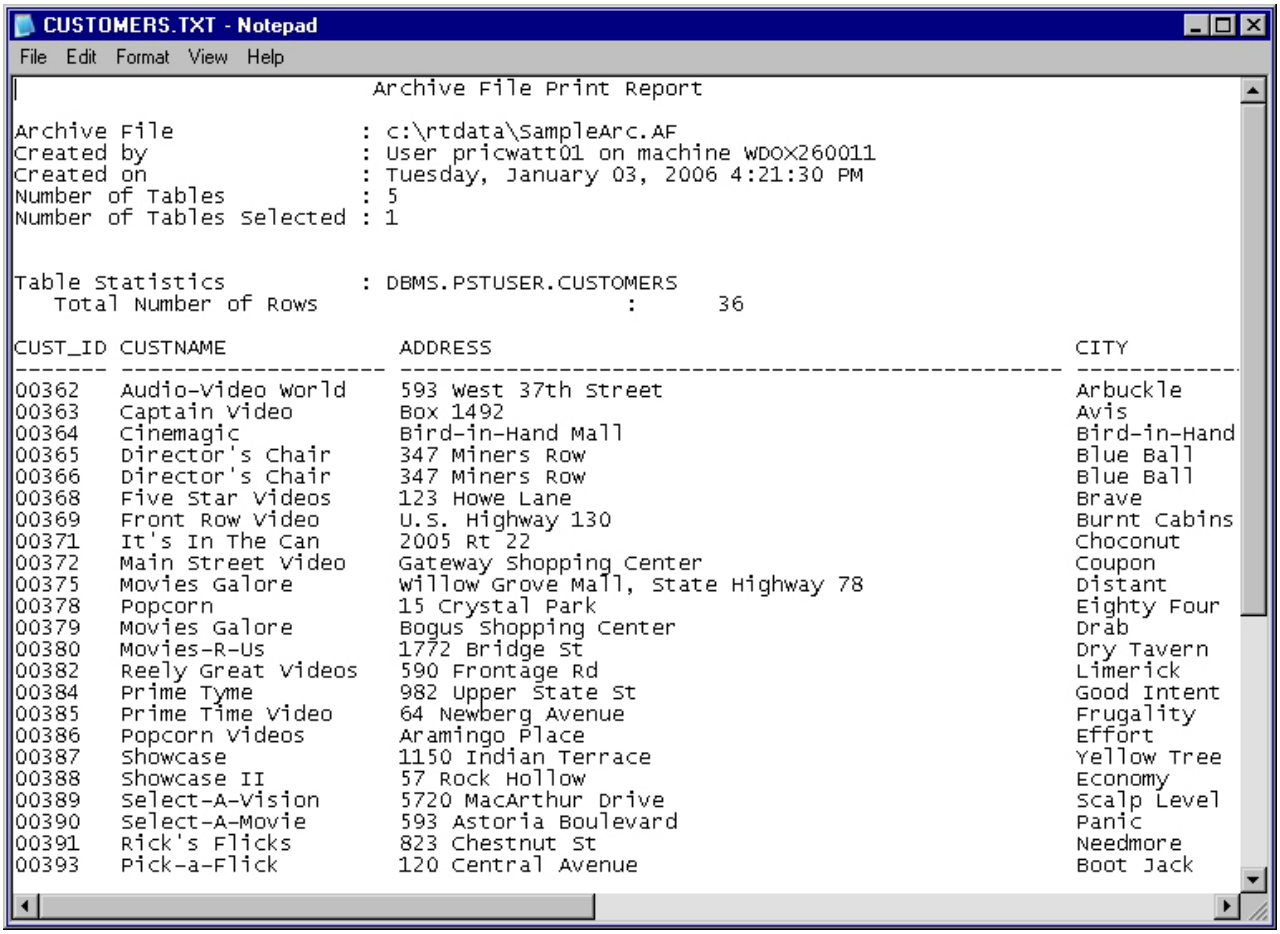
Note:

- LOB columns are not included in an output file, in either format.
- Additionally, you can use the Convert Request Editor to save joined table information in comma-separated format. For more information refer to the *Move User Manual*.

To save a file as an output file:

1. In the Browse dialog, select the table(s) you want to save.
2. Select **Save as Output File** from the **File** menu of the Browse dialog to open the Windows Save As dialog.
3. Assign a file name, choose a format and save the file for each selected table.

The next example is a portion of a text document version of a Control File. If you select more than one table to include in the text document, the information for each table is separated by a header.



Chapter 16. Export and Import

Use the Export and Import Utilities in Optim to migrate definitions of Optim Directory objects — including Access Definitions, Table Maps, Column Maps, Primary Keys, DB Aliases, Relationships, Calendars, and Convert, Delete, Extract, Insert, and Load Requests — from one Optim Directory to another.

Importing objects in UNIX has special considerations. See "Import for UNIX" on page 328.

To connect to a Optim Directory prior to exporting or importing, select **Optim Directory** from the **File** menu on the main window to open the Optim Directory window.

The Export and Import Utilities eliminate the need to recreate Optim objects manually and, more importantly, they promote consistent, reliable data handling. You migrate object definitions between Optim Directories in two steps:

1. Export

Copy object definitions from the Optim Directory to an Output File. You can reuse this file to import object definitions to any number of Optim Directories. You must have read access to view and export secured objects.

2. Import

Copy object definitions from a file to the current Optim Directory. The Input File for Import is the Output File from Export.

Export Files

The output from the Export Utility is a text file that you can view using an appropriate text editor, such as NotePad. If objects are exported from a multi-byte or unicode-enabled Optim Directory, any text editor that you use must be compatible with UTF-8.

Although not encouraged, you can edit object definitions in the text file. Deviations from the format required by the Import Utility or inserting or removing special characters, may cause errors. Do not use wordwrapping.

Secured Definitions

A secured-when-saved object is automatically secured, when imported, by an ACL modeled after the Optim Object Template ACL.

When you import a secured object, it is not secured in the importing directory unless the ACL for the object has been imported using the Import Security Definitions Utility or the object is secured automatically with the Optim Object Template ACL. If the object is not secured after import, you can define an ACL or import the ACL that secured the object.

If a Optim Object Template ACL has not been defined, you must define an ACL for each object that is secured automatically.

If Functional Security is enabled, you must have Create privilege for objects you import.

Export

The Export Utility copies Optim object definitions to an Output File. In a Windows environment, you can export all or specific definitions for each object type, using the graphical user interface or the command line. The command line utility also allows you to export object definitions in a Optim Directory residing on a UNIX- or Linux-based Server directly, without having to initiate processing from a Windows workstation.

Export Using the Interface

In a Windows environment, from the graphical user interface, you can use the Export Utility to copy Optim object definitions to an output file.

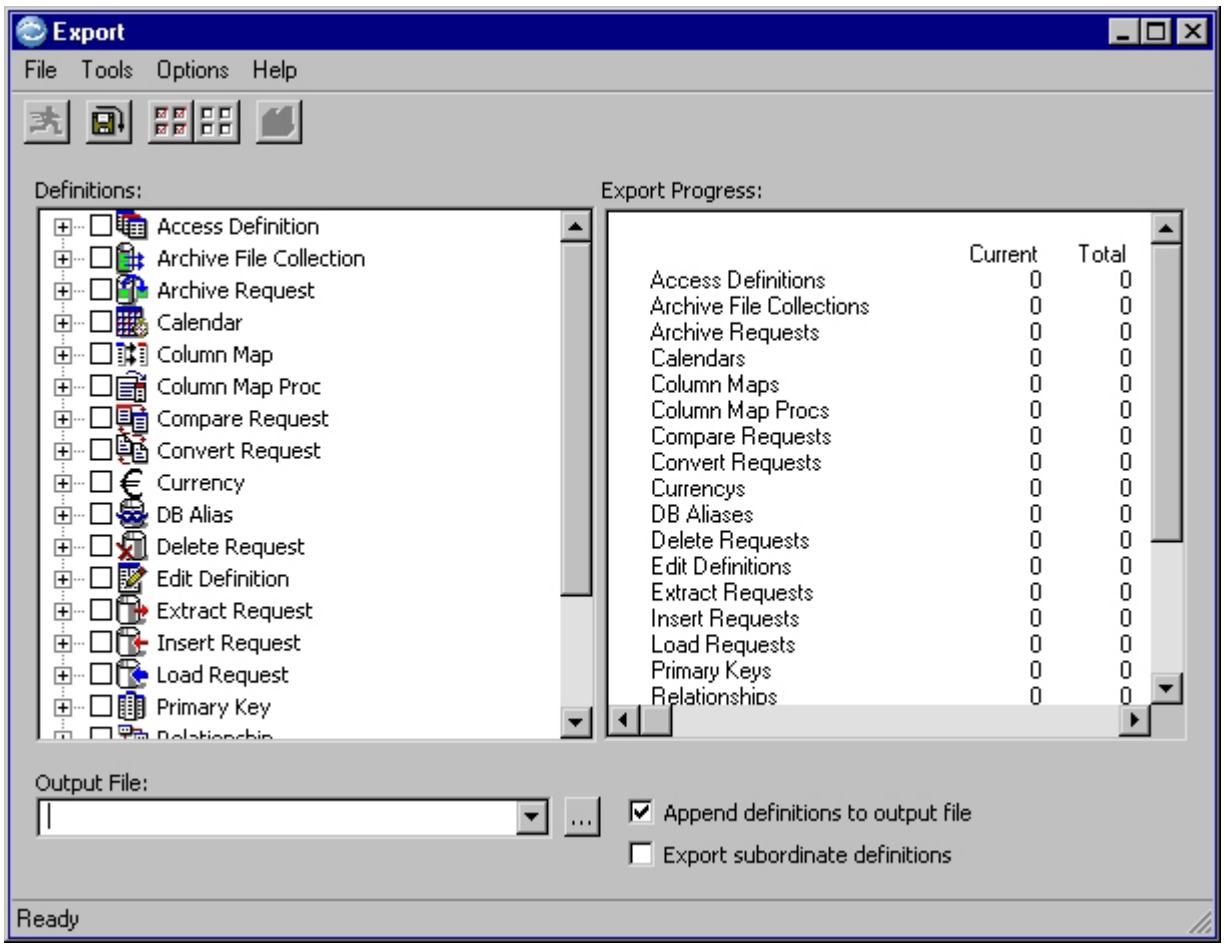
To use the Export Utility:

1. In the main window, select **Export** from the **Utilities** menu.
2. In **Definitions**, select the objects to export.
3. Enter an Output File name.
4. Select or clear options to append to an existing file and export subordinate definitions.
5. Choose **Run** from the **File** menu.
6. Monitor progress in the Export Progress pane.
7. When Export Processing is finished, choose Show Process Log from the **Tools** menu to review or print the Export Process Log.

These steps are the minimum required to export objects. The Output File to which you export objects is used as the input for Import Processing.

Export Dialog

When you open the Export dialog, the tree hierarchy on the Definitions pane is populated with information from the Optim Directory.



You can select the check box for an object type in order to export all definitions of that type or expand the list and select definitions to be exported by qualifier or name. **Tools** menu commands allow you to select or de-select all definitions.

Output File

The name of the Output File. The Output File is a text file and is given a **.txt** extension automatically. Other extensions may be specified, or the file can be designated without an extension. The Output File is used as the Input File for Import Processing.

Append definitions to output file

Option to append definitions to an existing Output File or replace the contents of the file.

- Select the check box to append all selected definitions to the Output File. If the name of a definition matches one in the Output File, the check box for the definition is shaded and you cannot export it unless you choose **Deselect All Definitions** from the **Tools** menu to clear check boxes for all definitions. You can then select definitions to be appended to the file. When imported, the last instance of a definition in the file supersedes all others.
- Clear the check box to replace the entire contents of the Output File with the exported definitions. The Utility prompts you for confirmation before the Output File is overwritten.

Export subordinate definitions

Option to import subordinate objects. Select the check box to export subordinate objects with the objects. Note that Local definitions are always exported (e.g., a Local Access Definition is exported as part of an Extract Request). Objects and possible subordinate objects are:

Primary Object	**Subordinate Object(s)**
Access Definition	Relationships
Archive File Collection	*none*
Archive Request	Access Definition Storage Profile Report Request
Calendar	*none*
Column Map	Column Map Procedures
Column Map Procedure	*none*
Compare Request	Access Definition Table Map Column Maps Column Map Procedures Report Request
Convert Request	Table Maps Column Maps Column Map Procedures Report Request
Currency Table	*none*
DB Alias	*none*
Delete Request	*none*
Edit Definition	Access Definition Report Request
Extract Request	Access Definition
Insert Request	Table Map Report Request Column Maps Column Map Procedures Report Request
Load Request	Table Maps Column Maps Column Map Procedures Report Request
Primary Key	*none*
Relationship	*none*
Report Request	*none*
Restore Request	Insert Requests Load Requests Table Maps Column Maps Column Map Procedures Report Requests
Storage Profile	*none*
Table Map	Column Maps Column Map Procedures

Menu Commands

In addition to standard **File** and **Edit** options, you can use the following commands from the **File** and **Tools** menus on the Export dialog:

File Menu

Set as Default
Save the setting for **Export subordinate definitions** and reuse it when the dialog is redisplayed.

Tools Menu

Select All Definitions
Select check boxes for all listed definitions. This command is especially useful when you want to export all or most of the definitions.

Deselect All Definitions
Clear the check boxes for all listed definitions, including shaded and/or selected check boxes.

Show Process Log
Display the Export Process Log generated by the last execution of Export.

Build the Output File

The Export Utility allows you to develop a strategy to meet the needs of your installation. For example, you may want to copy one Extract Request with all subordinate objects and another Extract Request with none of the subordinate objects. As another example, you may want to export an Access Definition from one Optim Directory, supplement it with relationships from a second Optim Directory, and include Table Maps from a third.

For either example, you would run Export several times. If certain objects are intended for one Optim Directory and others for another, you might direct the exported objects to different Output Files, and import each file as needed. Alternatively, you can use Export to direct the exported objects to a single Output File by selecting the **Append definitions to output file** check box for each Export. Then, you can import all of the objects in one step, using the single Output File as input.

Run Export

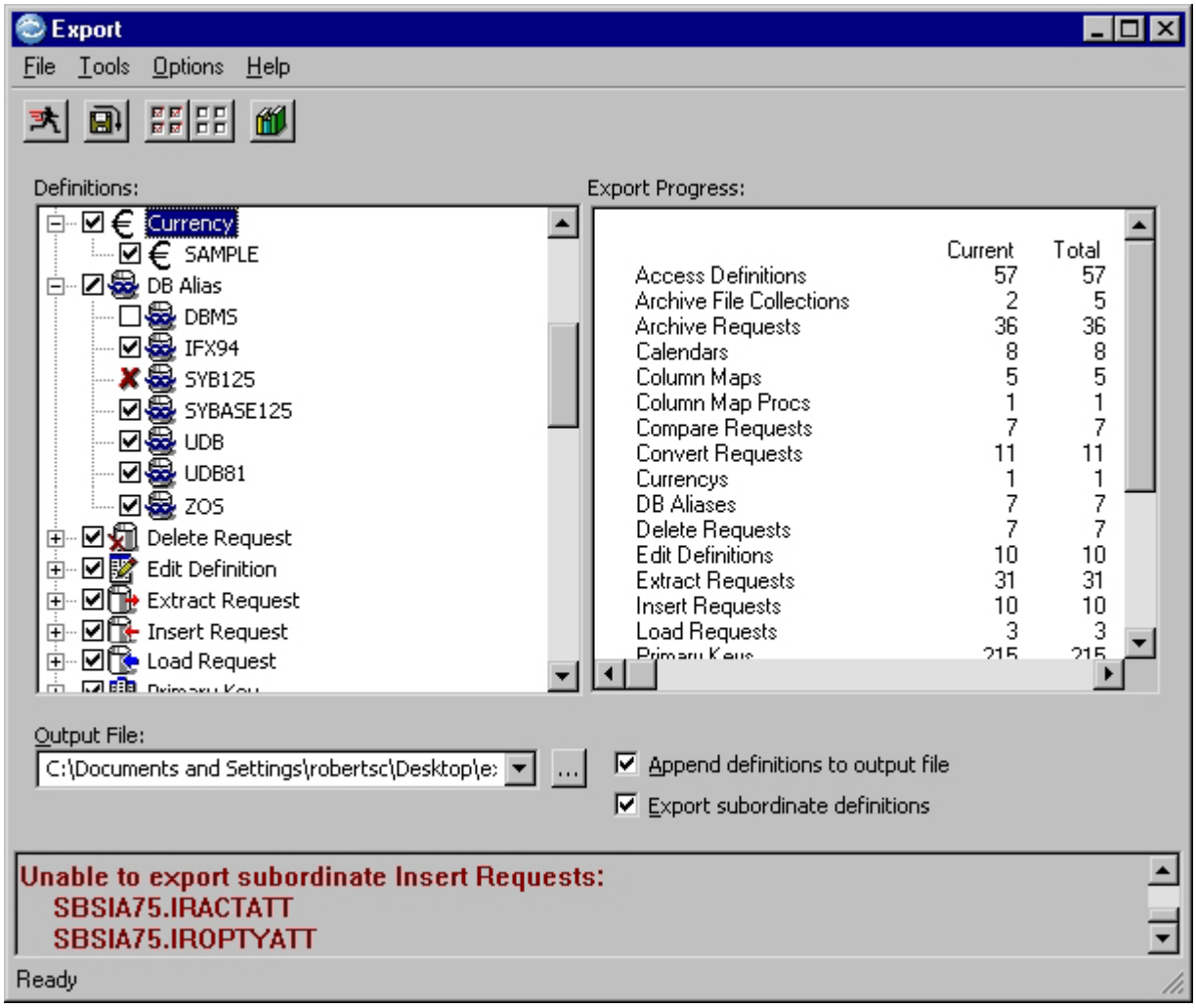
To start an export in the Export Utility, choose **Run** from the **File** menu. During Export Processing, the Export Progress pane displays the number of objects of each type that are exported and the number of errors encountered.

The status bar displays information about the current object that is processed. When Export Processing is finished, the status bar displays the message: “Ready.”

Export Errors

If errors occur during Export Processing (for example, a dropped object is selected for export), an error message is displayed. Objects for which processing errors occur are also represented visually by a red “X” and errors are written to the Export Process Log. You can check the Export Process Log for diagnostic information about errors.

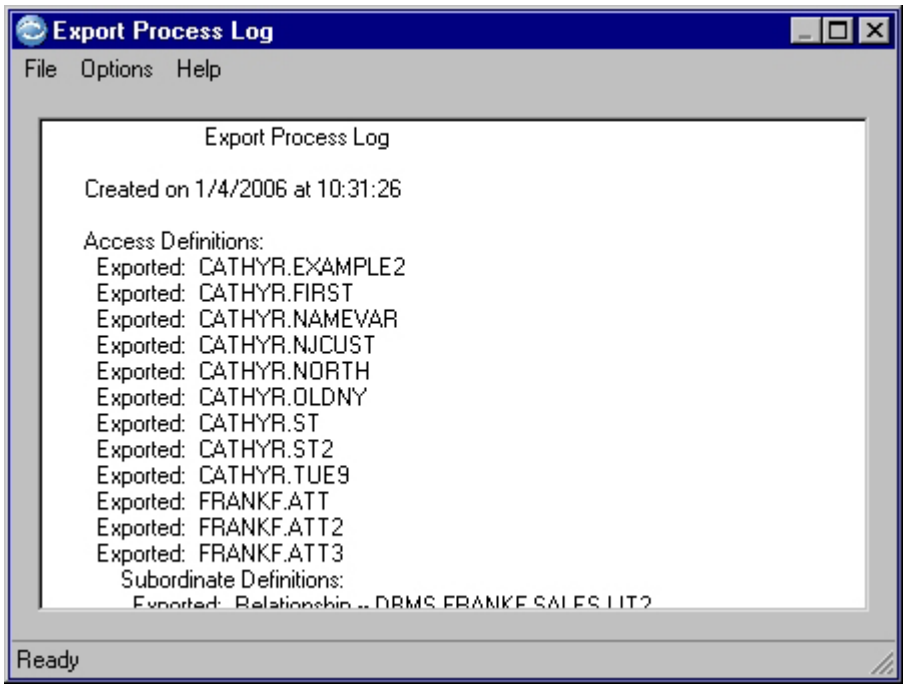
The following dialog shows error indicators after an Export:



If some object definitions fail to be exported due to errors (for example, the object was dropped after the list was generated), you can check the definitions and review the Export Process Log.

Export Process Log

When the Export is complete, select **Show Process Log** from the **Tools** menu to display the Export Process Log.



This log displays the following details:

Creation Date
Date and time the Export Process Log was created.

List of errors
Explanatory text for each error if errors were encountered.

List of exported objects
Names of the exported object definitions, grouped by object type.

Print

Print the log by choosing **Print** from the **File** menu. Each execution of Export clears the log file before information for the current execution is written. Previous log information is not retained.

Command Line Export

You can use the Command Line to export object definitions. This command line processing is available in the Windows, UNIX, and Linux environments.

Guidelines

The following guidelines apply:

- EXPORT must be prefixed by a forward slash (/) or dash (⌂). Other keywords may be prefixed by a forward slash (/) or a dash (⌂) but a prefix is not required. For example, PST, /PST, and -PST are equally valid as keywords.
- Keywords may be entered in mixed case and in any order, but must be separated by one or more spaces. Do not use commas.
- Keywords must be separated from arguments by an equal sign (=) or a colon (:).
- An argument that includes spaces must be enclosed in single or double quotes.
- An argument that includes a macro must be enclosed in double quotes.
- Comments must begin on a separate line and start with two forward slashes (//).

- You can include all keywords and arguments for an EXPORT task in a text file, and reference this parameter file on the command line.

Syntax Conventions

The syntax conventions used to describe the command line export are:

KEYword
Uppercase letters indicate the minimum abbreviation for a keyword, while lowercase letters indicate the optional portion. In practice, a command line keyword can be specified in UPPER, lower or MiXed case.

text Variable text is shown in lowercase italics.

( ) Statement delimiter to group a series of qualifiers for a parameter.

[ ] Indicates an optional parameter.

{ } Indicates a choice of two or more settings from which one (and only one) must be selected.

| Separates options.

Export Syntax and Keywords

Use the following command line syntax to export Optim Directory object definitions.

Note: If viewing this document in Portable Document Format (PDF), you can select a keyword in the following syntax to navigate to the explanatory text. Use **Go To Previous View** to return to the syntax on this page.

```
PR0CMND /EXPORT { @parameterfilename |
   [ { PSTdirectory= | DIRectory=} pstdirectory ]
   [ Output=rptfilename[ + | – ]
   [ SUBordinates[ + | – ] ] [ CONTinueonerror[ + | – ] ]
   TYpe=objtype
   NAme=obj.name
   File=expfilename[ + | - ]}
```

PR0CMND	Type PR0CMND to initiate command line processing. Note that the character following PR is the number 0 (zero).	
/EXPORT	Command to run the Export Process. Specify **/EXPORT** or **–EXPORT**.	
@parameterfilename	The name of an optional parameter file containing all remaining parameters, as follows. All keywords and arguments in the parameter file must be on a single line.	
PSTdirectory= | DIRectory=	The Optim Directory from which objects are exported. The current Optim Directory is the default.	
	pstdirectory	Name of the Optim Directory for the request.
Output=	The Export Process report.	
	rptfilename	Name of the file for the process report. Provide the full directory path to save the file in a directory other than the default Data Directory (specified in Personal Options). You may append one of the following to the file name.
		+ Option to append the report to an existing file.
		– Option to overwrite an existing file (default).
SUBordinates	Option to export definitions of objects subordinate to objects indicated by **TYPE** and **NAME**.	

	+	Export subordinate objects. See “Export subordinate definitions” on page 322 for a list of subordinate objects.
	–	Do not export subordinate objects (default).
CONTinueonerror	Option for continued processing upon an error condition.	
	+	Continue processing upon an error condition.
	-	Stop processing upon an error condition (default).
TYpe=	Type of object(s). Export all object definitions of the indicated TYPE that match the NAME parameter.	
	All	All object types.
	ACcessdefinition | AD	Access Definitions.
	AFC | COLLection	Archive File Collections
	ARCHive	Archive Request Definitions.
	CALendar	Calendar Definitions.
	COLMAP | CM	Column Map Definitions.
	COLMAPProc | CMProc	Column Map Procedure Definitions.
	COMPare	Compare Request Definitions.
	CONvert	Convert Request Definitions.
	CURrency	Currency Definitions.
	DBAlias	DB Alias Definitions.
	DELete	Delete Request Definitions.
	EDit	Edit Request Definitions.
	EXtract	Extract Request Definitions.
	LOad | LD	Load Request Definitions.
	PRImarykey | PRIMKEY | PK	Optim Primary Key Definitions.
	RELationship | RL	Optim Relationship Definitions.
	STOrageprofile | SP	Storage Profile Definitions.
	TABlemap | TM	Table Map Definitions.
	UPin | UPdate |INserT	Insert Request Definitions.
	REPort | REPT | RPT	Report Request Definitions.
	REStore	Restore Request Definitions.
NAme=	Name(s) of object(s) to export.	

	obj.name	Object name. Provide the full name or a pattern, using wild cards. When using a pattern, the naming convention must match that of the objects you wish to export. For example, specify TYPE=ALL NAME=PSTDEMO.% to export all Access Definitions, Table Maps, Column Maps, Column Map Procedures, Table Maps, and Process Request definitions with names qualified by PSTDEMO. DB Aliases, Primary Keys, Relationships, etc. must be exported in separate processes, using NAME criteria appropriate to the naming conventions for these objects.	
File=	Name of file for exported objects.		
	expfilename	Name of new or existing file. Provide full path if file is not in the default Data Directory specified in Personal Options. If you provide the name of an existing file, you can append one of the following to the file name:	
		+	Append definitions to the file.
		-	Overwrite any data in the file (default).

Import

The Import Utility copies object definitions from the specified Input File to the current Optim Directory. (The Output File generated by Export is used as the Input File.)

To Use the Import Utility:

1. In the main window, choose **Import** from the **Utilities** menu.
2. Specify an Input File name.
3. Select options on the **Process**, **Options**, **DB Aliases**, and **Objects** tabs.
4. Choose **Run** from the **File** menu.
5. Monitor progress on Import Progress.
6. Choose **Show Process Log** from the **Tools** menu to review or print the Import Process Log.

Import for UNIX

You can use the Import utility to import objects in a UNIX environment. The Import Utility copies object definitions from the specified Input File to the current Optim™ Directory.

The Output File generated by Export is used as the Input File. See “Export” on page 320 for information about creating a file to use as an Input File. These conditions apply when importing in a UNIX environment:

- Determine the DB Alias, Optim Directory, and Creator ID for the objects you want to import. In the pstlocal.cfg and pstserv.cfg files, ensure that definitions exist for both the source and target DB Alias, Optim Directory, Creator ID and password. See the *Installation and Configuration Guide,* Appendix A - Install and Configure the Server under UNIX or Linux.
- Before you attempt to import any other objects, import only the DB Alias in a separate process.
- z/OS object definitions, Primary Keys, and Foreign Keys are unavailable for import.

See “Import” for general information about the Import process. Details on using the Import utility are in “Import Using the Interface” on page 329.

Import Using the Interface

The Import dialog has four tabs. Each tab and menu command available on the dialog serves a unique purpose.

Process
Select object types, identify the Input File, and provide parameters for Import Processing.

Options
Provide parameters needed to import definitions obtained from the MVS platform.

DB Aliases
Map DB Aliases for the imported objects.

Objects
Designate names to be assigned to the imported objects.

Menu Commands

In addition to standard **File** and **Edit** options, you can choose the following options from the **File** and **Tools** menus on the Import dialog:

File Menu

Set as Default
Save your entries on the Import dialog as the default specifications. The settings for the following are saved:

- **Overwrite existing definitions** on the **Process** tab.
- **Continue import if error(s)** on the **Process** tab.
- **Default Qualifier** on the **Options** tab.
- **Specify two-part Access Definition name** on the **Options** tab.

If no dialog defaults are established, the Import dialog displays application default settings for the check boxes.

Tools Menu

Select All Definitions
Select all object definitions in the Input File.

Note: If you select individual definitions, the subordinate definitions are not included automatically. To import subordinate definitions, you must select them explicitly from the **Definitions** list.

Deselect All Definitions
Clear the check boxes that are shaded and selected, making all definitions in the Input File available for selection.

Note: Import overwrites the definition in the Directory if the object exists and **Overwrite existing definitions** is selected.

Show Process Log
Display the Import Process Log generated by the last execution of Import.

Read input file
Refresh the display with information from the Input File. For example, if an Input File is opened simultaneously in **Export** and a text editor, the file is locked and Import Processing cannot proceed. When the application or dialog causing the sharing violation is closed, choose the Read input file command to reread its contents.

Run Import

To Import definitions, choose **Run** from the **File** menu. The Import Progress pane displays the progress of the processing.

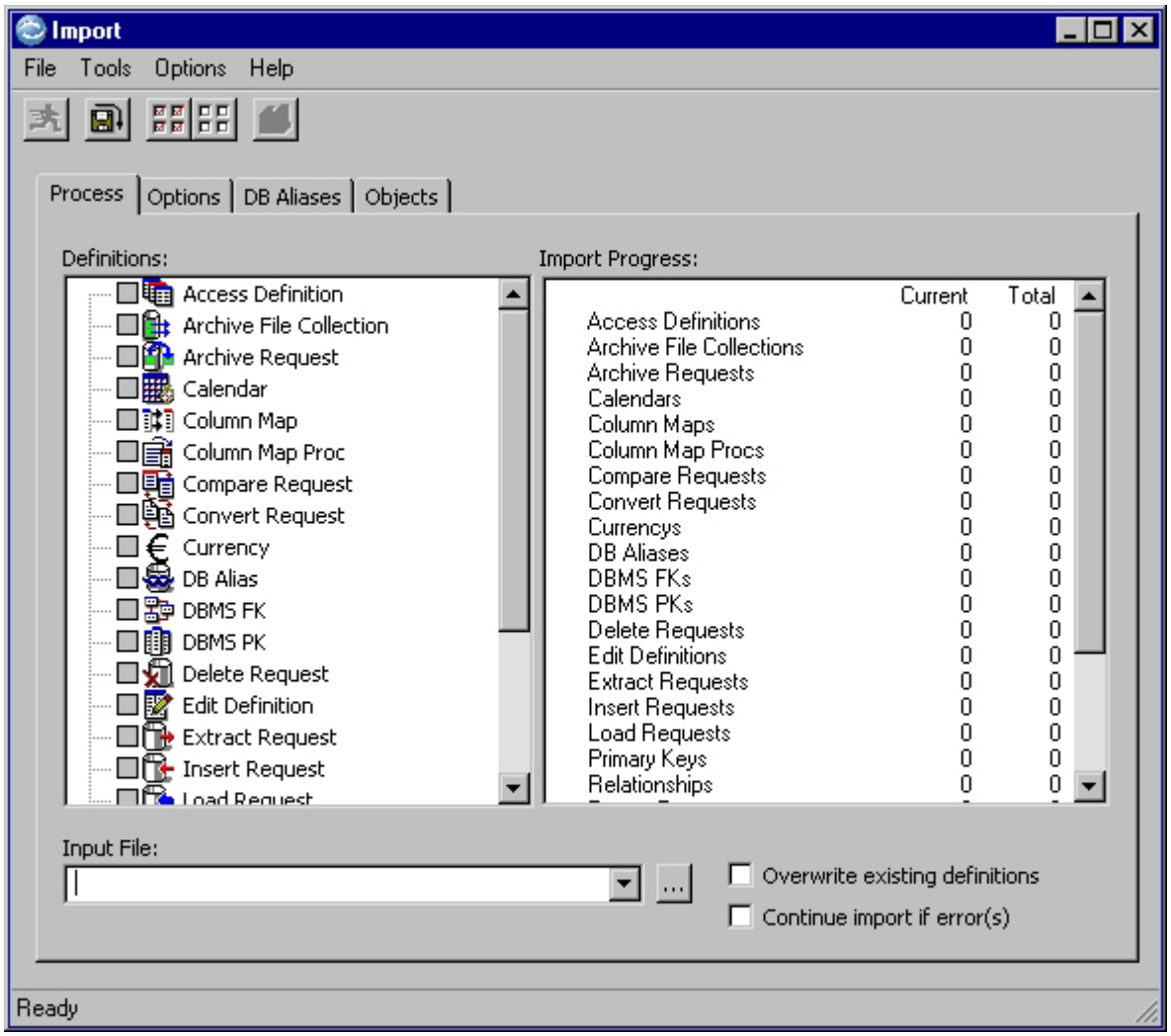
Note: To run the Import Utility, at least one available object must be selected.

When Import Processing is finished, the status bar displays the message: "Ready."

Process Tab

The tab is populated with the defaults you have specified. If you have not established defaults, the Import dialog displays application default settings for the check boxes, and the **Input File** box is populated with the name of the last Input File designated in the Import dialog.

Note: If the Input File is unavailable due to a sharing violation, select **Read input file** on the **Tools** menu to automatically revise the definitions list.



Definitions

The Import Utility populates the list of objects by:

- Identifying the objects in the Input File. If there are no objects of a specific type (e.g., no Access Definitions), the check box for the object type is shaded. Otherwise, you can expand to display a list of objects of the type by clicking on the plus (+) sign.
- Scanning the Optim Directory and identifying the objects in the Input File that exist in the Directory and those that do not. The check box to the left of each listed object is selected or not according to the **Overwrite existing definitions** setting.
  - If **Overwrite existing definitions** is not selected, the check boxes to the left of objects that exist in the current Optim Directory are shaded and selected and are unavailable for Import.
  - If **Overwrite existing definitions** is selected, all check boxes to the left of each definition are cleared and any definition can be selected. If a selected object exists in the Optim Directory, Import overwrites it.

At least one available object must be selected to run the Import Utility.

Import Progress

Statistics detail the current and total number of objects of particular types imported, and the current and total numbers of errors encountered (the "total" numbers are the composite counts for all Import Processes performed in the session). This display is updated during processing. The status bar displays information about the object being processed.

Input File

Specify an Input File. The Input File for Import is generally an Output File generated by Export (see "Export" on page 320 for details about creating this file).

- To select from a list of recent file names, click the down arrow or use the browse button. You may also copy a name into the box or type a name directly.
- If you do not provide a fully qualified path, the path from Personal Options is used.
- If no path is given in Personal Options, the current drive and directory are assumed.

Overwrite existing definitions

Indicate the action taken when the name of an imported object definition matches that of an object already in the current Optim Directory:

- To select any or all objects and overwrite existing definitions in the Directory, select the check box.
- To prevent overwriting objects, clear the check box. Duplicate definitions, indicated by check boxes that are shaded and selected, are not imported.

Note: The Import Utility never overwrites database object definitions. If a primary key or relationship name conflicts with a name in the database, an error message is written to the log file. Processing continues according to the specification for **Continue import if error(s)**.

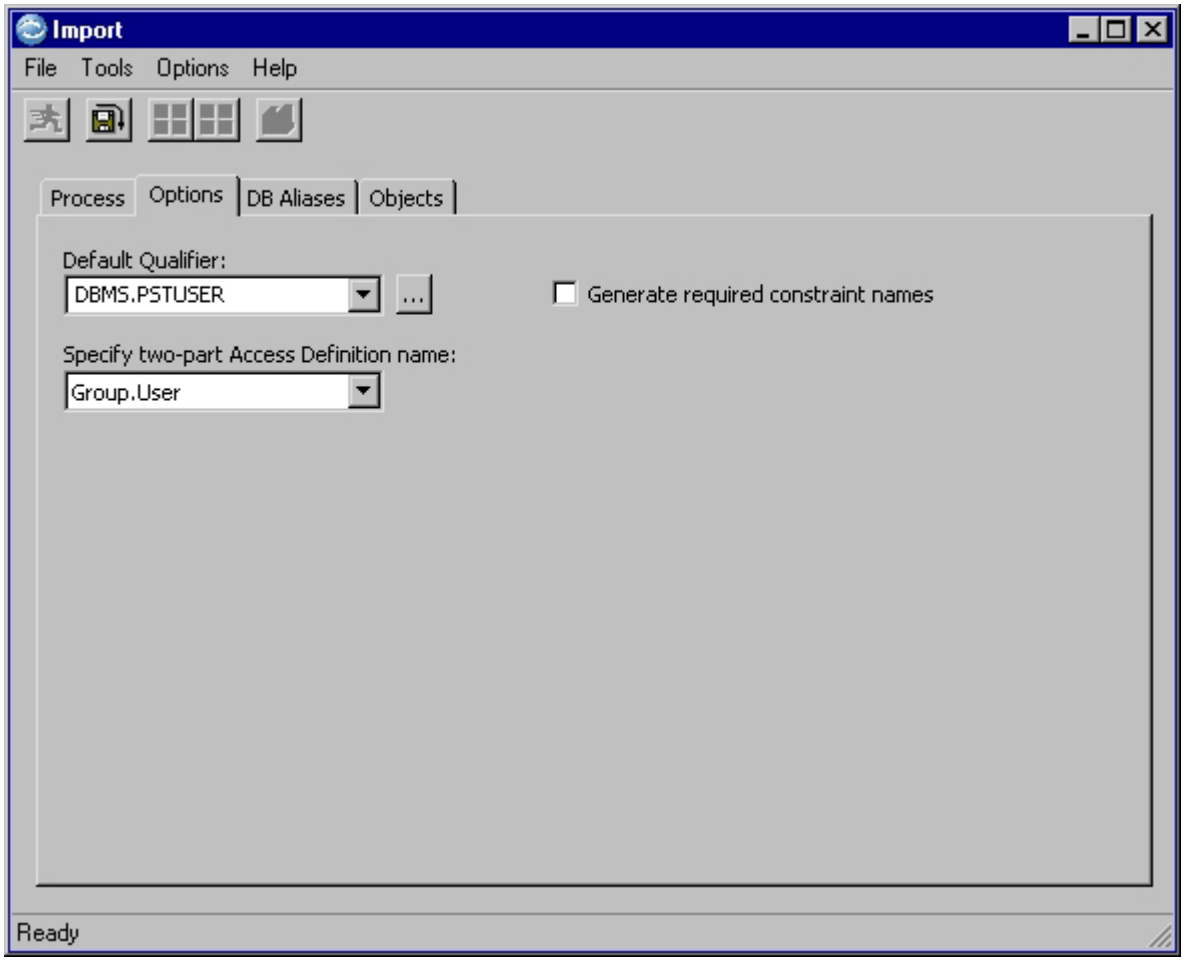
Continue import if error(s)

Indicate processing if an error occurs. Errors are written to the Import Process Log and displayed on the message bar.

- To continue processing if an error occurs, select the check box.
- To halt processing if an error occurs, clear the check box.

Options Tab

Use the **Options** tab to provide defaults for amending the names of object definitions imported from MVS.



Default Qualifier

Enter a DB Alias and Creator ID as the Default Qualifier. The Default Qualifier is assigned automatically to objects imported from MVS. Objects on these platforms do not use qualifiers. If the DB Alias is explicitly mapped on the **DB Aliases** tab, it overrides any default DB Alias specified on the **Options** tab.

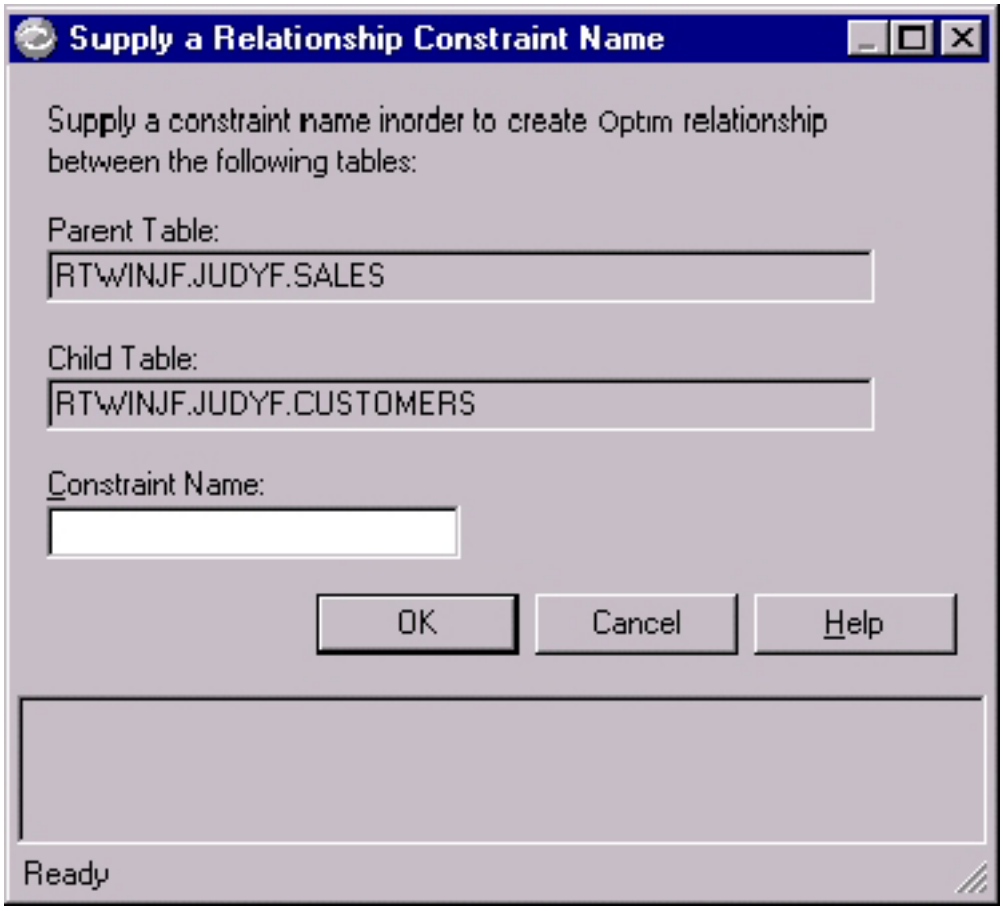
Generate required constraint names

Select this check box to automatically generate a constraint name for any object definition that has "Requires Constraint Name" as the last part of the qualified name. The first eight characters of the first column name for the child table are used as the constraint name, unless the name does not result in a unique Optim relationship name. Otherwise, one of the following characters is appended to the column name or used to replace the eighth character:

123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ.

Each character is applied in turn until a unique name is generated.

If you want to be prompted for a constraint name, clear the check box. The Supply a Relationship Constraint Name dialog is displayed when a name is required.



Specify two-part Access Definition name

Choose elements of a three-part Access Definition name (from an MVS definition) that should be used when the Access Definition is migrated to the two-part (*identifier.name*) Optim naming system.

Group.User
Use the **Group** and **User** portions. Drop the **Name** portion.

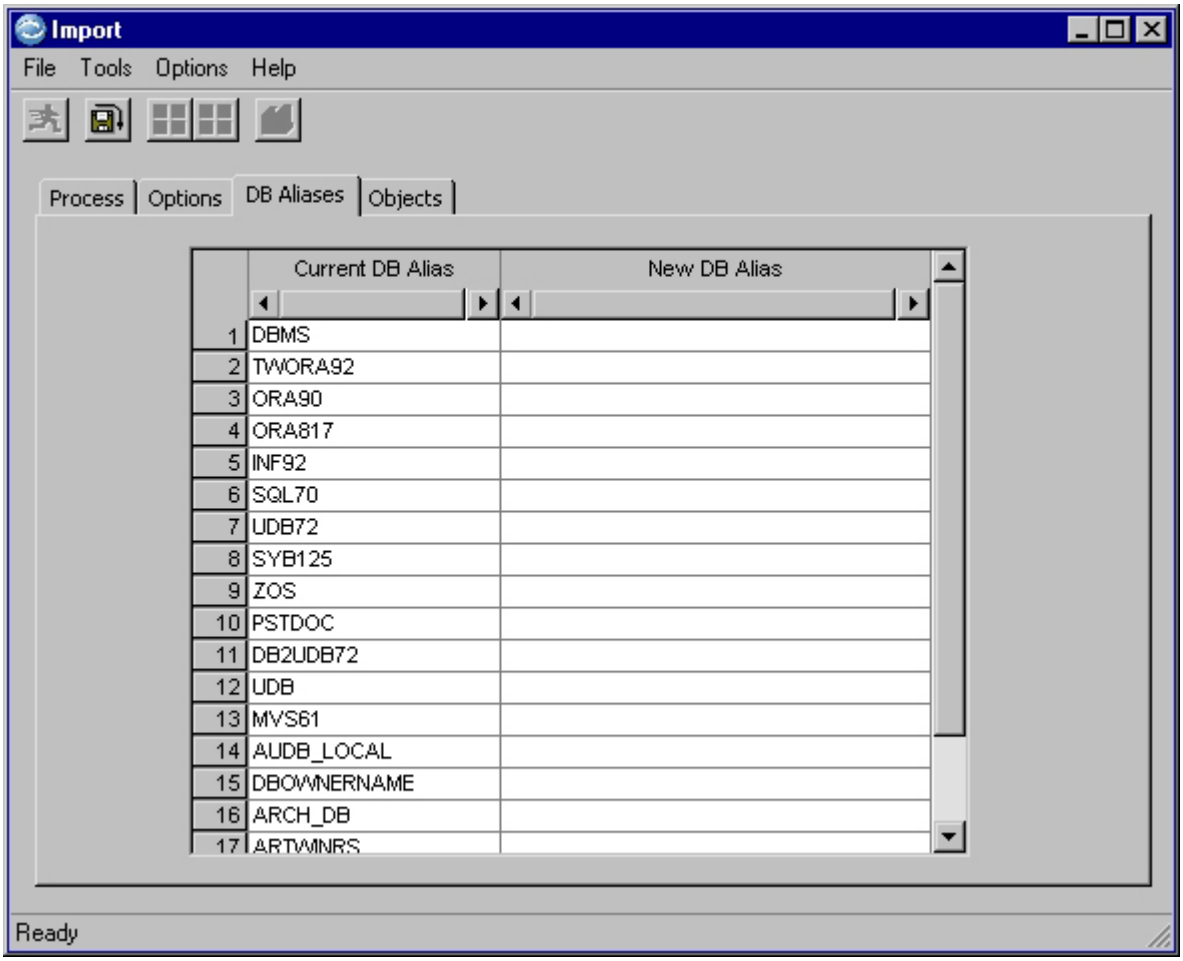
Group.Name
Use the **Group** and **Name** portions. Drop the **User** portion.

User.Name
Use the **User** and **Name** portions. Drop the **Group** portion.

If Access Definition names are explicitly mapped on the **Objects** tab, these names override the naming choice specified here.

DB Aliases Tab

Use the **DB Aliases** tab to map explicit DB Aliases for the imported objects.



Current DB Alias

This grid column is populated with the name of each DB Alias associated with objects in the Input File.

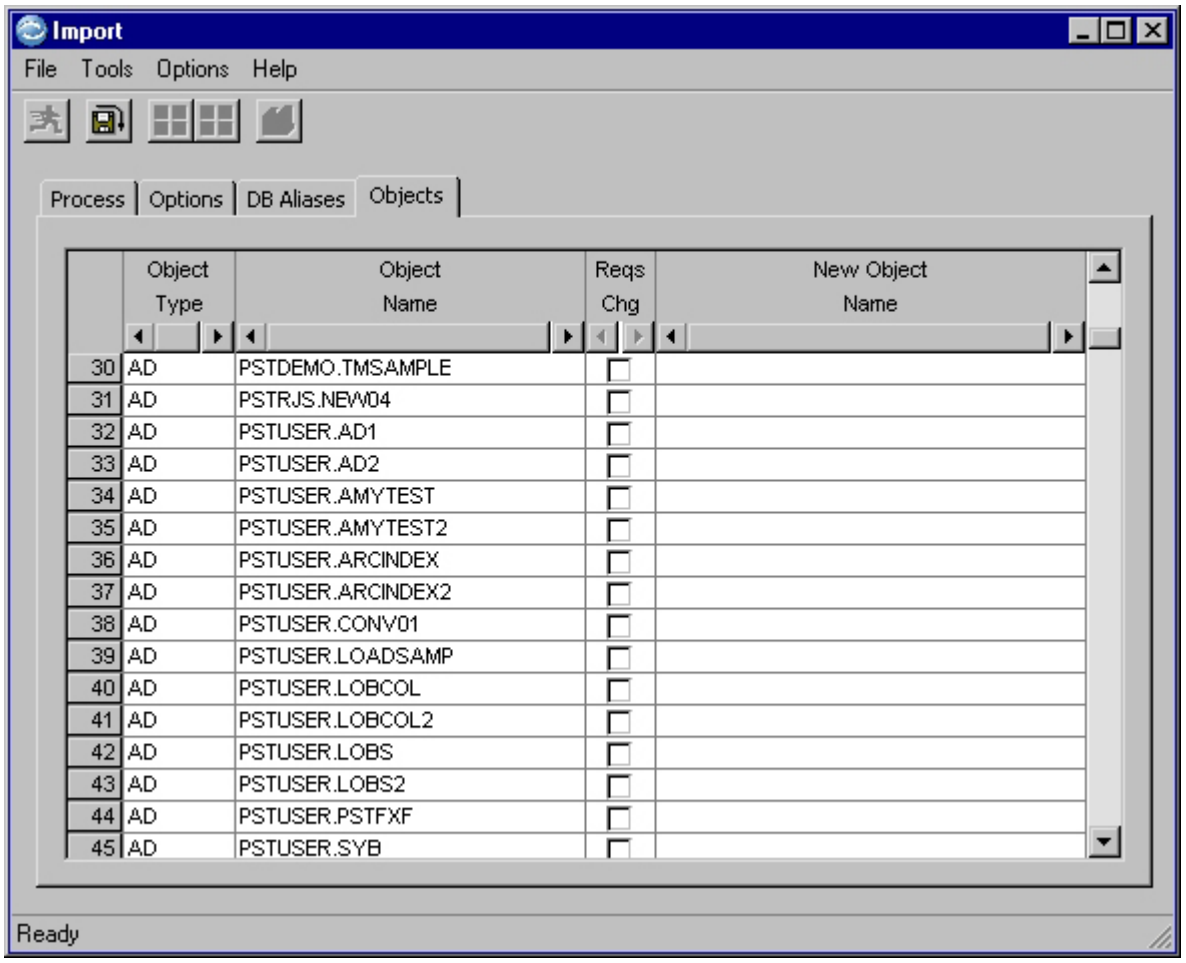
New DB Alias

You can provide a new DB Alias for objects to be imported. During Import Processing, the New DB Alias is substituted for the corresponding Current DB Alias. For example, the importing Optim Directory may include a DB Alias that corresponds to a database referenced in the import file by a different DB Alias name. You can specify the appropriate DB Alias for the importing Optim Directory and import the objects and use them without further change. Right-click to select from the following shortcut commands:

Copy Name	Right-click in the Current DB Alias column and select Copy Name to use the same name in the new DB Alias column.	
Populate	Clear	Removes all DB Alias names and inserts the default DB Alias names.
	Add	Inserts the default DB Alias names in any blank grid cells.
	Empty	Click to remove all DB Alias names (change to blanks).

Objects Tab

Use the **Objects** tab to review the objects in the Input File and specify new names for imported objects.



Object Type

Abbreviations for object types:

Abbreviation	Object
AD	Access Definition
AFC	Archive File Collection
ARCH	Archive Request
CALENDAR	Calendar
CM	Column Map
CMPROC	Column Map Procedure
COMP	Compare Request
CONV	Convert Request
CURRENCY	Currency Table
DBALIAS	DB Alias
DEL	Delete Request
ED	Edit Definition

Abbreviation	Object
EXTR	Extract Request
LOAD	Load Request
PK	Primary Key
REL	Relationship
REPT	Report Request
REST	Restore Request
STORPROF	Storage Profile
TM	Table Map
UPIN	Update or Update/Insert Request

Object Name

The names of the objects available for importing. The number of parts in the name depends on the object type and the platform of origin for the object.

Reqs Chg

If an object name requires changes before Import (e.g. the name already exists), the Utility selects the check box for that object. If you have specified values on the **Options** and/or **DB Aliases** tabs that resolve all discrepancies for the selected objects, you need not make changes on the **Objects** tab.

New Object Name

Specify a new name for the object. The name must comply with the naming conventions for that object type in Optim. For information on naming conventions, refer to the section that discusses the specific object type. Right-click to select from the following shortcut commands:

Copy Name
Right-click in the Object Name column and select Copy Name to use the same name in the New Object Name column.

Populate

Clear Removes all object names and inserts the default object names.

Add Inserts the default object names in any blank grid cells.

Empty Click to remove all object names (change to blanks).

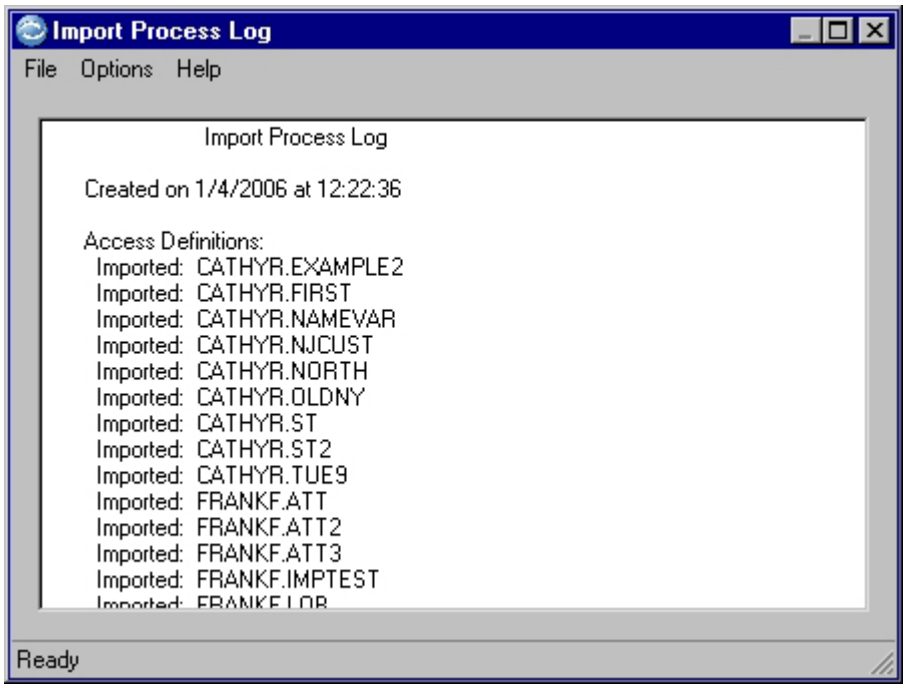
Import Errors

If Import encounters errors, processing continues according to the specification for the **Continue import if error(s)** option. Errors are displayed on the message bar and represented visually by a red "X" beside each object in error. Errors are written to the Import Process Log after Import Processing is finished. You can review and print the Import Process Log for diagnostic information about errors.

If some objects fail to be imported due to errors, check the specifications, and try Export and Import again. For details about the format of the Export/Import File, see "Output/Input File Format" on page 339.

Import Process Log

When Import Processing finishes, select **Show Process Log** from the **Tools** menu to display the Import Process Log.



Creation Date
Date and time the Import Process Log was created.

List of errors
Explanatory text for each error if errors were encountered.

List of imported objects
Names of the imported object definitions, listed by object type and name.

Print the log by choosing **Print** from the **File** menu. Each execution of Import clears the log before information for the current execution is written. Previous log information is not retained.

Command Line Import

You can use the Command Line Utilities to export and import object definitions from one Optim Directory to another. Command line processing is available in all Optim environments.

Guidelines

The following guidelines apply:

- IMPORT must be prefixed by a forward slash (/) or dash (⌂). Other keywords may be prefixed by a forward slash (/) or a dash (⌂) but a prefix is not required. For example, PST, /PST, and -PST are equally valid as keywords.
- Keywords may be entered in mixed case and in any order, but must be separated by one or more spaces. Do not use commas.
- Keywords must be separated from arguments by an equal sign (=) or a colon (:).
- An argument that includes spaces must be enclosed in single or double quotation marks.
- An argument that includes a macro must be enclosed in double quotation marks.
- Comments must begin on a separate line and start with two forward slashes (//).

When importing in a UNIX environment, the following conditions apply:

- Determine the DB Alias, Optim Directory, and Creator ID for the objects you want to import. In the pstlocal.cfg and pstserv.cfg files, ensure that definitions exist for both the source and target DB Alias, Optim Directory, Creator ID and password. See *Installation and Configuration Guide*, Appendix A - Install and Configure the Server under UNIX or Linux.
- Before you attempt to import any other objects, import only the DB Alias in a separate process.
- z/OS object definitions, Primary Keys, and Foreign Keys are unavailable for import.

Syntax Conventions

KEYword
Uppercase letters indicate the minimum abbreviation for a keyword, while lowercase letters indicate the optional portion. In practice, a command line keyword can be specified in upper, lower, or mixed case.

text Variable text is shown in lowercase italics.

( ) Statement delimiter to group a series of qualifiers for a parameter.

[ ] Indicates an optional parameter.

{ } Indicates a choice of two or more settings from which one (and only one) must be selected.

| Separates options.

The syntax conventions used to describe the command line import are:

Import Syntax and Keywords

Use the following command line syntax to import Optim Directory object definitions.

Note: If viewing this document in Portable Document Format (PDF), you can select a keyword in the following syntax to navigate to the explanatory text. Use **Go To Previous View** to return to the syntax on this page.

```
PR0CMND /IMPORT INput=impfilename
   [ { PSTdirectory= | Directory=} pstdirectory ]
   [ Output=rptfilename ]
   [ OVERWrite[ + | - ] ]
   [ ContinueOnError[ + | - ] ]
   [ DBAlias=(currentdbalias, newdbalias, ...) ]
```

PR0CMND	Type PR0CMND to initiate command line processing. Note: the character following PR is the number 0 (zero). **Note:** The command line IMPORT utility does not run in a UNIX or Linux environment.	
/IMPORT	Command to run the Import Process. Specify **/IMPORT** or **–IMPORT**.	
INput=	Name of the file that contains the objects to be imported.	
	filename	Name of file. Provide full path if file is not in the default Data Directory, specified in Personal Options.
PSTdirectory= | Directory=	The Optim Directory to which objects are imported. The current Optim Directory is the default.	
	pstdirectory	Name of the Optim Directory for the request.
Output=	File for the Import Process report.	
	rptfilename	Name of the file for the process report. Provide the full directory path to save the file in a directory other than the default Data Directory (specified in Personal Options). You may append one of the following to the file name.

		+	Option to append the report to an existing file.
		–	Option to overwrite an existing file (default).
OVERWrite	Option for processing object definitions of the same name and type.		
	+	Overwrite any object definitions in the Optim Directory.	
	⌂	Do not overwrite existing object definitions in the Optim Directory (default).	
ContinueOnError	Option for continued processing upon an error condition.		
	+	Continue processing upon an error condition.	
	-	Stop processing upon an error condition (default).	
DBAlias=	DB Aliases for object definitions you want to import.		
	currentdbalias	A DB Alias associated with object definitions in the input file.	
	newdbalias	A valid DB Alias for the object definitions after import.	
	During the Import Process, each new DB Alias is substituted for the current DB Alias with which it is paired.		

Example

Use the following command line syntax to import objects in the X:\FORIMPORT.TXT file from the command line, without overwriting any object definitions already in the Optim Directory. The process report is written to a file named X:\RESULTS.TXT and processing will continue in the event of an error. The example will also change the DB Alias xxx to yyy, and xxx2 to yyy2 for the imported objects.

PR0CMND /IMPORT INPUT=X:\FORIMPORT.TXT
OUTPUT=X:\RESULTS.TXT **OVERWRITE-**
CONTINUEONERROR+ DBALIAS=(xxx, yyy, xxx2, yyy2)

Output/Input File Format

The Export Utility generates a set of SQL-like statements for selected object definitions in the current Optim Directory and stores the statements in the Output File. This Output File is the input for the Import Utility.

The output from the Export Utility is a text file that can be viewed using any standard text editor. Although it is not recommended, you can edit this file. Use caution when editing because Import requires a specific format for the statements in the file and deviations in the format may cause errors. The definition statements for exported objects are documented in the following pages.

MVS Definitions

You can import object definitions exported from an Optim Directory on MVS. However, the definition statements differ, depending on the Optim z/OS Solution component (e.g., ACCESS) that created the definition and the options in effect when the definition was created. These differences may include parameters that are not supported when importing definitions to Optim. You may need to edit the Optim z/OS Solution Output File before importing it to Optim.

Unsupported keywords and values from the Optim z/OS Solution definitions can result in errors when importing to Optim. These keywords and values must be removed from a file or edited before the

definition can be imported. The keywords from Optim z/OS Solution definitions that are unsupported in Optim are listed after the discussion for each object type. Unsupported values for supported keywords are noted in the discussion for each keyword.

Additionally, definitions originating on MVS may contain parameters that are not relevant for Windows. These parameters have no effect when the definition is imported and used.

You must edit the names of the files (e.g., Extract Files, Control Files, Row List files, etc.) in the MVS definitions, to conform to Windows standards.

Format Rules

The following general format rules apply to the files:

- Each statement begins at the first position of a line with the keyword CREATE.
- Each statement is terminated with a semicolon. Export inserts a blank line after the semicolon, but it is not required.
- At least one space is inserted between parameters.
- The qualifiers for individual objects in sets of specifications, such as the Table List entries for an Access Definition, are enclosed in parentheses.
- If the length of the statement exceeds the length of the line, a break occurs before or after a parameter and its qualifiers. The remaining text continues on the next line. Export indents the continued text, but this is not required.

  Long character strings, such as description text entries, Row List definitions, SQL WHERE clauses, or selection criteria text, are delimited by two forward slashes (//). Continuation characters are not used.

  No spaces or indentations are added. (If you insert additional spaces or continuation characters, they are assumed to be part of the string.) The text continues for the complete length of the line, wrapping to the next line until the forward slashes are reached, indicating the end of the text.

Definition Statement Syntax

The definition statement syntax for each object type is documented in the following section. The syntax conventions for these statements are:

KEYWORD
: Statement keywords are shown in uppercase and must be supplied as shown.

text
: Variable text is shown in lowercase italics.

( )
: Required as a statement delimiter to group a series of qualifiers for a parameter.

[ ]
: Indicates an optional parameter.

{ }
: Indicates a choice of two or more settings from which one (and only one) must be selected.

< >
: Indicates a choice of two or more settings from which none or any may be selected.

|
: Separates options.

Note: The definition for a Local object is exported with the object for which they are defined. The definition for a Subordinate Named Object is exported only when subordinate objects are specifically included. Otherwise, named objects are referenced only.

The syntax for **Local** and **Named** objects is the same, except that Local objects are preceded by the keyword LOCAL (for example, LOCALAD), and the associated parameters are delimited by parentheses or double slashes.

Access Definitions

The following syntax is used to generate statements for exported Access Definitions:

```
CREATE AD identifier.name
   [DESC //'description'//]
   SRCQUAL srcqual  START starttable
   ADDTBLS {Y|N} MODCRIT {Y|N} ADCHGS {Y|N}
   USENEW {Y|N}
   PNSSTATE { N | L [LOCALRL //'pkval1','pkval2',...'pkvaln'// ] |
       F ROWLIST //'filname.pns'// }
   [GRPCOL colname GRPROWS n GRPVALS n]
   [VAR (VarName PRMPT //'promptstring'// [DFTL dfltval])]
   TABLE (tablename  REF {Y|N} [EXTRFREQ n] [EXTRLIMIT n]
      PREDOP{A|O} [VARDELIM {~ | ! | @ | $ | : | % | + | ? }]
      COLFLAG {Y|N} DAA {Y|N} UR {Y|N}
         [CORRELNAME corelname]
         [ARCIDX {indexname columnname1 columnnamen}
      [COLUMN (colname  DISP {Y|N} HEADING {N|L}{L|C|R}
         NATIVELOB {Y|N} EXTRACT {Y|N}
         [ASSOCIATION {colname|.ext}][PRED //selcrit//])]
      [SORT ({A|D}colname1[, {A|D}colname2][, {A|D}col...])]
      [SQL //whereclause//])
      [ARCHACTS {ACTION {SEP | BEFRT | BER | AELRT |
            EEP | SDP | BDFRT | BDR | ADR | ADLRT | EDP |
            SRP | BRFRT | BRR | ARR | ARLRT | ERP}
          SQL //SQL statement//
         [HOSTVAR {~ | ! | @ | $ | : | % | + | ? }]  |
         SAMEAS  actionname}[DBALIAS dbalias]
         [ON_ERROR {STOP | SKIP | PROCESS}])
      [FILEATTACH (PREFIX prefix [TRIGGER colname]
         STOP {Y|N} DELFILE {Y|N} INCL_DEFPATH {Y|N}
         NAME_PARTS(name1, name2,...namen)
         SEARCH_PATHS(//'path1'// //'path2'//... //'pathn'//) )]
   [REL (relname STATUS {K | NEWK | UNK | NEWUNK}
      USAGE {D|I|S}
      Q1 {Y|N} Q2 {Y|N} LIMIT n
      TYPE {dbms | PST | PST_GENERIC | UNK}
      PAR ptable CHI ctable
      [PAR_ACCESS {D | K | S}]
      [CHI_ACCESS {D | K | S}]
      [PAR_ KEYLIMIT n]
      [CHI_ KEYLIMIT n])]
   [DEFPATHS (//'path1'// //'path2'//... //'pathn'//)]);
```

The keywords correspond to values that can be specified for an Access Definition. For more information about an individual parameter, see “Using the Editor” on page 47.

Note: Some keywords are automatically populated with values during Export. If these keywords are marked as optional, however, they are not required for Import. If you edit an MVS definition to import it, you can omit these keywords.

Keywords

The keywords correspond to values that can be specified for an Access Definition.

AD The name of the Access Definition.

identifier.name
Two-part name of an Access Definition.

DESC Specify the description for the Access Definition

//description//
The description for the Access Definition, delimited by double slashes.

Tables

SRCQUAL
The default qualifier.

dbalias. [*creatorid*]
The one or two-part qualifier (DB Alias and Creator ID) for the tables in the Access Definition.

START
The Start Table for the Access Definition.

[[*dbalias.*] *cid.*] *tablename*
One, two, or three-part table name.

Only specify the DB Alias and Creator ID if they differ from the Default Qualifier.

Parameters

ADDTBLS
Indicator for ability to add tables to the Access Definition during a browse or edit session is selected. This capability is available for a future enhancement.

Y Tables can be added to the Access Definition.

N Tables cannot be added to the Access Definition.

MODCRIT
Indicator for the ability to add or modify criteria in the Access Definition during a browse or edit session. This capability is available for a future enhancement.

Y Selection Criteria for the Access Definition can be specified or modified.

N Selection Criteria for the Access Definition cannot be specified or modified.

ADCHGS
Indicator for the ability to modify and save the Access Definition during a browse or edit session. This capability is available for a future enhancement.

Y Access Definitions can be saved during a browse or edit session.

N Access Definitions cannot be saved.

Relationships

USENEW
Indicator for automatic inclusion of new relationships during processing.

Y New relationships are included in an Extract Process.

N New relationships are not included in an Extract Process.

Point and Shoot

Point and Shoot parameters for the Access Definition include:

PNSSTATE	Indicator for a Row List:		
	N	No Row List is included.	
	L	A Local Row List is included	
		LOCALRL	Indicator for a Local Row List.
		pkvaln	One or more primary key values, delimited by two forward slashes (//) with each value enclosed in single parentheses.
	F	Indicator for an external, named, Row List.	

		ROWLIST	Indicator for the name of the Row List file.
	The name of an external Point-and-Shoot file for the Access Definition.		
		//filename//	The name of the external Point-and-Shoot file for the Access Definition, enclosed in single parentheses. You must provide the full path, if the file does not reside in the default directory.

Group

GRPCOL
The name of a column for group selection processing.

colname
Name of the column in the Start Table for the set of data you want to extract.

GRPROWS
The number of rows to be selected for group selection processing.

n Maximum number of rows to extract for each unique group based on the specified column in the Start Table.

GRPVALS
The number of unique column values for group selection processing.

n Number of unique groups to extract based on a selected column in the Start Table.

Variables

Each variable definition has the following parameters:

VAR The name of the variable.

varname
The 1 to 12 character name for the variable in the Access Definition.

PRMPT
The prompt string for the variable.

promptstring
Text that prompts for the variable value at run time. Type the prompt string exactly as you want it to appear in the process request dialog (up to 50 characters).

DFTL The default value for the variable.

dfltval Default value for the variable to be used when no value is specified for the variable at run time.

Tables Keywords

A **TABLE** entry occurs for each table on the Table List for the Access Definition. The set of keywords for each table is enclosed in parentheses following the keyword "TABLE." The keywords correspond to values specified on the **Tables** tab of the Access Definition Editor and the tabs of the Table Specifications dialog

TABLE
The name of the table.

tablename
The name of the table in the Access Definition. The fully-qualified name is given only if the DB Alias and Creator ID differ from the Default Qualifier.

REF Reference table indicator. If a table is a reference table, all rows are extracted regardless of relationships.

Y The table is a reference table.

N The table is not a reference table.

EXTRFREQ
Selection factor for sampling rows.

n A value from 1 through 9999.

EXTRLIMIT
Maximum number of rows to extract from a table.

n A value from 1 through 99999999.

PREDOP
Indicator for combining selection criteria for the table.

A Match selection criteria for all columns.

O Match selection criteria for any column.

VARDELIM
The variable delimiter as one of the following characters.

~ ! @ $: % + ?
Character required to identify a column value or built-in variable in SQL statements for Selection Criteria. The VARDELIM value must differ from the HOSTVAR value.

COLFLAG
Indicator for modifications to column specifications.

Y Column specifications were modified.

N Column specifications were not modified.

DAA Indicator for deletion of rows from the table in the database after they are extracted to an Archive File.

Y Delete rows from the table.

N Do not delete rows from the table.

UR Indicator for extracting uncommitted rows from the table.

Y Extract uncommitted rows.

N Do not extract uncommitted rows.

CORRELNAME
The correlation name for the table, if any.

corelname
The abbreviation for the fully qualified table name. You can use this short name in the SQL WHERE Clause.

ARCIDX
The Archive Index name, if any, and the corresponding column(s) to index. Up to 16 indexes can be designated per table.

indexname
The name of the Archive Index.

columnnamex
Name of columns used to create the Archive Index.

Column Keywords

COLUMN keywords correspond to values for each column on the **Columns** and **Selection Criteria** tabs of the Table Specifications dialog, accessed using the right-click options on the **Tables** tab of the Access Definition Editor. This set of keywords is enclosed in parentheses following the keyword "COLUMN."

COLUMN
The name of the column.

colname
The name of the column. Column names appear in order of precedence. For example, the first column is listed first, followed by the second column, etc.

DISP Indicator for displaying the column during a Point-and-Shoot session.

Y The column is displayed.

N The column is not displayed.

HEADING
Heading options for positioning the column headings in a Point-and-Shoot session.

N Display column names as the heading during a Point-and-Shoot or browse session.

L Display column labels as the heading during a or browse Point-and-Shoot session.

C Position column heading as center.

L Position column heading as left.

R Position column heading as right.

NATIVELOB
Indicator for LOB column display mode in the Table Editor.

Y Display normal LOB data.

N Display LOB data as VARCHAR or VARBIN data.

EXTRACT
Indicator to extract CLOB or BLOB data in an Archive or Extract Process.

Y Extract data.

N Do not extract data.

ASSOCIATION
Association to identify the type of file in a LOB column

colname
Name of a column to reference for association with the LOB-type column. The first three characters in the column are used as the file name extension.

.ext A file name extension to associate with the LOB-type column.

PRED Indicator that selection criteria apply for the column.

//selcrit//
The selection criteria for the specified column.

Sort Keyword

If a column has a sort order, a **SORT** keyword is added. Add a sort order using the **Sort** tab on the Table Specifications dialog.

SORT Options for order of data in column:

A Display data in the column in ascending order.

D Display data in the column in descending order.

colname
Name of the column.

SQL Keyword

This keyword is included when an SQL WHERE clause applies to the table, as on the **SQL** tab of the Table Specifications dialog.

SQL Indicator for an SQL WHERE clause for the column.

//whereclause// The SQL WHERE clause for the specified column.

Archive Actions Keyword

The ARCHACTS keyword is used for any Archive Actions included in the Access Definition.

ARCHACTS
Indicator that Archive Actions are included with the table. The Archive Actions for each table are contained in the parentheses that follow this keyword.

ACTION
Identifier for the Archive Action phase. If an ACTION is specified, SQL parameters are required.

SEP Start of Extract Process.

BEFRT
Before Extract of First Row from Table.

BER Before Extract of Row.

AELRT
After Extract of Last Row from Table.

EEP End of Extract Process.

SDP Start of Delete Process.

BDFRT
Before Delete of First Row from Table.

BDR Before Delete of Row.

ADR After Delete of Row.

ADLRT
After Delete or Last Row from Table.

EDP End of Delete Process.

SRP Start of Restore Process.

BRFRT
Before Restore of First Row to Table.

BRR Before Restore of Row.

ARR After Restore of Row.

ARLRT
After Restore of Last Row to Table

ERP End of Restore Process.

SQL The SQL WHERE clause associated with the Archive Action.

//SQL statement//
An Insert, Update, Delete, or Stored Procedure Call SQL statement.

HOSTVAR
The variable delimiter as one of the following characters.

~ ! @ $: % + ?
Character required to identify a column value or built-in variable in SQL statements for Archive Actions. The HOSTVAR value must differ from the VARDELIM value.

SAMEAS
Option to use an SQL statement for an Action Phase of the same class. If **SAMEAS** is specified, SQL parameters are not required.

actionname
The name of the Action Phase.

DBALIAS
The DB Alias needed to reference a table in a database other than that referenced by the Default Qualifier.

dbalias The DB Alias name.

ON_ERROR
Processing if an error occurs when an SQL statement is executed.

PROCESS
Ignore the SQL statement for the row and continue processing.

SKIP Do not process the row, but continue processing other rows.

SKIP is valid for the Action Phases: Before Extract of Row, Before Delete of Row, and Before Restore of Row. The process report notes the number of rows skipped.

STOP Stop the process when an error occurs as a result of the SQL statement. (Default)

File Attachments Keyword

If a table will have file attachments extracted, the FILEATTACH keyword is added.

FILEATTACH
File attachments are included with the table. Parameters for each file extracted with the table follow this keyword, in parentheses.

PREFIX
The prefix for pseudocolumn names in the Archive or Extract File.

prefix The prefix, unique to the table, used to name *prefix*_FILE_NAME, *prefix*_BLOB, and *prefix*_ATTRIB.

TRIGGER
The column that controls processing of a file attachment.

colname
The column name.

STOP Indicator for processing if a file is not found.

Y Stop processing.

N Do not stop processing.

DELFILE
Indicator for processing the attached file during Delete Processing.

Y Delete the file.

An attached file is deleted only if the associated row is successfully deleted.

N Do not delete the file.

INCL_DEFPATH
Indicator to search the default paths for the file attachment.

Y Search the paths in **DEFPATHS**.

N Search paths in **SEARCH_PATHS**.

NAME_PARTS
The components in attached file names.

namen The components that make up the name of the attached file. You can provide a combination of column names and literals, separated by commas and with literals enclosed in quotes. Values in the named column(s) are concatenated with any literals to generate the file name, which is combined with the search path to locate the appropriate file.

SEARCH_PATHS
One or more file paths to search, in the order listed. Once the file is found, any remaining paths are not searched.

//pathn//
The file path(s) searched for the file.

Relationships Keyword

A **REL** entry is provided for each relationship between tables referenced in the Access Definition. This set of keywords is enclosed in parentheses following the keyword "REL." Keywords correspond to values specified for the use of a relationship by an Access Definition.

REL The relationship name.

relname
The name of the relationship.

STATUS
The status of the relationship.

K A known relationship.

NEWK
A known relationship that is new.

UNK An unknown relationship.

NEWUNK
An unknown relationship that is new.

USAGE
The **Select** status for the relationship.

D Dormant.

I Initial

S Selected

Q1 Option 1:

Y Extract a parent row for every child row to satisfy referential integrity rules.

N Do not extract a parent row for every child row.

Q2 Option 2:

Y Extract additional child rows for each parent row extracted to satisfy Option 1.

N Do not extract additional child rows.

LIMIT
The maximum number of rows to extract from a child table in any relationship.

n The maximum number of rows.

TYPE The type of the relationship.

type Indicates whether the relationship is generic, explicit (Optim), or defined to a specific DBMS (e.g., Oracle).

PAR Name of the parent table in the relationship.

ptable The parent table name.

CHI Name of the child table in the relationship.

ctable The child table name.

PAR_ACCESS
The default method of accessing the parent table in the relationship:

D Default.

K Force Key Lookup.

S Force Scan.

CHI_ACCESS
The default method of accessing the child table in the relationship:

D Default.

K Force Key Lookup.

S Force Scan.

PAR_ KEYLIMIT
The maximum number of key lookups performed at one time for the parent table.

n Specify the maximum number (1 to 100) of key lookups.

CHI_ KEYLIMIT
The maximum number of key lookups performed at one time for the child table.

n Specify the maximum number (1 to 100) of key lookups.

Default Paths Keyword

Default path for attached files.

DEFPATHS
One or more default paths to search for file attachments. Paths are searched in the order listed.

Once the file is found, any remaining paths are not searched.

//pathn//
The file path name used to search for the file.

Archive File Collections

The following syntax is used to generate a statement for each exported Archive File Collection:

```
CREATE AFC identifier.name
   DEFDATE //'yyyy-mm-dd'//
   FILES (archivefile1, archivefile2, ...);
```

Keywords

The keywords correspond to values that can be specified for an Archive File Collection. For more information about an individual parameter, see the *Archive User Manual*.

AFC *identifier.name*	The name of the Archive File Collection, specified in two parts (*identifier.name*), is required following the **CREATE AFC** keyword.
DEFDATE *//'yyyy-mm-dd'//*	The default date in yyyy-mm-dd format.
FILES *(archivefile1, archivefile2, ...)*	The **FILES** keyword is followed by a list of fully qualified Archive Files in the Archive File Collection. For example, (C:\Optim\customersFEB.af, C:\Optim\ customersMAR.af)

Calendars

The following syntax is used to generate a statement for each exported Calendar:

```
CREATE CALENDAR identifier.name
   [DESC //description//] [MATCH {PREVIOUS|NEXT}]
   WEEKENDN dayofweek  DFLTSEP(separator)  DFLTYEAR year
   MONTHS(shortmon1, longmon1,...shortmon12, longmon12)
   DATE(date) RULE(rule);
```

Keywords

The keywords correspond to values that can be specified for a Calendar. For more information about an individual parameter, refer to "Using the Editor" on page 261.

CALENDAR
Specify the name of the Calendar.

name Name of the Calendar.

DESC Specify the description of the Calendar.

//description//
Text to describe the content or purpose of the Calendar (up to 40 characters).

MATCH
Specify the direction for adjusting calculated dates. This adjustment applies when you specify Closest in a Calendar Rule, and the days on either side of a calculated date are equal.

NEXT Adjust the calculated date to the next day if needed.

PREVIOUS
Adjust the calculated date to the previous day if needed.

WEEKEND
Specify the day of week that represents a weekend in the calendar year.

N Number (1 or 2) used to indicate the first or second weekend day.

dayofweek
The day of the week (e.g., Monday, Friday) that represents all or part of a standard weekend.

DFLTSEP
Specify the date separator to use in formatting dates.

(*separator***)**
Date separator for formatting dates.

DFLTYEAR
Specify the default year.

yyyy The default year in yyyy format.

MONTHS
Specify the names of the months defined in the Calendar.

shortmon
Abbreviated name for month.

longmon
Full name for a month.

DATE Specify the unique names to describe the special days in a calendar year. Up to 14 names can be specified and each name may be specified as many times as required.

There must be a separate DATE entry for each instance defined to the calendar.

date The name of the special day in the calendar year.

RULE Enter the specifications for handling date aging to accommodate your unique requirements. Any number of rules may be specified.

There must be a separate RULE entry for each instance defined to the calendar.

rule The Specifications for handing date aging to accommodate your requirements.

DATE

The following information is required for each DATE in the Calendar.

```
DATE (datename
   [DESC //description//] DAY day WHICH {n|EVERY|LAST}
   OFFSET n  DURATION n  ABSORB {Y|N}
   RESOLVE {CLOSEST|PREVIOUS|NEXT|None}
   MONTH longmon YEAR nnnn
   REOCCUR_ TYPE {DAY|WEEK|MONTH|YEAR|None}
   REOCCUR_ VALUE n REOCCUR_ ENDDATE mm/dd/yyyy)
```

DATE Specify the name of the category to which the date applies.

datename
1 to 8 character name to describe the Calendar Date.

DESC Specify text to describe or explain the Calendar Date delimited by double slashes.

//desc// Text to describe the Calendar Date (up to 40 characters).

DAY Enter the name of the day that identifies the Calendar Date.

day The name of the day (e.g., Monday).

WHICH
Specify the instance that applies to the Calendar Date.

n Enter the number associated with the DAY keyword.

EVERY
Indicates every instance of a specific day.

LAST Indicates the last instance of a specific day.

OFFSET
Specify the number of days to adjust the Calendar Date.

+ Increment the Calendar Date by the number supplied.

– Decrement the Calendar Date by the number supplied.

n Number from 1 to 366 to adjust the Calendar Date.

DURATION
Specify the duration of the Calendar Date.

n Number from 1 to 366 to specify the duration of the Calendar date.

The duration value entered must not cause any portion of the Calendar date to occur in a different year.

ABSORB
Indicate whether to mark an additional day as a holiday if the date falls on a Tuesday or a Thursday.

Y Mark an additional day as a holiday.

N Do not mark an additional day as a holiday.

RESOLVE
Indicate how to resolve a Calendar Date if it occurs on a weekend or holiday.

None Do not resolve the Calendar Date.

CLOSEST
The closest workday.

NEXT The next workday.

PREVIOUS
The previous workday

MONTH
Specify the month to start using the Reoccurs (frequency) settings for a Calendar Date.

longmon
The month to start using the Reoccurs (e.g., JAN).

YEAR Specify the year to start using the Reoccurs (frequency) settings for the Calendar Date.

yyyy The year in yyyy format.

REOCCUR_TYPE
Specify the frequency of the Calendar Date. (For example, in some organizations, payday is every two weeks regardless of the date.)

None The date does not reoccur.

DAY The date reoccurs every **nth** day.

WEEK The date reoccurs every **nth** week.

MONTH
The date reoccurs every **nth** month.

YEAR The date reoccurs every **nth** year.

REOCCUR_VALUE
Specify the **nth** unit of time specified in the **REOCCUR_TYPE** keyword.

You can specify a value only when REOCCUR_TYPE keyword is not "(None)".

n Number for the **nth** unit of time.

REOCCUR_ENDDATE
Specify the date the frequency specifications are to be discontinued.

You can specify a value only when REOCCUR_TYPE keyword is not "(None)".

mm/dd/yyyy
Date to discontinue using the frequency specifications.

RULE

The following information is required for each RULE in the Calendar.

```
RULE rulename
   [DESC //description//] AVOIDDATES(date1, date2, ... daten)
   SEARCHDATES(date1, date2, ... daten)
   ADJUSTMENT{CLOSEST|PREVIOUS|NEXT};
```

RULE Specify the name of the Calendar Rule you want to define.

rulename
The Calendar Rule name.

DESC Enter the text to describe the type of date you are defining.

//description//
1 to 40 character description of the type of date you are defining.

AVOIDDATES
Specify the Calendar Date to avoid when making an adjustment.

daten The date type you defined by the DATE keyword or one of the sample date types provided with the Calendar Utility.

SEARCHDATES
Specify the Calendar Dates to search for when making an adjustment.

daten The date type you defined by the DATE keyword or one of the sample date types provided with the Calendar Utility.

ADJUSTMENT
Direction to adjust an aged date if it does not satisfy the Avoid or Search Specifications.

CLOSEST
The closest day.

NEXT The next day.

PREVIOUS
The previous day

Column Maps

This syntax is used to generate a statement for each exported Column Map.

```
CREATE CM identifier.name
   [DESC //description//] [EXTDSN extractfilename]
   SRC srctable DEST destable  VALRULES {M | C}
   (src-expr = dest-col
   [,src-expr = dest-col]);
```

Keywords

The keywords correspond to values that can be specified for a Column Map. For information about an individual parameter, see "Using the Editor" on page 127.

CM Specify the name of the Column Map.

identifier.name
The two-part name of the Column Map.

DESC Describe the content or purpose of the Column Map.

//description//
The 1 to 40 character description of the Column Map, delimited by double slashes.

EXTDSN
Enter the name of the Extract or Archive File from which column definitions are taken.

extractfilename
The name of the Extract or Archive File containing the Source table defined in the Column Map.

SRC Specify the fully-qualified name of the source table, containing the columns defined in the Column Map.

dbalias.cid.tablename
Fully qualified name of the source table.

DEST Specify the fully-qualified name of the destination table.

dbalias.cid.tablename
Fully qualified name of the destination table.

VALRULES
Indicate the validation rules for the Column Map.

M Move/ARchive for Column Maps for Convert, Extract, Insert, or Load Requests.

C Compare for Column Maps.

The following source column to destination column mapping information is required for each column in the Column Map. At least one pair of columns must be specified.

src-expr
Text of the expression specified for the source column (**Source Column**).

dest-col
The name of the destination column (**Destination Column**).

One of the following is included when a Column Map Procedure is specified for a pair of columns.

CMPROC
The name of the Column Map Procedure for the pair of columns.

LOCAL CMPROC
The local Column Map Procedure definition enclosed in parentheses.

Column Map Procedures

The following syntax is used to generate a statement for each exported Column Map Procedure.

```
CREATE CMPROC identifier.name
   [DESC //description//]
   TEXT //columnmapproceduretext//
```

Keywords

The text keyword in a Column Map Procedure is followed by SQL statements and Optim Basic specified for the specific procedure. For complete information, see Chapter 6, “Column Map Procedures,” on page 175.

CMPROC
Specify the name of the Column Map Procedure

identifier.name
The two-part name of the Column Map Procedure name.

DESC Describe the content or purpose of the Column Map Procedure.

//description//
The 1 to 40 character description of the Column Map, delimited by double slashes.

TEXT Allow you to write the desired Column Map Procedure.

//columnmap proceduretext//
The text that comprises the procedure.

Currency

The following syntax is used to generate a statement for each exported Currency Table.

```
CREATE CURRENCY identifier.name
   [DESC //description//]
   [DECPL dec]
   [PIVOT year]
   [SHOW {EURO | ISO | USER}]
   [ISOEDIT {Y|N}]
   [RATES ( fromdate todate
   [ (fromcurr 'fromdesc' tocurr 'todesc' rate) ]
   [ (fromcurr 'fromdesc' tocurr 'todesc' rate) ] )]
   [TYPEn ( [ (code currency) ]
   [ (code currency) ] )] ;
```

Keywords

The keywords correspond to values that can be specified for a Currency Table. For more information, see "Using the Currency Editor" on page 276.

CURRENCY
Specify the name of the Currency Definition.

identifier.name
The two-part name of the Currency Definition name.

DESC Describe the content or purpose of the Currency Definition.

//description//
The 1 to 40 character description of the Currency Definition, delimited by double slashes.

General

DECPL
Enter the number of decimal places for the value when triangulation is performed.

dec Specify a value between 3 and 6.

PIVOT
Enter the value to determine the pivot year.

year The value (00 to 99) that determines the appropriate century for two-digit years.

SHOW
Option to display currency codes on the Currency Editor selection lists available from the Rates tab.

EURO Show a list of all European Community currency codes and user-defined codes on the Rates tab.

ISO Show a list of all ISO and user-defined currency codes on the Rates tab.

USER Show a list of all user-defined currency codes on the Rates tab.

ISOEDIT
Specify whether to permit the editing the ISO currency descriptions on the Rates tab.

Y Allow editing of ISO description.

N Disallow editing of ISO description.

Rate

RATES
Option to specify parameters for the currency rate.

Y Allow editing of ISO description.

RATE Follow this keyword with parameters for the currency rate.

fromdate
From date MM/DD/YYYY

todate To date MM/DD/YYYY

Note: Calendar is the same as *fromdate*.

fromcurr
Source currency code (3 characters)

'*fromdesc*'
Source currency code description, enclosed in single or double quotation marks.

tocurr Target currency code (3 characters)

'*todesc*' Target currency code description, enclosed in single or double quotation marks.

rate Conversion rate (*n.n*) for the "from" and "to" currency codes.

Type

TYPE*n*
Specify up to 4 lookup tables for currency codes. Type Table: Table 1, Table 2, Table 3, or Table 4

code Code to represent a currency type as it appears in the Type Table.

currency
Currency code (3 characters)

DB Aliases

The following syntax is used to generate a statement for each exported DB Alias:

```
CREATE DBALIAS dbaliasname
   [DESC //description//]
   DIRDB {Y|N} TYPE type type VER version
   [IDCASE {U|L|M}] CODEPAGE code page}
   [DELIM delimitercharacter] [DECSEP decimalseparator]
   CONNECT servername DBQUAL dbqualifier
   [SPQUAL spqualifier] [NETSERVER netservername]
   [TABLESPACE tablespacename]
   USEDFLTTBLSP {Y|N}
   DATABASE database
   USEDFLTDB {Y|N} [TABLEALLOC percentage]
   IDXTYPE idxtype [IDXCID creatorid]
   IDXTBLSPCTYPE idxtblspctype [IDXTBLSPC tablespacename]
   [INDEXALLOC percentage]
```

Keywords

The keywords correspond to values that can be specified for a DB Alias. For more information about an individual parameter, see "Using the Editor" on page 190.

DBALIAS *dbaliasname*
The name of the DB Alias, specified as a single string (*dbaliasname*), is required following the **CREATE DBALIAS** keyword.

DESC //*description***//**
Description of the DB Alias, delimited by double slashes.

Long character strings such as description text entries are delimited by two forward slashes (//). Continuation characters are not used, and no spaces or indentations are added. If additional spaces or continuation characters are inserted, the string is imported incorrectly. The text continues for the complete length of the line width, wrapping to the next line until the forward slashes are reached, indicating the end of the text.

General

DIRDB {Y|N}
(**Optim Directory Database**) Indicates whether the DB Alias is the one used for the Optim Directory connection. This information is necessary to avoid conflicts during certain types of processing.

Y The DB Alias is the one used for the Optim Directory connection.

N The DB Alias is not the one used for the Optim Directory connection.

TYPE *type*
The database type:

Type	Database
2	DB2MVS
C	DB2CS
O	Oracle
S	Sybase ASE
I	Informix
G	SQL Server
M	ODBC

VER *version*
The version number of the DBMS (**DBMS Version**).

IDCASE {U|L|M}
Indicates the identifier case. This keyword is only valid for DBMS types that allow user control of object identifier case—i.e., Sybase ASE.

IDCASE value	Case
U	upper
L	lower
M	mixed

CODEPAGE *code page*
Indicates the code page selected for the DBMS interface to use for character data when communicating with the client. It does not indicate the code page of the data on the DBMS server. Not used for DB2 and Sybase ASE.

DELIM *delimitercharacter*
Indicates the delimiter needed to enclose strings with special characters or blanks.

DECSEP *decimalseparator*
Indicates the character needed to represent a decimal point in numeric strings.

Connection

CONNECT *servername*
Indicates the connection string required to access the DBMS Server.

DBQUAL *dbqualifier*
Indicates the string to use as a name qualifier that identifies the database that the DB Alias represents (Sybase ASE only).

Server

SPQUAL *spqualifier*
Indicates the name qualifier for the location of the Stored Procedure that Optim must execute to access the database.

NETSERVER *netservername*
Indicates the name of the Network Server where the database (identified in the Connection String) resides (Sybase ASE only).

Table Defaults

TABLESPACE *tablespacename*
The name of the default tablespace for objects created using the Create Utility.

USEDFLTTBLSP {Y|N}
Indicates whether default tablespace is used.

DATABASE *database*
The name of the default database if importing from MVS DB2.

USEDFLTDB {Y|N}
Indicates whether the default database is used.

TABLEALLOC *percentage*
Percentage of the source value storage (0-999) to be used for the target.

Index Defaults

IDXTYPE *idxtype*
Indicates what value to use for the qualifier, Index ID, when new indexes are created using the Create Utility. Specified as one of the following:

TABLE
SOURCE
CURRENT
EXPLICIT

IDXCID *creatorid*
Explicit value of the Owner ID. This is valid only if IDXTYPE is EXPLICIT.

IDXTBLSPCTYPE *idxtblspctype*
Indicates the value to use for the qualifier, Tablespace, when new indexes are created using the Create Utility. Specified as one of the following:

TABLE
SOURCE
EXPLICIT

IDXTBLSPC *tablespacename*
Explicit value of the Tablespace. This is valid only if IDXTBLSPCTYPE is EXPLICIT.

INDEXALLOC *percentage*
Percentage of the source value storage (0-999) to be used for the target.

Edit Definition

The following syntax is used to generate a statement for each exported Edit Definition.

```
CREATE ED identifier.name
   [DESC //description//]
   DEFQUAL defaultqualifier
   START starttable
   {AD adname | LOCALAD (addef)}
   MODE {E|B|O} SIDEVIEW {Y|N} DISPLAY_ATTRS {Y|N}
   DISPLAY_DELETED {Y|N} WARN_DELETE {Y|N}
   DISPLAY_ROWS nn  UNDOLEVELS nn,
   VAR_PROMPT {Y|N} {VAR (name, value)...}
   RETAIN_SELCRIT {Y|N}
   TABLE (tablename MODE {E|B|O} SIDEVIEW {Y|N}
      DISPLAY_ATTRS {Y|N} DISPLAY_DELETED {Y|N}
      WARN_DELETE {Y|N} DISPLAY_ROWS nn
      [COL_ORDER //...//] [COL_WIDTHS //...//]
      [COL_VISIBLE //...//] [COL_LOCKED //...//]
      [SIDE_COL_ORDER //...//] [SIDE_COL_WIDTHS //...//]
      [SIDE_COL_VISIBLE //...//] [SIDE_COL_LOCKED //...//]
      [OWNER [[dba.] cid.] tablename TBLID n
      CONSTRAINT cnstrname       PARENT {Y|N}])
   TABLE (...);
```

Keywords

The keywords correspond to values that can be specified for an Edit Definition.

ED *identifier.name*
The name of the Edit Definition, specified in two parts (*identifier.name*), is required following the **CREATE ED** keyword.

DESC *//description//*
A description of the Edit Definition, delimited by double slashes.

DEFQUAL *defaultqualifier*
The name of the default qualifier. The default qualifier applies to table names that are not fully qualified.

START *starttable*
The Start Table for the Edit Definition. The DB Alias and Creator ID for the table name are included only if they differ from the Default Qualifier.

AD *adname*
The full two-part name of the Named Access Definition being used.

LOCALAD (*addef***)**
A full definition of the Local Access Definition being used, contained in parentheses, follows this keyword. The syntax and parameters for the definition of a Local Access Definition are identical to those for a standalone Access Definition.

MODE {E | B | O}
The default Edit Definition mode to apply to newly added tables.

E Edit

B Browse

O Browse Only

SIDEVIEW {Y | N}
The default Edit Definition table display format.

Y View in side label format.

N View in columnar format (default).

DISPLAY_ATTRS {Y|N}
The default for displaying column attributes in columnar mode.

Y Display attributes.

N Do not display attributes (default).

DISPLAY_DELETED {Y|N}
The default for displaying deleted rows.

Y Display deleted rows (default).

N Do not display deleted rows.

WARN_DELETE {Y|N}
The default for notifying the user before a cascade delete is about to occur for one or more tables other than the table for which the delete operation was requested.

Y Display warning (default).

N Do not display warning.

DISPLAY_ROWS *nn*
The default for the number of rows to be displayed for tables that have been joined to subordinate tables. The default number is 1; the valid range is 1-999.

UNDOLEVELS *nn*
The default for the number of undo levels per table row. The default number is 5; the valid range is 1-20.

VAR_PROMPT {Y|N}
The default for prompting for variable values when the Edit Definition is based on an Access Definition that contains variables.

Y Prompt for variables.

N Do not prompt for variables (default).

VAR (*name,value***)**
The name and the value for each variable.

RETAIN_SELCRIT {Y|N}
Indicator for applying selection criteria each time the table is self-referenced while browsing or editing data.

Tables Keywords

There is a **TABLE** entry for every table on the Table List for the Edit Definition. The set of keywords for each table is enclosed in parentheses following the keyword "TABLE." The following keywords have the same meaning as above, but apply to the specific table: MODE, SIDEVIEW, DISPLAY_ATTRS, DISPLAY_DELETED, WARN_DELETE, DISPLAY_ROWS

Note: In addition to the keywords mentioned above, the table also contains grid-related keywords. Grid-related keywords define column ordering, width, visibility, and locking, as specified by the user prior to saving the Edit Definition. Grid-related keywords should not be modified.

COL_ORDER // ###...//
Column order for tables in columnar grid format.

COL_WIDTHS // ###...//
Column widths for tables in columnar grid format.

COL_VISIBLE // ###...//
Column visibility for tables in columnar grid format.

COL_LOCKED // ###...//
Column locking for tables in columnar grid format.

SIDE_COL_ORDER // ###...//
Column order for tables in side label grid format.

SIDE_COL_WIDTHS // ###...//
Column widths for tables in side label grid format.

SIDE_COL_VISIBLE // ###...//
Column visibility for tables in side label grid format.

SIDE_COL_LOCKED // ###...//
Column locking for tables in side label grid format.

The following keywords pertain to joined tables:

OWNER *[[dba.cid.]] tablename*
The name of the table to which the current table is joined. If the current table is the Start Table, this keyword will not exist. The joined table name must be fully qualified if the table is from a different database or Creator ID.

TBLID *n*
For all tables except the Start Table, number to indicate the parent table instance used for joining.

CONSTRAINT *cnstrname*
The constraint name of the relationship used to join the current table to the owner table.

PARENT {Y|N}
Indicates if the current table is the PARENT or CHILD in the relationship with the owner table.

Y Current table is the PARENT.

N Owner table is the PARENT.

Primary Keys

The following syntax is used to generate a statement for each exported primary key.

```
CREATE PK table
   [DESC //description//]
   [BASECREATORID basecreatorid]
   COLS (column [, column]);
```

Keywords

The keywords correspond to values that can be specified for a primary key. For more information about an individual parameter, see Chapter 8, “Primary Keys,” on page 203.

PK *table*
The name of the primary key, specified in three parts (*dbalias.creatorid.tablename*), is required following the **CREATE PK** keyword.

DESC *//description//*
Description of the primary key, delimited by double slashes.

BASECREATORID *basecreatorid*
Creator ID used to create the primary key, if it is generic.

COLS (*column,column***)**
This keyword is followed by a list of the columns in the primary key, enclosed in parentheses. Column names must be separated by commas and listed in order of precedence.

Relationships

The following syntax is used to generate a statement for each exported relationship.

```
CREATE REL relname
   [DESC //description//]
   CHI ctable
   PAR ptable
   [BASECREATORID basecreatorid]
   (par-expr = chi-expr
   [, par-expr = chi-expr]);
```

Keywords

The keywords correspond to values that can be specified for a relationship. For more information about an individual parameter, see Chapter 9, "Relationships," on page 213.

REL *relname*
The name of the relationship, specified as a single string (*relname*), is required following the **CREATE REL** keyword.

DESC *//description//*
Description of the relationship, delimited by double slashes.

CHI *ctable*
The fully qualified name of the child table included in the relationship.

PAR *ptable*
The fully qualified name of the parent table included in the relationship.

BASECREATORID *basecreatorid*
Creator ID used to create the relationship, if generic.

The list of corresponding columns or expressions that participate in the relationship is enclosed in parentheses with the parent table expression on the left of the equal sign and the child table expression on the right. The entries must be separated by commas. If an expression exceeds the length of the line, a break is placed before or after the expression or the equal sign, or after a comma or any space within the expression. Continuation characters are not used.

The following parent table to child table mapping information is required for each pair of tables in the relationship.

par-expr
The parent table expression.

chi-expr
The child table expression.

Storage Profiles

The following syntax is used to generate a statement for each exported Storage Profile.

```
CREATE STORPROF storageprofilename
   [DESC //description//]
   FIXED_SEGMENT_SIZE n
   REMOVABLE_SEGMENT_ SIZE n
   CREATE_DUPLICATE {Y|N}
   PRIM_PERFORM_BACKUP {Y|N}
   [PRIM_BACKUP_TAPESYS {TIVOLI | NETWORKER |CENTERA}]

   WORM_PRIM_TYPE {SnapLock | Arc | HCAP | Other}
   WORM_PRIM_MIN_ RETENTION {Interval | Max | None}
   WORM_PRIM_MIN_ RETENTION_YEARS n
   WORM_PRIM_MIN_ RETENTION_DAYS n
   [DUPL_ALTFILEPATH //path//
```

```
WORM_DUPL_TYPE {SnapLock | Arc | HCAP | Other}
WORM_DUPL_MIN_ RETENTION {Interval | Max | None}
WORM_DUPL_MIN_ RETENTION_YEARS n
WORM_DUPL_MIN_ RETENTION_DAYS n
[TIV_RECALLPATH //path//
TIV_OPTSPATH //path//
TIV_FILESPACE_PREFIX filespaceprefix
TIV_MANAGEMENT_CLASS mgmtclass]
[NET_RECALLPATH //path//
NET_BACKUPTYPE {SAVE | ARCHIVE}
[NET_VERIFY_ARCHIVE {Y|N}]
NET_USERID userid
NET_PASSWORD password
NET_DOMAIN domain
NET_SERVER server
[NET_POOL poolname]
[NET_CLONEPOOL clonepoolname]
[NET_GROUP groupname]]
[CEN_RECALLPATH //path//
MIN_RETENTION {DEFAULT | NONE | INTERVAL |INFINITE}

[MIN_RETENTION_YEARS n]
[MIN_RETENTION_DAYS n]
CEN_POOLADDRESSES //addr1, addr2, ... addrn//]
[ARC_RETENTION {I|A|K}
[ARC_RETENTION_DAYS n
ARC_RETENTION_HRS n]
RECALL_RETENTION {I|A}
[RECALL_RETENTION_DAYS n
RECALL_RETENTION_HRS n]]
FILE_AUTO_DELETE n;
```

Keywords

The keywords correspond to values that can be specified for a Storage Profile. For more information about individual parameters, see the *Archive User Manual*

.

STORPROF *storageprofilename*
The name of the Storage Profile, specified as a single string (*storageprofilename*), is required following the **CREATE STORPROF** keyword.

DESC *//description//*
Description of the Storage Profile, delimited by double slashes.

FIXED_SEGMENT_SIZE *n*
Segment size in megabytes (0 - 9999) for use when the target destination is a fixed drive (i.e. hard disk).

REMOVABLE_SEGMENT_SIZE *n*
Segment size in megabytes (1 - 9999) for use when the target destination is a removable device (i.e., floppy disk, zip drive).

CREATE_DUPLICATE {Y|N}
Indicates whether a duplicate Archive File is created.

Y Duplicate is created.

N Duplicate is not created.

Primary Copy

PRIM_PERFORM_BACKUP {Y|N}
Indicates whether to copy the primary Archive File to a backup device.

Y Copy to a backup device.

N Copy to a backup device.

PRIM_BACKUP_TAPESYS {TIVOLI | NETWORKER | CENTERA}
For the primary Archive File, copy to and recall from the selected backup device.

TIVOLI
Use IBM Tivoli.

NETWORKER
Use EMC NetWorker.

CENTERA
Use EMC Centera.

WORM_PRIM_TYPE {SnapLock | Arc | HCAP | Other}
For the primary Archive File, store the file on the selected WORM device.

SnapLock
NetApp SnapLock

Arc Archivas ArCSan

HCAP Hitachi Data Systems

Other A WORM device is used, but no retention period is defined.

WORM_PRIM_MIN_RETENTION {Interval | Max | None}
Indicates the period of time for which a primary Archive File on a WORM device is protected from deletion, measured from the time the Archive Process saves the file to the device.

Interval
Use a number of years, days, or a combination of both.

Max Use the maximum date, 01/17/2071.

None Do not use a retention period; allow the file to be deleted at any time.

WORM_PRIM_MIN_RETENTION_YEARS *n*
Specifies the number of years to protect a primary Archive File on a WORM device from deletion. The default value is zero (0).

WORM_PRIM_MIN_RETENTION_DAYS *n*
Specifies the number of days (0 to 999) to protect a primary Archive File on a WORM device from deletion. The default value is zero (0).

Duplicate Copy

DUPL_ALTFILEPATH *//path//*
Specifies the path to a directory for the duplicate Archive File.

WORM_DUPL_TYPE {SnapLock | Arc | HCAP | Other}
For the duplicate Archive File, store the file on the selected WORM device.

SnapLock
NetApp SnapLock

Arc Archivas ArCSan

HCAP Hitachi Data Systems

Other A WORM device is used, but no retention period is defined.

Note: If Other is selected, or users are not allowed to override the WORM device default set in Product Options, this is the only WORM device keyword for the duplicate file.

WORM_DUPL_MIN_RETENTION {Interval | Max | None}
Indicates the period of time for which a duplicate Archive File on a WORM device is protected from deletion, measured from the time the Archive Process saves the file to the device.

Interval
Use a number of years, days, or a combination of both.

Max Use the maximum date, 01/17/2071.

None Do not use a retention period; allow the file to be deleted at any time.

WORM_DUPL_MIN_RETENTION_YEARS *n*
Specifies the number of years to protect a duplicate Archive File on a WORM device from deletion. The default value is zero (0).

WORM_DUPL_MIN_RETENTION_DAYS *n*
Specifies the number of days (0 to 999) to protect a duplicate Archive File on a WORM device from deletion. The default value is zero (0).

Tivoli

TIV_RECALLPATH *//path//*
Specifies the complete path to the directory for storing Archive Files when recalled from Tivoli.

TIV_OPTSPATH *//path//*
Specifies the complete path to the Tivoli options file (dsm.opt).

TIV_FILESPACE_PREFIX *filespaceprefix*
The name of the filespace prefix that identifies where a group of Archive Files are stored in Tivoli.

TIV_MANAGEMENT_CLASS *mgmtclass*
For Tivoli, the name of a management class that defines the policy for backup device operations.

NetWorker

NET_RECALLPATH *//path//*
Specifies the complete path to the directory for storing Archive Files when recalled from NetWorker.

NET_BACKUPTYPE {SAVE|ARCHIVE}
Indicates the NetWorker backup type to use during processing.

SAVE NetWorker can retain the file temporarily.

ARCHIVE
NetWorker can retain the file forever.

NET_VERIFY_ARCHIVE {Y|N}
If the NetWorker backup type is ARCHIVE, indicates whether to verify the Archive File after copying to NetWorker.

Y Verify the Archive File.

N Do not verify the Archive File.

NET_USERID *userid*
The User ID under which NetWorker commands are run.

NET_PASSWORD *password*
The password for the User ID that allows you to access the NetWorker backup device.

NET_DOMAIN *domain*
The name of the Domain under which NetWorker is run.

NET_SERVER *server*
The name of the network computer that runs the NetWorker server software.

NET_POOL *poolname*
The name of the NetWorker pool in which backed up data is stored.

NET_CLONEPOOL *clonepoolname*
The name of the NetWorker clone pool in which an exact copy of the backed up data is stored, if **NET_BACKUPTYPE** is ARCHIVE.

NET_GROUP *groupname*
The name of the NetWorker group that determines the policy for backup device operations, if **NET_BACKUPTYPE** is SAVE.

Centera

CEN_RECALLPATH *//path//*
Specifies the complete path to the directory for storing Archive Files when recalled from Centera.

MIN_RETENTION {DEFAULT|NONE|INTERVAL |INFINITE}
Indicates the period of time for which an Archive File on Centera is protected from deletion, measured from the time the Archive Process copies the file to Centera.

DEFAULT
Use the Centera default retention period.

NONE
Do not use a retention period; allow the file to be deleted at any time.

INTERVAL
Use a number of years, days, or a combination of both.

INFINITE
Keep an Archive File forever; the file cannot be deleted.

MIN_RETENTION_YEARS *n*
Specifies the number of years (0 to 100) to protect an Archive File from deletion.

MIN_RETENTION_DAYS *n*
Specifies the number of days (0 to 18300) to protect an Archive File from deletion.

CEN_POOLADDRESSES *//addr1,addr2, ... addrn//*
The DNS names or IP addresses required to access the Centera server.

File Retention

ARC_RETENTION {I|A|K}
Indicates the period of time for which the Archive File is retained on disk after copying to a backup device.

I Delete the Archive File from disk immediately after the Archive Process is complete.

A Delete the Archive File from disk after the specified period, measured from the time the Archive File was last accessed, either by Optim or other software.

K Do not delete the Archive File from disk.

ARC_RETENTION_DAYS *n*
Number of days (0 to 999) to keep the Archive File on disk since last accessed, if **ARC_RETENTION** is A.

ARC_RETENTION_HRS *n*
Number of hours (0 to 23) to keep the Archive File on disk since last accessed, if **ARC_RETENTION** is A.

RECALL_RETENTION {I|A}
Indicates how long a recalled Archive File remains on disk after the recall is complete.

I Delete the Archive File from disk immediately after the recall is complete.

A Delete the Archive File from disk a specified number of days/hours after the recall is complete.

RECALL_RETENTION_DAYS *n*
Number of days (0 to 999) to keep the recalled Archive File on disk after the recall is complete, if **RECALL_RETENTION** is A.

RECALL_RETENTION_HRS *n*
Number of hours (0 to 23) to keep the recalled Archive File on disk after the recall is complete, if **RECALL_RETENTION** is A.

Retention Policy

FILE_AUTO_DELETE *n*
Number of days (1 to 9132) to keep the primary copy of the Archive File before it is automatically deleted.

Table Maps

The following syntax is used to generate a statement for each exported Table Map:

```
CREATE TM identifier.name
   [DESC //description//]
   SRCQUAL srcqual DESTQUAL destqual
   [COLMAPID cmapid] {SRCEXT exfilname | SRCAD adname}
   SRCTYPE {X|A}
   VALRULES {M|C}
   (srctable = destable [CM cmapname|LOCALCM (cmapdef)]
   [, srctable = destable [CM cmapname|LOCALCM (cmapdef)]])
   [ARCHACTS {ACTION
      {SRP | BRFRT | BRR | ARR | ARLRT | ERP}
      SQL //SQL statement//  [HOSTVAR {~ | ! | @ | $ | : | % | + | ? }]  |
      SAMEAS  actionname}[DBALIAS dbalias]
      [ON_ERROR  {STOP | SKIP | PROCESS}];
```

Keywords

The keywords correspond to values that can be specified for a Table Map. For more information about an individual parameter, see “Open the Table Map Editor” on page 229.

TM *identifier.name*
The name of the Table Map, specified in two parts (*identifier.name*), is required following the **CREATE TM** keyword.

DESC *//description//*
A description of the Table Map, delimited by double slashes.

SRCQUAL *srcqual*
(**Source Qualifier**) indicates the Default Qualifier for the source table(s) defined in the Table Map.

DESTQUAL *destqual*
(**Destination Qualifier**) indicates the Default Qualifier for the destination table(s) defined in the Table Map.

COLMAPID *cmapid*
(**Default Column Map ID**) indicates the Default Qualifier for the Column Maps included in the Table Map.

SRCEXT *exfilname*
Name of the Extract File used as the source. This keyword is included only when the source is an Extract File.

SRCAD *adname*
Name of the Access Definition used as the source. This keyword is included only when the source is an Access Definition.

SRCTYPE {X|A}
Indicates whether the source type is an Extract File (X) or an Access Definition (A). MVS definitions allow "T," which is invalid for Optim.

VALRULES {M|C}
Indicates the validation rules for the Table Map.

M Indicates Move/Archive Table Map validation for Convert, Create, Insert, or Load Requests.

C Indicates Compare Table Map validation for Compare Requests.

The following source table to destination table mapping information is required for each pair of tables in the Table Map. At least one pair of tables must be specified.

srctable
The name of the source table. The Creator ID is included only when it differs from the default specified for **SRCQUAL**.

destable
The name of the destination table. The Creator ID is included only when it differs from the default specified for **DESTQUAL**. The words "NOT SPECIFIED" are inserted when the destination table is omitted.

One of the following is included when a Column Map is specified for a pair of tables.

CM The name of the Column Map for the pair of tables. The Map ID is included only when it differs from the **COLMAPID** value.

LOCALCM
The local Column Map definition enclosed in parentheses. Only the (*src-expr* = *dest-col*) parameter of the Column Map statement is included.

Note: If a Table Map contains Column Maps which are exported as subordinate objects, the Column Maps are exported before the Table Map is exported.

Archive Actions Keyword

If a table has one or more Archive Actions defined from the original Access Definition, the ARCHACTS keyword is added.

ACTION
The Action Phase. If an ACTION is specified, SQL parameters are required.

ACTION value	Phase
SRP	Start of Restore Process
BRFRT	Before Restore of First Row to Table
BRR	Before Restore of Row
ARR	After Restore of Row
ARLRT	After Restore of Last Row to Table
ERP	End of Restore Process

SQL *//SQL stmt//*

The text of the SQL WHERE clause. Long character strings such as SQL WHERE clauses are delimited by two forward slashes (//). Continuation characters are not used, and no spaces or indentations are added. If additional spaces or continuation characters are inserted, the string is imported incorrectly. The text continues for the complete length of the line width, wrapping to the next line until the forward slashes are reached, indicating the end of the text.

Note: Archive Actions for all processes may be defined in an Access Definition. Archive Actions defined for the Restore Process in a Table Map override those same actions defined in an Access Definition. With this in mind, you can define Archive Actions in a Table Map with an empty SQL Statement (e.g. SQL // //) to prevent an Archive Action defined in the Access Definition from being executed.

HOSTVAR *c*

Identifies the variable delimiter. If **HOSTVAR** is supplied, the delimiter must be one of the following:
~ ! @ $: % + ?

If **HOSTVAR** is not supplied, the colon symbol (:) is used by default.

SAMEAS *actionname*

Specifies that the same SQL statement is used as for the named Action Phase. If **SAMEAS** is specified, SQL parameters are not required.

DBALIAS *dbalias*

If **DBALIAS** is supplied, the value used must refer to a valid DB Alias of the same DBMS type as the target table.

ON_ERROR {STOP|SKIP|PROCESS}

If **ON_ERROR** is supplied, the value STOP, SKIP, or PROCESS must be supplied.

If **ON_ERROR** is not supplied, the value STOP is used by default.

Unsupported Keywords

The following keywords and associated values may be present if the file was created using Optim z/OS. These keywords are not valid when importing the definition to Optim and must be removed from the statement.

DESTEXT *dsname*
DESTAD *adname*
DESTTYPE {X|A|T}

Archive Requests

The following syntax is used to generate a statement for each exported Archive Request definition:

```
CREATE ARCH  id.name
   [DESC //description//]
   AF archive file
   [AFX  archive index file]
   [GROUP group]
   [STORAGE_PROFILE storprofname]
   [ROWLIMIT n]
   [SERVERNAME server]
   DEFER_DAA {Y|N}
   REVIEW_DELETE {Y|N}
   COMPRESSFILE {Y|N}
   GENSTATISTIC {Y|N}
   PROCESS_ FILEATTACH {Y|N}
   CREATEREPORT {Y|N}
```

```
{AD adname | LOCALAD (addef)}
[{REPORT id.name | LOCALREPORT (reportdef)}]
[INCLPK {Y|N}] [INCLFK {Y|N}]
[INCLIDX {Y|N}] [INCLALIAS {Y|N}]
[INCLFUNCTION {Y|N}] [INCLPACKAGE {Y|N}]
[INCLPROCEDURE {Y|N}] [INCLSEQUENCE {Y|N}]
[INCLTRIGGER {Y|N}] [INCLVIEW {Y|N}]
[INCLDEF {Y|N}] [INCLRULE {Y|N}] [INCLUDT {Y|N}]
PNSOVERRIDE {Y|N}
PNSOPT {N|L|F}
[ROWLIST filename]
[LOCALRL //string//]
[VARS //string// ALWAYSPROMPT {Y|N}]
[CF control file  DELCF {Y|N}
   DISCARDLIMIT n COMMITFREQ n LOCKTBLS {Y|N}]
OBJQUAL dbalias[.creator]
IGNOREUNKNOWN {Y|N}
[OBJECT ( objname =
   {FUNCTION | PACKAGE | PROCEDURE | SEQUENCE |
      VIEW | DEFAULT | RULE | UDT } objname ... )]
[EMAILNOTIFY ({A|S|F} emailaddress)];
```

Keywords

The keywords correspond to values that can be specified for an Archive Request.

ARCH *id.name*
The name of the Archive Request, specified in two parts (*identifier.name*) is required following the **CREATE ARCH** keyword.

DESC *//description//*
A description of the Archive Request, delimited by double slashes.

General

AF *archivefile*
The fully qualified name of the Archive File, which is the output of the Archive Process.

AFX *archiveindexfile*
The fully qualified name of the Archive Index File, which is the file created when Archive index criteria is specified in the Access Definition for the Archive Process.

GROUP *group*
Logical group name to help qualify and categorize the Archive File and corresponding archived data.

STORAGE_PROFILE *storprofname*
The name of the Storage Profile for the Archive Request. The name of the Storage Profile follows the keyword.

ROWLIMIT *n*
A limit for the number of rows (*n*) that can be archived. If the Archive Definition does not include a Row Limit, the value is "0."

SERVERNAME *server*
If the optional Optim Server is installed on your network, specifies the server on which to process the request.

DEFER_DAA {Y|N}
When the Access Definition includes instructions to delete archived rows, this keyword can instruct Archive to bypass those instructions.

Y Do not delete archived rows as part of the Archive Process.

N Delete archived rows

REVIEW_DELETE {Y|N}
Displays the Delete After Archive Specifications dialog during processing for review and override of Access Definition delete options for the tables to be archived.

COMPRESSFILE {Y|N}
Indicator for whether to compress the Archive File after processing.

GENSTATISTIC {Y|N}
Indicates if statistical information is included in the Archive Process Report.

PROCESS_FILEATTACH {Y|N}
Indicates if file attachments specified in the Access Definition are extracted.

CREATEREPORT {Y|N}
Indicator for whether to create a Report on the Archive File after processing.

AD Keywords

An Access Definition parameter must be included. The Access Definition can be Local or Named.

AD *adname*
The full two-part name of the Named Access Definition being used.

LOCALAD (*addef***)**
A full definition of the Local Access Definition being used, contained in parentheses, follows this keyword. The syntax and parameters for the definition of a Local Access Definition are identical to those for a Named Access Definition.

Report Keywords

The optional Report parameter can be defined as Local or Named:

REPORT *id.name*
The full two-part name of the Named Report Request Definition being used.

LOCALREPORT (*reportdef***)**
A full definition of the Local Report being used, contained in parentheses, follows this keyword. The syntax and parameters for the definition of a Local Report are identical to those for a Named Report Request.

Objects

This parameter relates to the ability to archive objects when performing the Archive Process.

INCLPK {Y|N}
Indicates whether primary keys are included in the Archive Process.

INCLFK {Y|N}
Indicates whether foreign keys and relationships are included in the Archive Process.

INCLIDX {Y|N}
Indicates whether indexes are included in the Archive Process.

INCLALIAS {Y|N}
Indicates whether aliases are included in the Archive Process.

INCLFUNCTION {Y|N}
Indicates whether functions are included in the Archive Process.

INCLPACKAGE {Y|N}
Indicates whether packages are included in the Archive Process.

INCLPROCEDURE {Y|N}
Indicates whether procedures are included in the Archive Process.

INCLSEQUENCE {Y|N}
Indicates whether sequences are included in the Archive Process.

INCLTRIGGER {Y|N}
Indicates whether triggers are included in the Archive Process.

INCLVIEW {Y|N}
Indicates whether views are included in the Archive Process.

INCLDEF {Y|N}
Indicates whether defaults are included in the Archive Process.

INCLRULE {Y|N}
Indicates whether rules are included in the Archive Process.

INCLUDT {Y|N}
Indicates whether UDTs are included in the Archive Process.

Point and Shoot

The **PNSOPT**, **LOCALRL** and **ROWLIST** keywords are related:

- **PNSOPT** or **PNSOVERRIDE** specifies the Point and Shoot option for the Archive Request.
- If **PNSOPT** is F, the **ROWLIST** parameter must follow.
- If **PNSOPT** is L, the **LOCALRL** parameter *may* follow, as an option (you can select LOCAL for Point and Shoot without defining a Row List file).
- **ROWLIST** may be included in the exported definition even if the **PNSOPT** is N or L (or omitted), because the Archive Request may have previously had a **ROWLIST** in use. A previously used Named Row List file reference will be retained as part of the Archive Request, even if the **PNSOPT** is no longer F.

PNSOVERRIDE {Y|N}
Indicates whether to override the Point and Shoot Definition in the Access Definition for the Archive Request. Specify Y to override the Point and Shoot Definition in the Access Definition. Specify N to use the Point and Shoot Definition in the Access Definition.

PNSOPT {L|F|N}
(**Point and Shoot**) indicates the Point and Shoot option for the Archive Request. The options are:

L Local Row List; **LOCALRL** and **ROWLIST** are optional additional keywords.

F Named Row List; **ROWLIST** must follow.

N None; **ROWLIST** is an optional additional keyword.

ROWLIST *filename*
The fully qualified name of the external Named Row List for the Archive Request.

LOCALRL *//string//*
The definition of the Local Row List for the Archive Request. Long character strings such as Row List definitions are delimited by two forward slashes (//). Continuation characters are not used, and no spaces or indentations are added. If additional spaces or continuation characters are inserted, the string is imported incorrectly. The text continues for the complete length of the line, wrapping to the next line until the forward slashes are reached, indicating the end of the text.

Variables

There is a **VARS** entry only when a variable(s) has been defined for the Access Definition being used in the Archive Request.

VARS *//string//*
A list of the variables used in the Archive Request:

- The information is delimited by double slashes.
- Commas are used between variables.
- Single slashes are used between variable names and default values.

ALWAYSPROMPT {Y|N}
(**Always Prompt for Values at Run Time**) indicates whether a prompt for variable values is automatically displayed during Archive Processing.

Y The prompt for variable values is automatically displayed during Archive Processing.

N The prompt for variable values is not automatically displayed during Archive Processing.

Note: Prompt strings are not exported as part of the Archive Request, only variable names and default values. If the Access Definition used in the Archive Request is exported as a subordinate object (or if a Local Access Definition is used), prompt strings are exported as part of the Access Definition.

Delete

There are Delete options only when the keyword value for DEFER_DAA is N.

CF *control file*
The fully qualified name of the Control File for the Delete Process, as applicable.

DELCF {Y|N}
Indicator for whether to discard the Control File if action is successful.

DISCARDLIMIT *n*
The maximum number of rows (*n*) that can be discarded.

COMMITFREQ *n*
The frequency (every *n*th row) at which commits are issued.

LOCKTBLS {Y|N}
Indicates whether tables are locked during the Delete.

Y Tables are locked.

N Tables are not locked.

Object List

This parameter relates to the ability to extract extended objects.

OBJQUAL *dbalias.creator*
The name of the Default Qualifier for the extended objects (*dbalias.creator*).

IGNOREUNKNOWN {Y|N}
Indicator for ignoring unknown or unavailable objects.

OBJECT *objname=type*
Each extended object, identified by name, and type of object: function, package, procedure, sequence, view, default, rule, or UDT.

Notify

This parameter relates to the ability to send notification via email about the outcome of the process.

EMAILNOTIFY {A|S|F} *emailaddress*
Indicates the process outcome under which to send notification to the corresponding email address.

A Always send notification.

S Send notification when process succeeds.

F Send notification when process fails.

Compare Requests

The following syntax is used to generate a statement for each exported Compare Request definition:

```
CREATE COMP identifier.name
   [DESC //description//]
   SRCTYPE srctype
   SRC1XF src1xfname SRC2XF src2xfname CMF cmpname
   BROWSERESULTS {Y|N}
   CREATEREPORT {Y|N}
   EXTRACTRUNMODE {S|P}
   FORCEEDITTM {Y|N}
   {LOCALTM (localtmdef) | TM tmname}
   {LOCALAD1 (localad1def) | AD1 ad1name}
   {LOCALAD2 (localad2def) | AD2 ad2name}
   MATCHKEY (matchkeydef)
   [{REPORT id.name | LOCALREPORT (reportdef)}]
   [EMAILNOTIFY ({A|S|F} emailaddress)];
```

Keywords

The keywords correspond to values that can be specified for a Compare Request. For more information about individual parameters, see the *Compare User Manual* .

COMP *identifier.name*
The name of the Compare Request, specified in two parts (*identifier.name*), is required following the **CREATE COMP** keyword.

DESC *//description//*
A description of the Compare Request, delimited by double slashes.

SRCTYPE *srctype*
Indicates S or M for single or multiple tables, and then E for Extract File, A for Access Definition, or D for database tables for each source. (For example, “MAA” indicates Multiple tables, comparing a Source 1 Access Definition to a Source 2 Access Definition.)

SRC1XF *src1xfname*
The fully qualified name of the Source 1 Extract File used as input for the Compare Process.

SRC2XF *src2xfname*
The fully qualified name of the Source 2 Extract File used as input for the Compare Process.

CMF *cmpname*
The fully qualified name of the Compare File for the Compare Process.

BROWSERESULTS {Y|N}
Indicates whether the Browse dialog opens immediately upon completion of the Compare Process.

Y The Browse dialog opens automatically.

N The Browse dialog does not open automatically.

CREATEREPORT {Y|N}
Indicator for whether to create a Report on the Compare File after processing.

EXTRACTRUNMODE {S|P}
When both Source 1 and Source 2 require an Extract Process to be performed, **EXTRACTRUNMODE** (Run Mode for Extract) indicates whether the two Extract Processes should run in sequence or parallel.

S Run the Extract Processes in sequence (one, then the other).

P Run the Extract Processes in parallel (both at the same time).

FORCEEDITTM {Y|N}
(**Always View Table Map**) indicates whether the Table Map Editor opens automatically when processing the Compare Request.

Y The Table Map Editor opens automatically.

N The Table Map Editor does not open automatically.

LOCALTM (*localtmdef***)**
The definition of a Local Table Map, enclosed in parentheses.

TM *tmname*
The name of the Named Table Map for the Compare Request. The name of the Table Map follows the keyword.

LOCALAD1
A full definition of the Local Access Definition for Source 1 (or Source 2), contained in parentheses, follows this keyword.

LOCALAD2
A full definition of the Local Access Definition for Source 1 (or Source 2), contained in parentheses, follows this keyword.

AD1 The full two-part name of the Named Access Definition for Source 1 (or Source 2) for the Compare Request. The name of the Access Definition follows the keyword.

AD2 The full two-part name of the Named Access Definition for Source 1 (or Source 2) for the Compare Request. The name of the Access Definition follows the keyword.

MATCHKEY (*matchkeydef***)**
The Source 1 table name(s) and column name(s), enclosed in parentheses, used in the match key definition.

Report Keywords

The optional Report parameter can be defined as Local or Named:

REPORT *id.name*
The full two-part name of the Named Report Request Definition being used.

LOCALREPORT (*reportdef***)**
A full definition of the Local Report being used, contained in parentheses, follows this keyword. The syntax and parameters for the definition of a Local Report are identical to those for a Named Report Request.

Notify

This parameter relates to the ability to send notification via email about the outcome of the process.

EMAILNOTIFY {A|S|F} *emailaddress*
Indicates the process outcome under which to send notification to the corresponding email address.

A Always send notification.

S Send notification when process succeeds.

F Send notification when process fails.

Convert Requests

The following syntax is used to generate a statement for each exported Convert Request definition:

```
CREATE CONV identifier.name
   [DESC //description//]
   SRCXF srcxfname DESTXF destxfname
   CF cfname {LOCALTM (localtmdef) | TM tmname}
   [DISCARDLIMIT n]
   COMPRESSFILE {Y|N}
   INCL_FILEATTACH {Y|N}
   FORCEEDITTM{Y|N}SHOWCURRENCY{Y|N}
   SHOWAGE {Y|N}
   [FUNCTION_AGING (AGETYPE agetype
      [YEARS nn] [MONTHS nn]
      [WEEKS nn] [DAYS nn] [SPECIFICYEAR nnnn]
      [SPECIFICDATE mm/dd/yyyy]
      [TARGETSTART mm/dd/yyyy TARGETEND mm/dd/yyyy]
      [MULTIPLE nn [RULE rulename]
      [CALENDAR calendarname]
      [PIVOT nn] [INVALIDDATES {Y|N}]
      [SKIPPEDDATES {Y|N}])]
   [GLOBAL_AGING (AGETYPE agetype
      [YEARS nn] [MONTHS nn]
      [WEEKS nn] [DAYS nn] [SPECIFICYEAR nnnn]
      [SPECIFICDATE mm/dd/yyyy]
      [TARGETSTART mm/dd/yyyy TARGETEND mm/dd/yyyy]
      [MULTIPLE nn] [RULE rulename]
      [CALENDAR calendarname]
      [PIVOT nn] [INVALIDDATES {Y|N}]
      [SKIPPEDDATES {Y|N}])]
   [REPORT_OPTION (RPTERROR {T|F} [MAXTBLERR nnn]
      [MAXRUNERR nnn] [RPTSUMMARY {T|F}]
      [RPTINVALID{T|F}]
      [RPTSKIPPED {T|F}])]
   [CURRENCY_OPTION (DEFAULT currencytablename
      [GLOBAL currencytablename] [FROM currencytype]
      [TO currencytype] [TRIANG {T|F}])]
   [EMAILNOTIFY ({A|S|F} emailaddress)]
   [CSF (TABLE (tablename OUTCOLS (columnname)
      OUTPOSITION (n))];
```

Keywords

The keywords correspond to values that can be specified for a Convert Request. For additional information about an individual parameter, see the *Move User Manual* .

CONV *identifier.name*
: The name of the Convert Request, specified in two parts (*identifier.name*), is required following the **CREATE CONV** keyword.

DESC *//description//*
: A description of the Convert Request, delimited by double slashes.

General

SRCXF *srcxfname*
: The fully qualified name of the Source Extract File used as input for the Convert Process.

DESTXF *destxfname*
: The fully qualified name of the Destination Extract File to be the output of the Convert Process.

CF *cfname*
: The fully qualified name of the Control File for the Convert Process.

LOCALTM (*localtmdef***)**
The definition of a Local Table Map, enclosed in parentheses. For details on specifying Table Maps, see Chapter 10, "Table Maps," on page 229.

TM *tmname*
The name of the Named Table Map for the Convert Request. The name of the Table Map follows the keyword.

DISCARDLIMIT *n*
(**Discard Row Limit**) indicates a limit (*n*) for discarded rows, if one is included.

COMPRESSFILE {Y|N}
Indicates whether to compress the Destination Extract File after processing.

INCL_FILEATTACH {Y|N}
Indicates whether to include file attachments in the destination file. File attachments are not converted.

FORCEEDITTM {Y|N}
(**Always View Table Map**) indicates whether the Table Map Editor opens automatically when processing the Convert Request.

Y The Table Map Editor opens automatically.

N The Table Map Editor does not open automatically.

SHOWCURRENCY {Y|N}
Indicates whether the **Currency** tab displays.

Y Currency tab displays.

N Currency tab does not display.

SHOWAGE {Y|N}
Indicates whether the **Aging** tabs display.

Y Age Function and Global Aging tabs display.

N Age Function and Global Aging tabs do not display

Age Function

If options for the Aging function are specified, they follow the FUNCTION_AGING keyword in parentheses.

FUNCTION_AGING
Indicates date aging function is used for the Convert.

AGETYPE *agetype*
The function type specified as one of the following:

NONE
INCREMENTAL
SPECIFICYEAR
SPECIFICDATE
TARGETDATES
RULEBASED

YEARS *nn*
The positive or negative numeric value (*nn*) to age the date by years for incremental aging.

MONTHS *nn*
The positive or negative numeric value (*nn*) to age the date by months for incremental aging.

WEEKS *nn*
The positive or negative numeric value (*nn)* to age the date by weeks for incremental aging.

DAYS *nn*
The positive or negative numeric value (*nn*) to age the date by days for incremental aging.

SPECIFICYEAR *nnnn*
An explicit year (*nnnn*) to which the Convert should age.

SPECIFICDATE *mm/dd/yyyy*
The date (*mm/dd/yyyy*) to which the Convert should age.

TARGETSTART *mm/dd/yyyy*
The date (*mm/dd/yyyy*) Convert should use as the starting date when calculating the amount to age using relative aging.

TARGETEND *mm/dd/yyyy*
The date (*mm/dd/yyyy*) Convert should use as the target date when calculating the amount to age using relative aging.

MULTIPLE *nn*
The number of times (*nn*) Convert should apply a rule for date aging.

RULE *rulename*
The name of the rule (*rulename*) being used.

CALENDAR *calendarname*
The name of the calendar (*calendarname*) being used.

PIVOT *nn*
The pivot year (*nn*).

INVALIDDATES {Y|N}
Indicates whether to Convert rows with dates that are invalid to use.

SKIPPEDDATES {Y|N}
Indicates whether to Convert rows with dates that should be skipped.

Global Aging

If options for global aging are specified, they follow the GLOBAL_AGING keyword in parentheses.

GLOBAL_AGING
Indicates global date aging is used in conversion.

AGETYPE *agetype*
The function type specified as one of the following:

NONE
INCREMENTAL
SPECIFICYEAR
SPECIFICDATE
TARGETDATES
RULEBASED

YEARS *nn*
The positive or negative numeric value (*nn*) to age the date by years for incremental aging.

MONTHS *nn*
The positive or negative numeric value (*nn*) to age the date by months for incremental aging.

WEEKS *nn*
The positive or negative numeric value (*nn*) to age the date by weeks for incremental aging.

DAYS *nn*
The positive or negative numeric value (*nn*) to age the date by days for incremental aging.

SPECIFICYEAR *nnnn*
An explicit year (*nnnn*) to which the Convert should age.

SPECIFICDATE *mm/dd/yyyy*
The date (*mm/dd/yyyy*) to which the Convert should age.

TARGETSTART *mm/dd/yyyy*
The date (*mm/dd/yyyy*) Convert should use as the starting date when calculating the amount to age using relative aging.

TARGETEND *mm/dd/yyyy*
The date (*mm/dd/yyyy*) Convert should use as the target date when calculating the amount to age using relative aging.

MULTIPLE *nn*
The number of times (*nn*) Convert should apply a rule for date aging.

RULE *rulename*
The name of the rule (*rulename*) being used.

CALENDAR *calendarname*
The name of the calendar (*calendarname*) being used.

PIVOT *nn*
The pivot year (*nn*).

INVALIDDATES {Y|N}
Indicates whether to process rows with dates that are invalid to use.

SKIPPEDDATES {Y|N}
Indicates whether to process rows with dates that should be skipped.

Report Options

If Report options are specified, they follow the REPORT_OPTION keyword in parentheses.

REPORT_OPTION
Indicates reporting options are used for the Convert.

RPTERROR {T|F}
Indicates whether to report errors (True or False).

MAXTBLERR *nnn*
The maximum number (*nnn*) of errors per table to report.

MAXRUNERR *nnn*
The maximum number (*nnn*) of errors per run to report.

RPTSUMMARY {T|F}
Indicates whether to report the Aging summary (True or False).

RPTINVALID {T|F}
Indicates whether to report invalid dates (True or False).

RPTSKIPPED {T|F}
Indicates whether to report skipped dates (True or False).

Currency

If options for currency conversion are specified, they follow the CURRENCY_OPTION keyword in parentheses.

CURRENCY_OPTION
Indicates options for currency conversion are used for the Convert.

DEFAULT *currencytablename*
The name of the default Currency Table (*currencytablename*) being used.

GLOBAL *currencytablename*
The name of the global Currency Table (*currencytablename*) being used.

FROM *currencytype*
The three character code for the source currency type (*currencytype*)

TO *currencytype*
The three character code for the target currency type (*currencytype).*

TRIANG {T|F}
Indicates whether to convert currency via the euro dollar (True or False).

Notify

This parameter relates to the ability to send notification via email about the outcome of the process.

EMAILNOTIFY {A|S|F} *emailaddress*
Indicates the process outcome under which to send notification to the corresponding email address.

A Always send notification.

S Send notification when process succeeds.

F Send notification when process fails.

Comma Separated

If options for creating a Comma Separated file are specified, they follow the CSF keyword in parentheses.

CSF Indicates a Comma Separated file is specified.

TABLE
This keyword precedes each fully qualified table name listed on the **Comma Separated** tab of the Convert Request. The table name is followed by additional keywords to indicate the relationship between the joined tables.

OUTCOLS
The names of any columns from the listed table included in each outputted row of the Comma Separated file.

OUTPOSITION
The relative position of each column in the outputted row of the Comma Separated file.

Delete Requests

The following syntax is used to generate a statement for each exported Delete Request definition:

```
CREATE DEL identifier.name
   [DESC //description//]
   XF xfilname CF cfilname DISCARDLIMIT n COMMITFREQ n
   LOCKTBLS {Y|N} INCLUDE_LOBS {Y|N}
   [EMAILNOTIFY ({A|S|F} emailaddress)];
```

Keywords

The keywords correspond to values that can be specified for a Delete Request. For more information about an individual parameter, see the *Archive User Manual* .

DEL *identifier.name*
The name of the Delete Request, specified in two parts (*identifier.name*), is required following the **CREATE DEL** keyword.

DESC *//description//*
A description of the Delete Request, delimited by double slashes.

General

XF *xfilname*
The fully qualified name of the Extract File used as input for the Delete Process.

CF *cfilname*
The fully qualified name of the Control File for the Delete Process.

DISCARDLIMIT *n*
The maximum number of rows (*n*) that can be discarded.

COMMITFREQ *n*
The frequency (every *n*th row) at which commits are issued.

LOCKTBLS {Y | N}
(**Lock Tables**) indicates whether tables are locked during the Delete.

Y Tables are locked.

N Tables are not locked.

INCLUDE_LOBS {Y | N}
Indicates whether LOB columns are included in the row comparison performed before rows are deleted.

Y LOB Columns are included.

N LOB Columns are not included.

Notify

This parameter relates to the ability to send notification via email about the outcome of the process.

EMAILNOTIFY {A | S | F} *emailaddress*
Indicates the process outcome under which to send notification to the corresponding email address.

A Always send notification.

S Send notification when process succeeds.

F Send notification when process fails.

Extract Requests

The following syntax is used to generate a statement for each exported Extract Request definition:

```
CREATE EXTR identifier.name
   [DESC //description//] XF xfilname
   {AD adname | LOCALAD (addef)}
   PNSOVERRIDE {Y|N} PNSOPT {N [ROWLIST filname] |
      L [LOCALRL//rowlistdef//]
      [ROWLIST filname] | F ROWLIST filname}
   [PNSSTART starttable]
   [VARS//VarName/dfltvalue[VarName/dfltvalue]//
      ALWAYSPROMPT {Y|N}]
   OPTION {D|O|B} [INCLPK {Y|N}] [INCLFK {Y|N}]
      [INCLIDX {Y|N}] [INCLALIAS {Y|N}]
      [INCLFUNCTION {Y|N}] [INCLPACKAGE {Y|N}]
      [INCLPROCEDURE {Y|N}] [INCLSEQUENCE {Y|N}]
```

```
      [INCLTRIGGER {Y|N}] [INCLVIEW {Y|N}]
      [INCLDEFAULT {Y|N}] [INCLRULE {Y|N}]
      [INCLDT {Y|N}]
   [COMPRESSFILE {Y|N}] [GENSTATISTIC {Y|N}]
   [PROCESS_ FILEATTACH {Y|N}] ROWLIMIT n
   [DBCONNECTIONS n]
   [OBJECT (name1 = {FUNCTION | PACKAGE | PROCEDURE |
SEQUENCE | VIEW}  namen = {FUNCTION | PACKAGE |
PROCEDURE | SEQUENCE | VIEW}]
   [OBJQUAL identifier.name] [IGNOREUNKNOWN {Y|N}]
   [EMAILNOTIFY ({A|S|F} emailaddress)]
   [{CV identifier.name | LOCALCV (cvdef) } CVDELEXF {Y|N}];
```

Keywords

The keywords correspond to values that can be specified for an Extract Request. For more information about individual parameters, see the *Move User Manual* or the *Compare User Manual*

EXTR *identifier.name*
The name of the Extract Request, specified in two parts (*identifier.name*), is required following the **CREATE EXTR** keyword.

DESC *//description//*
A description of the Extract Request, delimited by double slashes.

XF *xfilname*
The fully qualified name of the Extract File, which is the output of the Extract Process.

AD Keywords

An Access Definition parameter must be included. The Access Definition can be Local or Named:

AD *adname*
The full two-part name of the Named Access Definition being used.

LOCALAD (*addef***)**
A full definition of the Local Access Definition being used, contained in parentheses, follows this keyword. The syntax and parameters for the definition of a Local Access Definition are identical to those for a Named Access Definition.

Point and Shoot

The **PNSOPT**, **LOCALRL** and **ROWLIST** keywords are related:

- **PNSOPT** or **PNSOVERRIDE** specifies the Point and Shoot option for the Extract Request.
- If **PNSOPT** is F, the **ROWLIST** parameter must follow.
- If **PNSOPT** is L, the **LOCALRL** parameter *may* follow, as an option (you can select LOCAL for Point and Shoot without defining a Row List file).
- **ROWLIST** may be included in the exported definition even if the **PNSOPT** is N or L (or omitted), because the Extract Request may have previously had a **ROWLIST** in use. A previously used Named Row List file reference will be retained as part of the Extract Request, even if the **PNSOPT** is no longer F.

PNSOVERRIDE {Y|N}
Indicates whether to override the Point and Shoot Definition in the Access Definition for the Extract Request.

Y Override the Point and Shoot Definition in the Access Definition.

N Do not override the Point and Shoot Definition in the Access Definition.

PNSOPT {L|F|N}
(**Point and Shoot**) indicates the Point and Shoot option for the Extract Request. The options are:

L Local Row List; **LOCALRL** and **ROWLIST** are optional additional keywords.

F Named Row List; **ROWLIST** must follow.

N None; **ROWLIST** is an optional additional keyword.

ROWLIST *filname*
The fully qualified name of the external Named Row List for the Extract Request.

LOCALRL *//rowlistdef//*
The definition of the Local Row List for the Extract Request. Long character strings such as Row List definitions are delimited by two forward slashes (//). Continuation characters are not used, and no spaces or indentations are added. If additional spaces or continuation characters are inserted, the string is imported incorrectly. The text continues for the complete length of the line, wrapping to the next line until the forward slashes are reached, indicating the end of the text.

PNSSTART *starttable*
The start table name for the Point and Shoot Row List.

Variables

There is a **VARS** entry only when a variable(s) has been defined for the Access Definition being used in the Extract Request.

VARS *//varname/dfltvalue//*
A list of the variables used in the Extract Request:

- The information is delimited by double slashes.
- Commas are used between variables.
- Single slashes are used between variable names and default values.

ALWAYSPROMPT {Y|N}
(**Always Prompt for Values at Run Time**) indicates whether a prompt for variable values is automatically displayed during Extract Processing.

Y The prompt for variable values is automatically displayed during Extract Processing.

N The prompt for variable values is not automatically displayed during Extract Processing.

Note: Prompt strings are not exported as part of the Extract Request, only variable names and default values. If the Access Definition used in the Extract Request is exported as a subordinate object (or if a Local Access Definition is used), prompt strings are exported as part of the Access Definition.

Options

This parameter relates to the ability to extract either data, objects, or data and objects when performing extracts.

OPTION {D|O|B}
Indicates whether data, objects, or both are extracted.

D Only data is extracted.

O Only objects are extracted.

B Data and objects are extracted.

INCLPK {Y|N}
Indicates whether primary keys are included in the Extract if **OPTION** is O or B.

INCLFK {Y|N}
Indicates whether foreign keys and relationships are included in the Extract if **OPTION** is O or B.

INCLIDX {Y|N}
Indicates whether indexes are included in the Extract if **OPTION** is O or B.

INCLALIAS {Y|N}
Indicates whether aliases are included in the Extract if **OPTION** is O or B.

INCLFUNCTION {Y|N}
Indicates whether functions are included in the Extract if **OPTION** is O or B.

INCLPACKAGE {Y|N}
Indicates whether packages are included in the Extract if **OPTION** is O or B.

INCLPROCEDURE {Y|N}
Indicates whether procedures are included in the Extract if **OPTION** is O or B.

INCLSEQUENCE {Y|N}
Indicates whether sequences are included in the Extract if **OPTION** is O or B.

INCLTRIGGER {Y|N}
Indicates whether triggers are included in the Extract if **OPTION** is O or B.

INCLVIEW {Y|N}
Indicates whether views are included in the Extract if **OPTION** is O or B.

INCLDEFAULT {Y|N}
Indicates whether defaults are included in the Extract if **OPTION** is O or B.

INCLRULE {Y|N}
Indicates whether rules are included in the Extract if **OPTION** is O or B.

INCLDT {Y|N}
Indicates whether UDTs (user defined types) are included in the Extract if **OPTION** is O or B.

Extract Options

The following keywords relate to options for the Extract Process.

COMPRESSFILE {Y|N}
Indicates if the process automatically compresses the Extract File for storage.

GENSTATISTIC {Y|N}
Indicates if statistical information is included in the Extract Process Report.

PROCESS_FILEATTACH {Y|N}
Indicates if file attachments specified in the Access Definition are extracted.

ROWLIMIT *n*
A limit for the number of rows (*n*) which can be extracted. The default value for this keyword is "0."

DBCONNECTIONS *n*
Indicate the number of concurrent database connections for the Extract Process. Increasing database connections improves performance when processing large quantities of data by allowing multiple threads to extract rows concurrently.

To increase the maximum number of connections, enter an even number from 2 to the site maximum (O) as specified on the Product Options dialog.

Object List

This parameter relates to the ability to extract extended objects.

OBJECT *name=type*
Each extended object, identified by name, and type of object: Default, Function, Package, Procedure, Rule, Sequence, UDT, or View.

OBJQUAL *identifier.name*
The name of the Default Qualifier for the extended objects (*identifier.name*).

IGNOREUNKNOWN {Y|N}
Indicator for ignoring unknown or unavailable objects.

Notify

This parameter relates to the ability to send notification via email about the outcome of the process.

EMAILNOTIFY {A|S|F} *emailaddress*
Indicates the process outcome under which to send notification to the corresponding email address.

A Always send notification.

S Send notification when process succeeds.

F Send notification when process fails.

Convert

This parameter relates to the ability to run the Convert Process immediately following the Extract Process. The Convert Request can be Local or Named:

CV *identifier.name*
The full two-part name of the Convert Request being used.

LOCALCV (*cvdef***)**
A full definition of the Local Convert Request being used, contained in parentheses, follows this keyword. The syntax and parameters for the definition of a Local Convert Request are identical to those for a Named Convert Request.

CVDELEXF {Y|N}
Indicator for deleting the Extract File if the Convert Process fails.

Insert Requests

The following syntax is used to generate a statement for each exported Insert Request definition:

```
CREATE UPIN identifier.name
   [DESC //description//]
   XF xfilname CF cfilname {LOCALTM (tmdef) | TM tmname}
   DISCARDLIMIT n COMMITFREQ n FORCEEDITTM {Y|N}
   LOCKTBLS {Y|N} UPDINS {U|I|M}
   DELETEROWS {A|M|N}
   [DELCOMMIT {E|C}] ALWAYSCALLCREATE {Y|N}
   TRIGMODE{A|P|N} CONSMODE{A|P|N}
   SHOWCURRENCY{Y|N}
   SHOWAGE {Y|N}  PROCESS_ FILEATTACH {Y|N}
   [FUNCTION_AGING (AGETYPE {NONE | INCREMENTAL |
      SPECIFICYEAR | SPECIFICDATE | TARGETDATES |
      RULEBASED} [YEARS nn] [MONTHS nn]
      [WEEKS nn] [DAYS nn] [SPECIFICYEAR nnnn]
      [SPECIFICDATE mm/dd/yyyy]
      [TARGETSTART mm/dd/yyyy TARGETEND mm/dd/yyyy]
      [MULTIPLE nn] [RULE rulename]
      [CALENDAR calendarname]
      [PIVOT nn] [INVALIDDATES {Y|N}]
      [SKIPPEDDATES {Y|N}])]
   [GLOBAL_AGING (AGETYPE {NONE | INCREMENTAL |
```

```
   SPECIFICYEAR | SPECIFICDATE | TARGETDATES |
   RULEBASED} [YEARS nn] [MONTHS nn] [WEEKS nn]
   [DAYS nn] [SPECIFICYEAR nnnn]
   [SPECIFICDATE mm/dd/yyyy]
   [TARGETSTART mm/dd/yyyy TARGETEND mm/dd/yyyy]
   [MULTIPLE nn] [RULE rulename]
   [CALENDAR calendarname]
   [PIVOT nn] [INVALIDDATES {Y|N}]
   [SKIPPEDDATES {Y|N}])]
[REPORT_OPTION (RPTERROR {T|F} [MAXTBLERR nnn]
   [MAXRUNERR nnn] [RPTSUMMARY{T|F}]
   [RPTINVALID{T|F}]
   [RPTSKIPPED {T|F}])]
[CURRENCY_ OPTION (DEFAULT currencytablename
   [GLOBAL currencytablename] [FROM currencytype]
   [TO currencytype] [TRIANG {T|F}])]
TBL (tblname DELETEROWS {Y|N} UPDINS {Y|N}
   STATUS {O|D|N|T})
[EMAILNOTIFY ({A|S|F} emailaddress)]
DEFAULT (DEFAULT //path//
   PATHMAP (//srcpath//  //destpath//) ) ;
```

Keywords

The keywords correspond to values that can be specified for an Insert Request. See the *Move User Manual* or the *Archive User Manual* .

UPIN *identifier.name*
The name of the Insert Request, specified in two parts (*identifier.name*) is required following the **CREATE UPIN** keyword.

DESC *//description//*
A description of the Insert Request, delimited by double slashes.

General

XF *xfilname*
The fully qualified name of the Extract File or Archive File used as input for the Insert Process.

CF *cfilname*
The fully qualified name of the Control File for the Insert Process.

Table Map Keywords

A Table Map parameter must be included. The Table Map can be Local or Named. If the keyword is **LOCALTM**, the Table Map is Local. The full definition of the Table Map follows, contained in parentheses. If the keyword is **TM**, the Table Map is Named and the name follows the keyword.

LOCALTM (*tmdef*)	The definition of a Local Table Map, enclosed in parentheses. For details on specifying Table Maps, see Chapter 10, "Table Maps," on page 229.
TM *tmname*	The two-part name (identifier.name) of the Named Table Map.

Keywords

DISCARDLIMIT *n*
The maximum number of rows (*n*) that can be discarded.

COMMITFREQ *n*
The frequency (every *n*th row) at which commits are issued.

FORCEEDITTM {Y|N}
(**Always View Table Map**) indicates whether the Table Map Editor opens automatically when processing an Insert Request.

Y The Table Map Editor opens automatically.

N The Table Map Editor does not open automatically.

LOCKTBLS {Y|N}
(**Lock Tables**) indicates whether tables are locked during the Insert.

Y Tables are locked.

N Tables are not locked.

UPDINS {U|I|M}
Indicates the type of Insert performed.

U Update/Insert

I Insert

M Mixed

Note: The specifications for particular tables selected for updating (using the Insert Table Specification dialog) are included in the TBL keywords.

DELETEROWS {A|M|N}
Indicates the type of delete performed as part of the Insert.

A All tables

M Selected tables

N Do not delete

Note: The tables selected for delete using the Insert Table Specification dialog are included in the TBL keywords.

DELCOMMIT {E|C}
Indicates the COMMIT frequency when rows are deleted during an Insert.

E After each table

C On completion

ALWAYSCALLCREATE {Y|N}
Indicates whether the Create dialog displays each time.

Y Create dialog displays.

N Create dialog displays when needed.

TRIGMODE {A|P|N}
TRIGMODE indicates whether triggers are disabled.

A Always

P Prompt for specific triggers

N Never

CONSMODE {A|P|N}
Indicates whether constraints are disabled

A Always

P Prompt for specific constraints.

N Never

SHOWCURRENCY {Y|N}
Indicates whether the **Currency** tab displays.

Y Currency tab displays.

N Currency tab does not display.

SHOWAGE {Y|N}
Indicates whether the **Aging** tabs display.

Y Age Function and Global Aging tabs display.

N Age Function and Global Aging tabs do not display.

PROCESS_ FILEATTACH {Y|N}
Indicates if file attachments in the Archive or Extract File are processed.

Age Function

If options for aging are specified, they follow the FUNCTION_AGING keyword in parentheses.

FUNCTION_AGING
Indicates date aging is used for the Insert.

AGETYPE *type*
The function type specified as one of the following:

NONE	SPECIFICDATE
INCREMENTAL	TARGETDATES
SPECIFICYEAR	RULEBASED

YEARS *nn*
The positive or negative numeric value (*nn*) to age the date by years.

MONTHS *nn*
The positive or negative numeric value (*nn*) to age the date by months.

WEEKS *nn*
The positive or negative numeric value *(nn)* to age the date by weeks.

DAYS *nn*
The positive or negative numeric value *(nn)* to age the date by days.

SPECIFICYEAR *nnnn*
An explicit year (*nnnn*) to which Insert should age.

SPECIFICDATE *mm/dd/yyyy*
The date (*mm/dd/yyyy*) to which Insert should age.

TARGETSTART *mm/dd/yyyy*
The date (*mm/dd/yyyy*) Insert should use as the starting date when calculating the amount to age using relative aging.

TARGETEND *mm/dd/yyyy*
The date (*mm/dd/yyyy*) Insert should use as the target date when calculating the amount to age using relative aging.

MULTIPLE *nn*
The number of times (*nn*) Insert should apply a rule to determine the aging amount.

RULE *rulename*
The name of the rule (*rulename*) being used.

CALENDAR *calendarname*
The name of the calendar (*calendarname*) being used.

PIVOT *nn*
The pivot year (*nn*).

INVALIDDATES {Y|N}
Indicates whether to insert rows with dates that are invalid to use.

SKIPPEDDATES {Y|N}
Indicates whether to insert rows with dates that should be skipped.

Global Aging

If options for function aging are specified, they follow the GLOBAL_AGING keyword in parentheses.

GLOBAL_AGING
Indicates global date aging is used for the Insert.

AGETYPE *type*
The function type specified as one of the following:

NONE	SPECIFICDATE
INCREMENTAL	TARGETDATES
SPECIFICYEAR	RULEBASED

YEARS *nn*
The positive or negative numeric value (*nn*) to age the date by years.

MONTHS *nn*
The positive or negative numeric value (*nn*) to age the date by months.

WEEKS *nn*
The positive or negative numeric value (*nn*) to age the date by weeks.

DAYS *nn*
The positive or negative numeric value (*nn*) to age the date by days.

SPECIFICYEAR *nnnn*
An explicit year (*nnnn*) to which Insert should age.

SPECIFICDATE *mm/dd/yyyy*
The date (*mm/dd/yyyy*) to which Insert should age.

TARGETSTART *mm/dd/yyyy*
The date (*mm/dd/yyyy*) Insert should use as the starting date when calculating the amount to age using relative aging.

TARGETEND *mm/dd/yyyy*
The date (*mm/dd/yyyy*) Insert should use as the target date when calculating the amount to age using relative aging.

MULTIPLE *nn*
The number of times (*nn*) Insert should use a rule to determine the aging amount.

RULE *rulename*
The name of the rule (*rulename*) being used.

CALENDAR *calendarname*
The name of the calendar (*calendarname*) being used.

PIVOT *nn*
The pivot year (*nn*).

INVALIDDATES {Y|N}
Indicates whether to Insert rows that have dates that are invalid to use.

SKIPPEDDATES {Y|N}
Indicates whether to Insert rows that have dates that should be skipped.

Report Options

If Report options are specified, they follow the REPORT_OPTION keyword in parentheses.

REPORT_OPTION
Indicates reporting options are used for the Insert.

RPTERROR {T|F}
Indicates whether to report errors
(True or False).

MAXTBLERR *nnn*
The maximum number *(nnn)* of errors per table to report.

MAXRUNERR *nnn*
The maximum number *(nnn)* of errors per run to report.

RPTSUMMARY {T|F}
Indicates whether to report the Aging summary (True or False).

RPTINVALID {T|F}
Indicates whether to report invalid dates (True or False).

RPTSKIPPED {T|F}
Indicates whether to report skipped dates (True or False).

Currency

If options for currency conversion are specified, they follow the CURRENCY_OPTION keyword in parentheses.

CURRENCY_OPTION
Indicates options for currency conversion are used for the Insert.

DEFAULT *currencytablename*
The name of the default Currency Table (*currencytablename*) being used.

GLOBAL *currencytablename*
The name of the global Currency Table (*currencytablename*) being used.

FROM *currencytype*
The three character code for the source currency type (*currencytype*).

TO *currencytype*
The three character code for the target currency type (*currencytype*).

TRIANG {T|F}
Indicates whether to convert currency via the euro dollar (True or False).

Table Keywords

There is a TBL entry for every table in the Table Map. The set of keywords for each table is enclosed in parentheses following the keyword "TBL." The keywords correspond to values specified both on the **General** tab of the Insert Request Editor and on the Insert Table Specification dialog.

tblname
The name of the table is required.

DELETEROWS {Y | N}
Indicates whether the rows in the table should be deleted prior to the insertion of new rows.

Y Rows should be deleted prior to insertion.

N Rows should not be deleted prior to insertion

UPDINS {Y | N}
Indicates whether the rows in the table should be updated and inserted or inserted without updating.

Y Update and insert.

N Insert only, do not update.

STATUS {O | D | N | T}
Indicates the Table Map status for the table.

O The table exists in the database.

D A new table using default settings.

N A new table for which options must be specified (because of the specifications for Mixed Insert or Selective Delete).

T A table that is not used, because it is not in the Table Map.

Notify

This parameter relates to the ability to send notification via email about the outcome of the process.

EMAILNOTIFY {A | S | F} *emailaddress*
Indicates the process outcome under which to send notification to the corresponding email address.

A Always send notification.

S Send notification when process succeeds.

F Send notification when process fails.

File Attachments

The **FILE_ATTACH** keyword relates to the Directory Map paths.

DEFAULT (//*path***//)**
The default destination path.

PATHMAP (//*srcpath***//***destpath***//)**
The source and destination paths. Use this keyword for each source and destination pair.

Load Requests

This syntax is used for each exported Load Request definition:

```
CREATE LOAD identifier.name
   [DESC //description//]
   XF xfilname CF cfilname {LOCALTM (tmdef) | TM tmname}
   FORCEEDITTM {Y|N} STOPONERROR {Y|N}
   STOPONFIRST CONVERTERROR {Y|N} MODE {S|P}
   ALWAYSCALL CREATE {Y|N} SHOWCURRENCY {Y|N}
   SHOWAGE {Y|N} PROCESS_ FILEATTACH {F|C}
   [DB2CS ( TYPE {I|R} LOAD {Y|N} DELOK {Y|N}
      DELFAIL {Y|N}
      EXCPFAIL {Y|N} EXCPCNST {Y|N} [EXCPCIID cid]
      [LOCAL path | REMOTE path]
      COPY {I IMAGE imagepath | A SESS numsession | N}
```

```
    [STATS {S|O|I|D}] [COMMITFREQ n]
    [DISCARDLIMIT n] )]
  [ORACLE (TYPE{I|R|A|T} LOAD{Y|N}
    DELOK{Y|N} DELFAIL{Y|N}
    DSCFILE {Y|N} [USEDIRECTPATH {Y|N} ]
    STARTVIO {Y|N} DISCARDLIMIT n
 [LOCAL path | REMOTE path]
    [COMMITFREQ n] TRIGMODE {A|P|N}
     CONSMODE {A|P|N} )]
  [SYBASE ( TYPE {I|R} LOAD {Y|N} DELOK {Y|N}
    DELFAIL {Y|N}
    DSCFILE {Y|N} DISCARDLIMIT n
    [LOCAL path | REMOTE path]
    TRIGMODE {A|P|N} CONSMODE {A|P|N} )]
    [PARTS (table_partition_mapping)]
  [SQLSERVER (TYPE{I|R} LOAD{Y|N} DELOK{Y|N}
    DELFAIL{Y|N}
    DSCFILE{Y|N}[DISCARDLIMIT n]
    [LOCAL path | REMOTE path] TRIGMODE {A|P|N}
    CONSMODE {A|P|N} WINAUTH {Y|N} )]
  [INFORMIX (TYPE{I|R} LOAD{Y|N} DELOK{Y|N}
    DELFAIL{Y|N}
    STARTVIO {Y|N} DELETEVIO {Y|N} DSCFILE {Y|N}
    [DISCARDLIMIT n]
    [LOCAL path | REMOTE path] TRIGMODE {A|P|N}
    CONSMODE {A|P|N} )]
  [FUNCTION_AGING (AGETYPE {NONE | INCREMENTAL |
    SPECIFICYEAR | SPECIFICDATE | TARGETDATES |
    RULEBASED} [YEARS nn] [MONTHS nn]
    [WEEKS nn] [DAYS nn] [SPECIFICYEAR nnnn]
    [SPECIFICDATE mm/dd/yyyy]
    [TARGETSTART mm/dd/yyyy TARGETEND mm/dd/yyyy]
    [MULTIPLE nn] [RULE rulename]
    [CALENDAR calendarname]
    [PIVOT nn] [INVALIDDATES {Y|N}]
    [SKIPPEDDATES {Y|N}])]
  [GLOBAL_AGING (AGETYPE {NONE | INCREMENTAL |
    SPECIFICYEAR | SPECIFICDATE | TARGETDATES |
    RULEBASED} [YEARS nn] [MONTHS nn]
    [WEEKS nn] [DAYS nn] [SPECIFICYEAR yyyy]
    [SPECIFICDATE mm/dd/yyyy]
    [TARGETSTART mm/dd/yyyy TARGETEND mm/dd/yyyy]
    [MULTIPLE nn] [RULE rulename]
    [CALENDAR calendarname]
    [PIVOT nn] [INVALIDDATES {Y|N}]
    [SKIPPEDDATES {Y|N}])];
  [REPORT_OPTION (RPTERROR {T|F} [MAXTBLERR nnn]
    [MAXRUNERR nnn] [RPTSUMMARY{T|F}]
    [RPTINVALID{T|F}]
    [RPTSKIPPED {T|F}])]
  [CURRENCY_ OPTION (DEFAULT currencytablename
    [GLOBAL currencytablename] [FROM currencytype]
    [TO currencytype] [TRIANG {T|F}])]
  [EMAILNOTIFY ({A|S|F} emailaddress)];
```

Keywords

The keywords correspond to values that can be specified for a Load Request. For additional information, see the *Move User Manual* or *Archive User Manual* .

LOAD *identifier.name*

The name of the Load Request, specified in two parts (*identifier.name*) is required following the **CREATE LOAD** keyword.

DESC *//description//*

A description of the Load Request, delimited by double slashes.

Editor Options

XF *xfilname*
The fully qualified name of the Extract File or Archive File used as input for the Load Process.

CF *cfilname*
The fully qualified name of the Control File for the Load Process.

Table Map Keywords

A Table Map parameter must be included. The Table Map can be Local or Named. If the keyword is **LOCALTM**, the full definition of the Table Map follows in parentheses. If the keyword is **TM**, the name follows.

LOCALTM (*tmdef***)**
A full definition of the Local Table Map being used, contained in parentheses, follows this keyword. The syntax and parameters for the definition of a Local Table Map are identical to those for a standalone Table Map.

TM *tmname*
The full two-part name of the Named Table Map being used for the Load Request.

General

FORCEEDITTM {Y | N}
(**Always View Table Map**) indicates whether the Table Map Editor opens automatically when processing a Load Request.

Y The Table Map Editor opens automatically.

N The Table Map Editor does not open automatically.

STOPONERROR {Y | N}
This keyword is only valid if **MODE** is set to **In Sequence**. Indicates whether processing continues in another DBMS loader after an

Y Processing halts.

N Processing continues.

STOPONFIRST CONVERTERROR {Y | N}
This keyword is only valid if **MODE** is set to **In Sequence**. Indicates whether processing continues if an error occurs in the Convert Process.

Y Processing halts.

N Processing continues.

MODE {S | P}
Indicates the mode for processing in multiple databases.

S In Sequence

P In Parallel

ALWAYSCALL CREATE {Y | N}
Indicates whether the Create dialog displays each time.

Y Create dialog displays.

N Create dialog displays when needed.

SHOWCURRENCY {Y | N}
Indicates whether the **Currency** tab displays.

Y Currency tab displays.

N Currency tab does not display.

SHOWAGE {Y|N}
Indicates whether the **Aging** tabs display.

Y Age Function and Global Aging tabs display.

N Age Function and Global Aging tabs do not display.

PROCESS_FILEATTACH {F|C}
If the source file contains file attachment pseudocolumns, indicate how the Load Process should proceed.

F Fail the process.

C Process pseudocolumns as normal table columns. If matching columns do not exist in the table, the pseudocolumns are ignored.

DB Alias - DB2CS

DB2CS keywords correspond to values that can be specified for a Load Request using the **DB Alias** tab of the Load Request Editor.

TYPE {I|R}
Indicates the type of Load that is performed.

I Insert

R Replace

LOAD {Y|N}
Indicates whether the Load is processed, or only prepared for processing.

Y The load is performed.

N The load is not performed.

DELOK {Y|N}
Indicates whether the .sql, .ixf, and .msg files are deleted when the Load is successful.

Y The files are deleted.

N The files are not deleted.

DELFAIL {Y|N}
Indicates whether the .sql, .ixf, and .msg files are deleted when the Load is not successful.

Y The files are deleted.

N The files are not deleted.

EXCPFAIL {Y|N}
Indicates whether rows that violate unique index or primary key rules are stored in the exception tables.

Y Rows in violation are stored in the exception tables.

N Rows in violation are not stored in the exception tables.

EXCPCNST {Y|N}
Indicates whether rows that violate referential integrity or table check constraints are stored in the exception tables.

Y Rows in violation are stored in the exception tables.

N Rows in violation are not stored in the exception tables.

EXCPCIID *cid*
The **Creator ID** for creating the exception tables.

LOCAL *path*
Specifies the workstation path for temporary loader files.

REMOTE *path*
Specifies the server path for temporary loader files.

COMMITFREQ *n*
Indicates the frequency (every *n*th row) at which commits are issued.

DISCARDLIMIT *n*
Indicates the maximum number of rows (*n*) that can be discarded.

Copy Keywords

These keywords reflect values selected on the Select Copy Option dialog.

COPY {I | A | N}
Indicates which **Copy** option is selected:

I Copy image to a directory (**IMAGE** must follow)

A Copy image using ADSM (**SESS** must follow)

N Do not copy

IMAGE *imagepath*
(**Path Name**) indicates the path and directory (*imagepath*) for the image copy.

SESS *numsession*
(**I/O Sessions**) indicates the number of I/O sessions (*numsession*) to be used with ADSM.

Statistics Keywords

These keywords reflect values selected on the Select Statistics dialog.

STATS {S | O | I | D}
Indicates which **Select Statistics** options are selected:

S **Table and distribution** is selected.

O **Indexes only** is selected.

I **Table and Indexes** is selected.

D **Extended stats for Indexes** is selected

DB Alias - ORACLE

Oracle keywords correspond to values that can be specified for a Load Request using the **DB Alias** tab of the Load Request Editor.

TYPE {I | R | A | T}
Indicates the type of Load that is performed.

I Insert

R Replace

A Append

T Truncate

LOAD {Y | N}
Indicates whether the Load is processed, or only prepared for processing.

Y The load is performed.

N The load is not performed.

DELOK {Y | N}
Indicates whether the .sql, .ixf, and .msg files are deleted when the Load is successful.

Y The files are deleted.

N The files are not deleted.

DELFAIL {Y | N}
Indicates whether the .sql, .ixf, and .msg files are deleted when the Load is not successful.

Y The files are deleted.

N The files are not deleted.

DSCFILE {Y | N}
Indicates whether discarded rows are written to a Discard File.

Y The rows are written.

N The rows are not written.

USEDIRECTPATH {Y | N}
Indicates the path as either Direct or Conventional.

Y The path is direct. Select this option when you want to load and index a large volume of data quickly.

N The path is conventional.

STARTVIO {Y | N}
Indicates whether violation tables are created.

Y The violation tables are created.

N The violation tables are not created.

DISCARDLIMIT *n*
Indicates the maximum number of rows (*n*) that can be discarded.

LOCAL *path*
Specifies the workstation path for temporary loader files.

REMOTE *path*
Specifies the server path for temporary loader files.

COMMITFREQ *n*
Indicates the frequency (every *n*th row) at which commits are issued.

TRIGMODE {A | P | N}
Indicates whether triggers are disabled.

A Always

P Prompt for specific triggers

N Never

CONSMODE {A | P | N}
Indicates whether constraints are disabled.

A Always

P Prompt for specific constraints

N Never

DB Alias - SYBASE

Sybase ASE keywords correspond to values that can be specified for a Load Request using the **DB Alias** tab of the Load Request Editor.

TYPE {I|R}
Indicates the type of Load that is performed.

I Insert

R Replace

LOAD {Y|N}
Indicates whether the Load is processed, or only prepared for processing.

Y The load is performed.

N The load is not performed.

DELOK {Y|N}
Indicates whether the .sql, .ixf, and .msg files are deleted when the Load is successful.

Y The files are deleted.

N The files are not deleted.

DELFAIL {Y|N}
Indicates whether the .sql, .ixf, and .msg files are deleted when the Load is not successful.

Y The files are deleted.

N The files are not deleted.

DSCFILE {Y|N}
Indicates whether discarded rows are written to a Discard File.

Y The rows are written.

N The rows are not written.

DISCARDLIMIT *n*
Indicates the maximum number of rows (*n*) that can be discarded.

LOCAL *path*
Specifies the workstation path for temporary loader files.

REMOTE *path*
Specifies the server path for temporary loader files.

TRIGMODE {A|P|N}
Indicates whether triggers are disabled.

A Always

P Prompt for specific triggers

N Never

CONSMODE {A|P|N}
Indicates whether constraints are disabled.

A Always

P Prompt for specific constraints

N Never

PARTS *identifier.tablename = partition name*
If you want to store extracted data in a specific partition, you must pass the appropriate partition name when you run the Load process. You can specify one partition name only for each table, such as: `qadba.my_publishers = west`. In this example, `qadba.my_publishers` identifies the destination table within the table map, and `west` is the Sybase partition name for that table.

DB Alias – SQL Server

SQL Server keywords correspond to values that can be specified for a Load Request using the **DB Alias** tab of the Load Request Editor.

TYPE {I|R}
Indicates the type of Load that is performed.

I Insert

R Replace

LOAD {Y|N}
Indicates whether the Load is processed, or only prepared for processing.

Y The load is performed.

N The load is not performed.

DELOK {Y|N}
Indicates whether the .sql, .ixf, and .msg files are deleted when the Load is successful.

Y The files are deleted.

N The files are not deleted.

DELFAIL {Y|N}
Indicates whether the .sql, .ixf, and .msg files are deleted when the Load is not successful.

Y The files are deleted.

N The files are not deleted.

DSCFILE {Y|N}
Indicates whether discarded rows are written to a Discard File.

Y The rows are written.

N The rows are not written.

DISCARDLIMIT *n*
Indicates the maximum number of rows (*n*) that can be discarded.

LOCAL *path*
Specifies the workstation path for temporary loader files.

REMOTE *path*
Specifies the server path for temporary loader files.

TRIGMODE {A|P|N}
Indicates whether triggers are disabled.

A Always

P Prompt for specific triggers

N Never

CONSMODE {A|P|N}
Indicates whether constraints are disabled.

A Always

P Prompt for specific constraints

N Never

WINAUTH {Y|N}
Indicates whether NT authentication is used.

Y NT authentication is used.

N NT authentication is not used.

DB Alias - Informix

Informix keywords correspond to values that can be specified for a Load Request using the **DB Alias** tab of the Load Request Editor.

TYPE {I|R}
Indicates the type of Load that is performed.

I Insert

R Replace

LOAD {Y|N}
Indicates whether the Load is processed, or only prepared for processing.

Y The load is performed.

N The load is not performed.

DELOK {Y|N}
Indicates whether the .sql, .ixf, and .msg files are deleted when the Load is successful.

Y The files are deleted.

N The files are not deleted.

DELFAIL {Y|N}
Indicates whether the .sql, .ixf, and .msg files are deleted when the Load is not successful.

Y The files are deleted.

N The files are not deleted.

STARTVIO {Y|N}
Indicates whether violation tables are created.

Y The violation tables are created.

N The violation tables are not created.

DELETEVIO {Y|N}
Indicates whether to delete all rows in existing violation tables before the Load begins.

Y Delete rows from violation tables.

N Do not delete rows from violation tables.

DSCFILE {Y|N}
Indicates whether discarded rows are written to a Discard File.

Y The rows are written.

N The rows are not written.

DISCARDLIMIT *n*
Indicates the maximum number of rows (*n*) that can be discarded.

LOCAL *path*
Specifies the workstation path for temporary loader files.

REMOTE *path*
Specifies the server path for temporary loader files.

TRIGMODE {A | P | N}
Indicates whether triggers are disabled.

A Always

P Prompt for specific triggers

N Never

CONSMODE {A | P | N}
Indicates whether constraints are disabled.

A Always

P Prompt for specific constraints

N Never

Age Function

If options for function aging are specified, they follow the FUNCTION_AGING keyword in parentheses.

FUNCTION_AGING
Indicates function date aging is used for the Load.

AGETYPE *type*
The function type specified as one of the following:

NONE	SPECIFICDATE
INCREMENTAL	TARGETDATES
SPECIFICYEAR	RULEBASED

YEARS *nn*
The positive or negative numeric value (*nn*) to age the date by years.

MONTHS *nn*
The positive or negative numeric value (*nn*) to age the date by months.

WEEKS *nn*
The positive or negative numeric value (*nn*) to age the date by weeks.

DAYS *nn*
The positive or negative numeric value (*nn*) to age the date by days.

SPECIFICYEAR *yyyy*
An explicit year (*yyyy*) to which Load should age.

SPECIFICDATE *mm/dd/yyyy*
The date (*mm/dd/yyyy*) to which Load should age.

TARGETSTART *mm/dd/yyyy*
The date (*mm/dd/yyyy*) Load uses as the starting date when calculating the amount to age using relative aging.

TARGETEND *mm/dd/yyyy*
The date (*mm/dd/yyyy*) Load should use as the target date when calculating the amount to age using relative aging.

MULTIPLE *nn*
The number of times (*nn*) to apply a rule to determine the aging amount.

RULE *rulename*
The name of the rule (*rulename*) being used.

CALENDAR *calendarname*
The name of the calendar (*calendarname*) being used.

PIVOT *nn*
The pivot year (*nn*).

INVALIDDATES {Y|N}
Indicates whether rows containing dates that are invalid are used.

SKIPPEDDATES {Y|N}
Indicates whether rows containing dates that should be skipped are used.

Global Aging

If options for function aging are specified, they follow the GLOBAL_AGING keyword in parentheses.

GLOBAL_AGING
Indicates global date aging is used for the Load.

AGETYPE *type*
The function type specified as one of the following:

NONE	SPECIFICDATE
INCREMENTAL	TARGETDATES
SPECIFICYEAR	RULEBASED

YEARS *nn*
The positive or negative numeric value (*nn*) to age the date by years.

MONTHS *nn*
The positive or negative numeric value (*nn*) to age the date by months.

WEEKS *nn*
The positive or negative numeric value (*nn*) to age the date by weeks.

DAYS *nn*
The positive or negative numeric value (*nn*) to age the date by days.

SPECIFICYEAR *yyyy*
An explicit year (*yyyy*) to which Load should age.

SPECIFICDATE *mm/dd/yyyy*
The date (*mm/dd/yyyy*) to which Load should age.

TARGETSTART *mm/dd/yyyy*
The date (*mm/dd/yyyy*) Load should use as the starting date when calculating the amount to age using relative aging.

TARGETEND *mm/dd/yyyy*
The date (*mm/dd/yyyy*) Load should use as the target date when calculating the amount to age using relative aging.

MULTIPLE *nn*
The number of times (*nn*) to apply a rule to determine the aging amount.

RULE *rulename*
The name of the rule (*rulename*) being used.

CALENDAR *calendarname*
The name of the calendar (*calendarname*) being used.

PIVOT *nn*
The pivot year (*nn*).

INVALIDDATES {Y|N}
Indicates whether rows containing dates that are invalid are processed.

SKIPPEDDATES {Y|N}
Indicates whether rows containing dates that should be skipped are processed.

Report Options

If Report options are specified, they follow the REPORT_OPTION keyword in parentheses.

REPORT_OPTION
Indicates reporting options are used in the Load.

RPTERROR {T|F}
Indicates whether to report errors (True or False).

MAXTBLERR *nnn*
The maximum number (*nnn*) of errors per table to report.

MAXRUNERR *nnn*
The maximum number (*nnn*) of errors per run to report.

RPTSUMMARY {T|F}
Indicates whether to report the Aging summary (True or False).

RPTINVALID {T|F}
Indicates whether to report invalid dates (True or False).

RPTSKIPPED {T|F}
Indicates whether to report skipped dates (True or False).

Currency

If options for currency conversion are specified, they follow the CURRENCY_OPTION keyword in parentheses.

CURRENCY_OPTION
Indicates options for currency conversion are used for the Load.

DEFAULT *currencytablename*
The name of the default Currency Table (*currencytablename*) being used.

GLOBAL *currencytablename*
The name of the global Currency Table (*currencytablename*) being used.

FROM *currencytype*
The three character code for the source currency type (*currencytype*).

TO *currencytype*
The three character code for the target currency type (*currencytype*).

TRIANG {T|F}
Indicates whether to convert currency via the euro dollar (True or False).

Notify

This parameter relates to the ability to send notification via email about the outcome of the process.

EMAILNOTIFY {A|S|F} *emailaddress*

Indicates the process outcome under which to send notification to the corresponding email address.

A Always send notification.

S Send notification when process succeeds.

F Send notification when process fails.

Report Requests

Syntax used for exported Report Requests varies depending on the type of source file used for the report — Archive File or Compare File.

```
CREATE REPT identifier.name
   [DESC //description//]
   [TITLE //reporttitle//]
   LOCALFILE {Y|N}
   LOCALPRINTER {Y|N}
      [PRINTERSETUP (PRINTERNAME printername
      DRIVERNAME driver PORT port ORIENTATION {P|L})]
   REPORTFILENAME rtffilename
   REPORTFILEFORMAT {1|2} [SERVERNAME server]
   DEFTYPE {A|R}
   ANDFLAG {Y|N} DELIMITER {+ | : | ; | |}
   ARCCRIT (TYPE criteria VALUES value SORTASC {Y|N})
   SOURCEFILENAME sourcefilename
   SOURCEFILETYPE {CMP|AF}
   AUTOREPNEWTABLES  {Y|N}
   SHOWSUMMARY  {Y|N} SHOWDETAILS {Y|N}
   SHOWEMPTYTABLES {Y|N}
   COMPAREDETAILTYPE {C|A|D}
      SHOWEQUALROWS {Y|N}
      SHOWDIFFERENTROWS {Y|N}
      SHOWROWSONLYINS1 {Y|N}
      SHOWROWSONLYINS2 {Y|N}
      SHOWDUPLICATESFROMS1 {Y|N}
      SHOWDUPLICATESFROMS2 {Y|N}
      COMPTABLEDATA (tablename  tablename {Y|N})
   ARCHIVEDETAILTYPE {C|S}
      SHOWJOINS {Y|N}
         TABLE (tablename {Y|N}) |
         TABLE (NAME tablename NEWPAGEPERROW{Y|N}
         PREDOP {AND|OR} [COLUMN (NAME columnname
OP operator VALUE "value" )] [SQL //sqlwhereclause//]
         REL identifier.name
   ROWSPERTABLE n LINESPERPAGE n LINELENGTH n
   CHARCOLWIDTH n LINESBETWEENROWS n
   SPACEBETWEENCOLS n LINESBETWEENLEVELS n
   SUBORDTABLEINDENTSPACE n ADJUSTLONGLINE
{T|W}
   SHOWTABLEHEAD {Y|N}
   SHOWTABLEHEADSUBORD {Y|N}
   SHOWTABLEHEADREDUND {Y|N}
   [EMAILNOTIFY ({A|S|F} emailaddress)];
```

Keywords

The keywords correspond to values that can be specified for a Report Request. For additional information, see the *Archive* or *Compare* documentation.

REPT *identifier.name*

The name of the Report Request, specified in two parts (*identifier.name*) is required following the **CREATE REPT** keyword.

DESC *//description//*
A description of the Report Request, delimited by double slashes.

General

The keywords correspond to values that can be specified for a Report Request. For additional information, see the *Archive User Manual* or *Compare User Manual*.

TITLE *//description//*
The title of the report, referenced at the top of each page of the report, delimited by double slashes.

DEFTYPE {A|R}
Indicates the type of Report Request: A for Archive Directory or R for Source File.

LOCALFILE {Y|N}
Indicates whether report output is directed to a file.

LOCALPRINTER {Y|N}
Indicates whether report output is directed to a printer.

PRINTERSETUP
Follow this keyword with the set of keywords for the printer setup, enclosed in parentheses.

PRINTERNAME *printername*
Name of the local printer.

DRIVERNAME *drivername*
Name of the driver for the local printer.

PORT *port*
Name of the port for the local printer.

ORIENTATION {P|L}
Print orientation: P for portrait or L for landscape.

REPORTFILENAME *rtffilename*
The fully qualified name of a text file for the output of the report process.

REPORTFILEFORMAT {1|2}
Numerical value to indicate the report output format. A value of 1 indicates plain text; a value of 2 indicates Rich Text.

Source File - General

These keywords correspond to values that can be specified for Report Requests where the Report Type is File, regardless of the Source File type.

SERVERNAME *server*
Identifies the name of the server containing the source file, as applicable.

SOURCEFILENAME *sourcefilename*
The fully qualified name of the Compare File or Archive File used for the report process.

SOURCEFILETYPE {CMP|AF}
Indicates whether the source file is a Compare File (CMP) or Archive File (AF).

AUTOREPNEWTABLES {Y|N}
Indicates whether to automatically report on new tables.

Source File - Archive File

These keywords correspond to values that can be specified for Report Requests where the Report Type is File and the Source File is an Archive File.

ARCHIVEDETAILTYPE {C|S}
For an Archive File in a Report Request, C indicates columnar format, S indicates side label format.

SHOWJOINS {Y|N}
Indicates whether archived tables may be joined in the Report.

TABLE (*tablename* **{Y|N})**
This keyword precedes each fully qualified table name in the Archive File, and an indicator for whether the table is included in the report. If tables in the Report Request are joined, the table name is followed by additional keywords to indicate the relationship between the joined tables, selection criteria, and formatting options, as applicable.

NAME *tablename*
This keyword precedes each fully qualified table name in the Archive File, and an indicator for whether the table is included in the report. If tables in the Report Request are joined, the table name is followed by additional keywords to indicate the relationship between the joined tables, selection criteria, and formatting options, as applicable.

NEWPAGEPERROW {Y|N}
For joined tables, this keyword specifies whether a new page is used for each row and its related row(s).

PREDOP {AND|OR}
For joined tables with selection criteria specified, the predicate operator indicates whether to AND or OR the criteria.

[COLUMN (NAME columnname OP operator VALUE *"value"***)] [SQL** *//sqlwhereclause//***]**
For joined tables, this keyword precedes each column name for which criteria is specified. Follow the column name with additional keywords to indicate the applicable relational operator and value used, or the text of the SQL Where clause delimited by double slashes.

REL *identifier.name*
The fully qualified name of the relationship used to join a pair of tables.

Source File - Compare File

These keywords correspond to values that can be specified for Report Requests where the Report Type is File and the Source File is a Compare File.

SHOWSUMMARY {Y|N}
Indicates whether to summarize the report output.

SHOWDETAILS {Y|N}
Indicates whether report output is detailed.

SHOWEMPTYTABLES {Y|N}
Indicates whether headers for empty tables are included in the report output.

COMPAREDETAILTYPE {C|A|D}
For a Compare File in a Report Request, C indicates columnar format, A indicates side label format - all columns, and D indicates side label format - different columns.

SHOWEQUALROWS {Y|N}
Indicates whether to show rows in the report output that the Compare Process found to be equal.

SHOWDIFFERENTROWS {Y|N}
Indicates whether to show rows in the report output that the Compare Process found to be unequal.

SHOWROWSONLYINS1 {Y|N}
Indicates whether to show rows in the report output that the Compare Process found only in Source 1.

SHOWROWSONLYINS2 {Y|N}
Indicates whether to show rows in the report output that the Compare Process found only in Source 2.

SHOWDUPLICATESFROMS1 {Y|N}
Indicates whether to show rows in the report output from Source 1 that contain the same Match Key.

SHOWDUPLICATESFROMS2 {Y|N}
Indicates whether to show rows in the report output from Source 2 that contain the same Match Key.

COMPTABLEDATA *tablename tablename* **{Y|N}**
This keyword precedes each matched pair of fully qualified table names in the Compare File, and an indicator for whether the pair is included in the report.

Archive Directory

These keywords correspond to values that can be specified for Report Requests where the Report Type is Archive Directory.

ANDFLAG {Y|N}
Indicates whether criteria in the Report Request is joined (e.g., ANDed).

DELIMITER {+ | : | ; | |}
Indicates the delimiter used in the Report Request.

ARCCRIT
Indicates archive criteria are referenced in the Report Request. The criteria are contained in parentheses that follow this keyword.

TYPE *criteria*
Indicates the type of criteria used to find Archive Directory entries.

VALUES *value*
Indicates the value for the associated criteria.

SORTASC {Y|N}
Indicate whether the criteria is sorted in ascending order.

Details Formatting

These keywords control the appearance of the report.

ROWSPERTABLE *n*
Maximum number of rows for each table in the report.

LINESPERPAGE *n*
Maximum number of lines per page in the report.

LINELENGTH *n*
Maximum number of characters per line in the report.

CHARCOLWIDTH *n*
Maximum number of characters per column in the report.

LINESBETWEENROWS *n*
Number of blank lines inserted between each row in the report.

SPACEBETWEENCOLS *n*
Number of blank spaces inserted between columns in the report.

LINESBETWEENLEVELS *n*
Number of blank lines inserted between each level of joined tables.

SUBORDTABLEINDENTSPACE *n*
Number of blank spaces to indent rows from each subordinate joined table in the report.

ADJUSTLONGLINE {T|W}
When row data exceeds the line length limitation, T indicates truncate the row, W indicates wrap to the next line.

SHOWTABLEHEAD {Y|N}
Indicates whether column names are displayed with Start table data.

SHOWTABLEHEADSUBORD {Y|N}
Indicates whether column names are displayed for the first row of data from a joined table.

SHOWTABLEHEADREDUND {Y|N}
Indicates whether column names are displayed with every row of data reported from a joined table.

Notify

This parameter relates to the ability to send notification via email about the outcome of the process.

EMAILNOTIFY {A|S|F} *emailaddress*
Indicates the process outcome under which to send notification to the corresponding email address.

A Always send notification.

S Send notification when process succeeds.

F Send notification when process fails.

Restore Requests

This syntax is used for each exported Restore Request definition:

```
CREATE REST identifier.name
   [DESC //description//]
   ACTION {INSERT | LOAD}
   SELECTION_MODE {DATAMODEL |GROUP |DESC |DATE}
   DELETE_XFFILES {Y|N}  AUTO_GEN{Y|N}
   CONTINUE_ON_ERROR {Y|N}
   COMPRESSFILE {Y|N}
   FILE {archivefile [ARCHIVE_ID n] [SERVERNAME server]
      [XF subsetextractfilename] [AD_OVERRIDE //string//]
      [PARMS //string//]  [ROWLIM n]  SELCRIT_USE {N | G |L}
      TABLEOP {A|O}
      TABLE ( tablename PREDOP {A|O} [SQL // string //]
         [COLUMN (column OP operator PRED // string //)
         COLUMN (...) ] )
      TABLE (...))
   FILE (...)
   PARM ( {LOCALDEF { text for insert | text for load }
      | DEF id.name} [VALUE1 //string//] [VALUE2 //string//]}
   PARM  (...)
   [GLBLCRIT ( TABLEOP { A|O }
      TABLE ( tablename PREDOP { A|O } [SQL // string//]
```

```
        [COLUMN (column OP operator PRED // string //)
        COLUMN (...) ] )
    TABLE (...))
[EMAIL NOTIFY ({A|S|F} emailaddress)];
```

Keywords

The keywords correspond to values that can be specified for a Restore Request. For additional information, see the *Archive User Manual,* Chapter 8. Restore.

REST *identifier.name*
The name of the Restore Request, specified in two parts (*identifier.name*) is required following the **CREATE REST** keyword.

DESC *//description//*
A description of the Restore Request, delimited by double slashes.

ACTION {INSERT|LOAD}
The type of process to use for the Restore Request, Insert or Load.

SELECTION_MODE {DATAMODEL|GROUP| DATE|DESC}
Indicator for type of criteria used to match Archive Files to Restore with the appropriate Insert or Load Request.

DELETE_XFFILES {Y|N}
Indicator for deleting subset Extract Files, if any, after the Restore Process. When selection criteria are applied to Archive Files in a Restore Request, subset Extract Files are created. Y causes the subset Extract Files to be deleted at the successful conclusion of the Restore Process.

AUTO_GEN {Y|N}
Indicator for subset Extract File names, as applicable. If selection criteria are applied to Archive Files for the Restore Process, subset Extract Files are created. Y indicates the names are automatically generated. If N, the keyword XF is used to identify the subset Extract File name.

CONTINUE_ON_ERROR {Y|N}
Indicator for processing when errors occur.

COMPRESSFILE {Y|N}
Indicator for whether to compress the Subset Extract File after processing.

FILE *archivefile*
The fully qualified name of the Archive File used as input for the Restore Process.

ARCHIVE_ID *n*
The numerical Archive File identifier, which can be used instead of the file name.

SERVERNAME *server*
Name of the server on which the specified Archive File resides.

XF *subsetextractfile-name*
The name of the subset Extract File. If global or local selection criteria are specified, a subset Extract File is created.

AD_OVERRIDE //*string***//**
Indicates changes made to the Access Definition for the Archive File.

PARMS *//string//*
There is a **PARMS** entry only when a variable(s) has been defined for the specified Archive File in the Restore Request.

Each variable is listed in the following form:

(*name,"prompt string",default value*)

- Each variable is enclosed in parentheses and is separated by commas.

- Commas are used between variable names, default values, and prompt strings.

ROWLIM *n*
A limit for the number of rows (*n*) that can be restored from the Archive File. If the Restore Request does not include a Row Limit, the keyword is not included in the definition.

SELCRIT_USE {N | G | L}
Indicator for type of selection criteria to use.

N None

G Global

L Local

TABLEOP {A | O}
Criteria connector for when multiple local selection criteria or SQL statements are specified.

A And

O Or

TABLE *tablename*
The fully qualified name of the table for which local selection criteria is defined.

PREDOP {A | O}
Criteria connector for when multiple selection criteria or multiple SQL statements are specified.

A And

O Or

SQL ***//string//***
An SQL WHERE clause for the specified table.

COLUMN *column*
The column name for which local selection criteria applies.

OP *operator*
Indicates the relational operator used for the local selection criteria.

PRED ***//string//***
Value for the local selection criteria.

PARM Indicates each of the Insert or Load Requests contained in the Restore Request.

LOCALDEF *text for insert | text for load*
Parameters for each local Insert or Load Request follow this keyword, as applicable.

DEF *id.name*
The name of the Insert or Load Request, specified in two parts (*identifier.name*).

VALUE1 ***//string//***
The value used as criteria to match the Insert or Load Request with the Archive File to be restored. Provide two values if specifying a date range.

GLBLCRIT
Global selection criteria for a specified table applied to one or more Archive Files in a Restore Process.

TABLEOP {A | O}
Criteria connector for when multiple global table selection criteria or SQL statements are specified.

A And

O Or

TABLE *tablename*
The fully-qualified name of the table for which global selection criteria is defined.

PREDOP {A | O}
Criteria connector for when multiple global column selection criteria or SQL statements are specified.

A And

O Or

SQL //*string***//**
An SQL WHERE clause for the specified table.

COLUMN *column*
The column name for which global selection criteria applies.

OP *operator*
Indicates the relational operator used for the global selection criteria.

PRED //*string***//**
Value for the global selection criteria.

Notify

This parameter relates to the ability to send notification via email about the outcome of the process.

EMAIL NOTIFY {A | S | F} *emailaddress*
Indicates the process outcome under which to send notification to the corresponding email address.

A Always send notification.

S Send notification when process succeeds.

F Send notification when process fails.

Chapter 17. Create

Use the Create Utility to create objects on the basis of definitions in a source Extract or Archive File.

For example, if authorized, you can use the utility to create:

- A database that mirrors the production database as it existed at the time data was archived so that production or other applications (e.g., report applications) can be run against it.
- A test database of tables, indexes, primary keys, foreign keys, etc., that are identical to those for a production database.
- A temporary database for previously archived information.

Using the Create Utility, you identify the destination and select objects from the source file to be created or dropped at the destination. The Create Utility then generates the necessary SQL, lets you edit the SQL before it is executed, and displays information returned from the DBMS after execution.

From the Main Window

To create objects at a destination without loading data, open the Create Utility from the main window.

During a Process

To create the objects and load data, use an Insert, Restore, or Load Process, as appropriate. If destination tables and other objects (such as primary keys and relationships) do not exist in the destination, you are prompted during the process to create them. These processes do not automatically create objects for destination tables or objects that already exist.

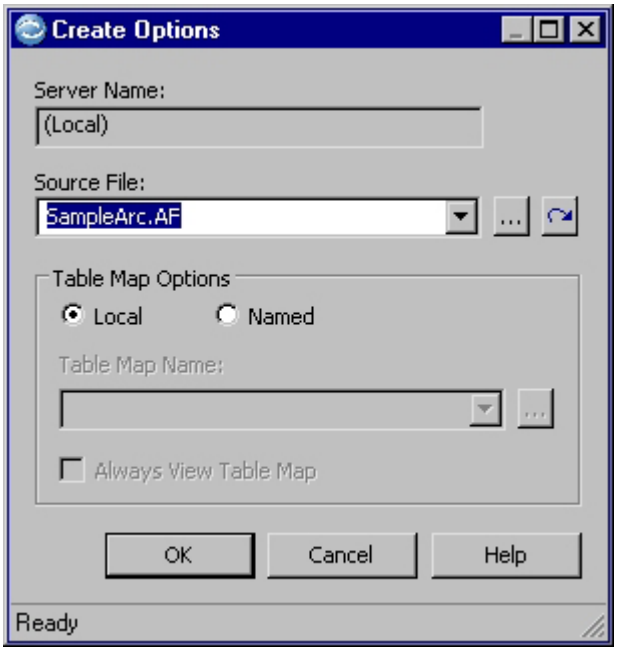
Contents

This section explains how to use the Create Utility, including how to:

- Specify options for the Create Process.
- Review a Table Map to map the source definitions to the destination.
- Browse and review the SQL.
- Create and drop objects.

Open the Create Utility

In the main window, select **Create** from the **Utilities** menu to display the Create Options dialog.



Server Name

If the optional Optim Server is available on your network, you can click the down arrow and select the Server on which the file resides, or **Local** (to access the source file on the workstation).

Source File

The name of the source Extract or Archive File containing the database object definitions to create.

- To select from names of recently created files, click the down arrow.
- To select a file from your system directories, click the browse button.
- To select the Extract or Archive File you last created, click the retrieve button.

Note: If you select the name of an Archive File stored on tape or Centera from the drop-down list, the file is recalled to the Alternate File Path specified in the Storage Profile.

Table Map Options

Local Use a temporary Table Map, retained for the current Create Process only. A local Table Map is discarded when you close the Create dialog.

Named
Use a Table Map saved in the Optim Directory. You must provide a Table Map name.

Table Map Name
The name of a new or existing Table Map to be used for the Create Process. Click the down arrow to select from a list of recently referenced Table Maps, or click the browse button to open a dialog where you can select a Table Map.

Always View Table Map
Option to review or modify the Table Map specifications before you create the objects. Select this check box to review the Table Map each time you open the Create Utility.

If you clear this check box, the Table Map Editor opens automatically if you specify a local Table Map or a named Table Map that does not exist or that does not reference all tables in the Extract or Archive File.

Using Table Maps

A Table Map matches source tables with tables to be created at the destination (i.e., the target). You can exclude tables from processing or create objects with names different from those of source objects.

Edit a Table Map

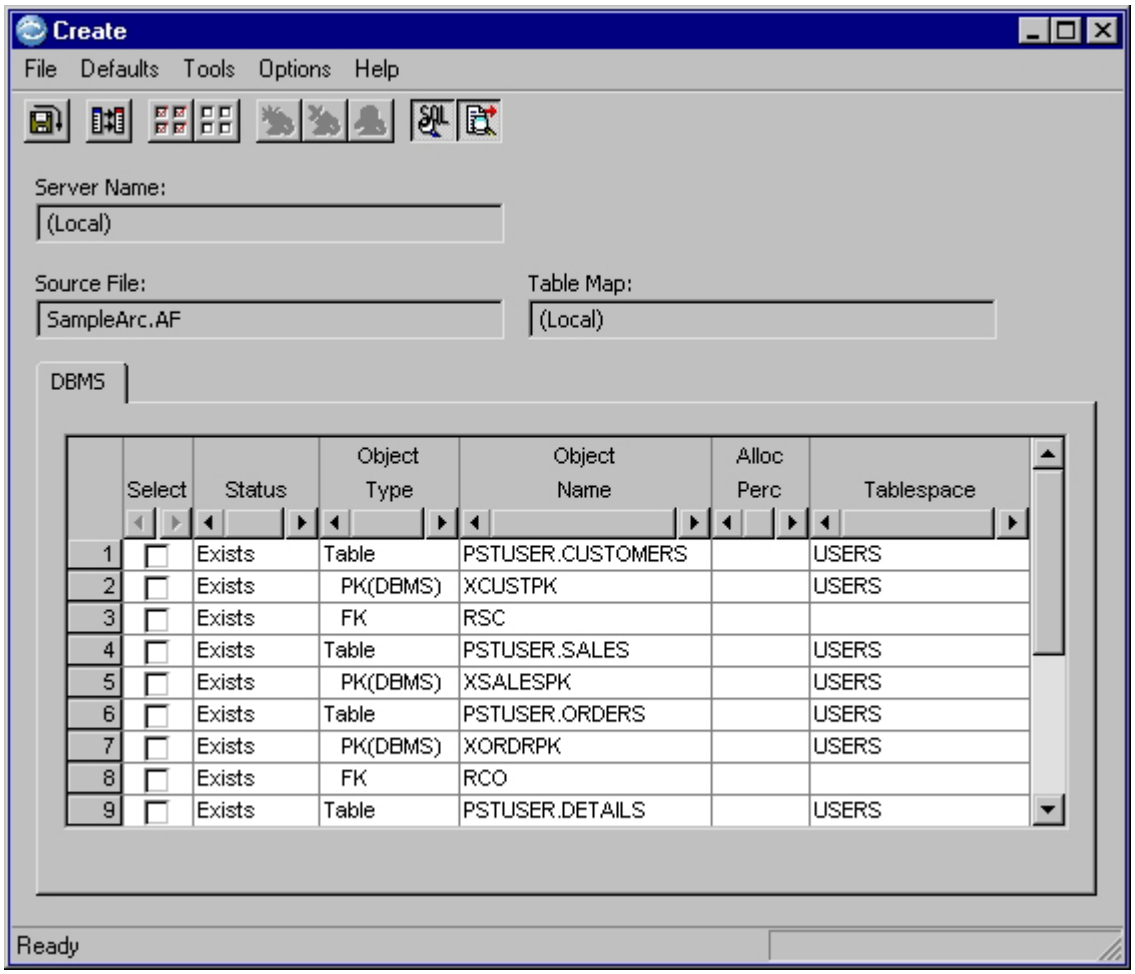
If needed, you can edit the Table Map. For example, if changes to database tables and other database objects have occurred since the Table Map was last used, the Table Map specifications may not be valid.

If the Table Map Editor is not displayed automatically, select **Edit Table Map** from the **Tools** menu on the Create dialog to open the Table Map Editor. For details on how to create, edit, or merge Table Maps, see "Open the Table Map Editor" on page 229.

After reviewing or modifying the Table Map, select **Update and Return** from the **File** menu to redisplay the Create dialog.

Using the Create Utility

Close the Table Map Editor or select **Update and Return** from the **File** menu to display the Create dialog.



Server Name

The location of the source file.

Source File

The name and path for the source file.

Table Map

The name of the Table Map or (Local). To create objects using a different source file or Table Map, select **Respecify Options** from the **Tools** menu to redisplay the Create Options dialog. You can then provide a different source file or Table Map name.

To change the names of the tables or to process tables previously removed from the Table Map, open the Table Map Editor by selecting **Edit Table Map** from the **Tools** menu.

Tabs

The Create dialog provides a tab for each DB Alias referenced by the Table Map. The objects are listed on the appropriate tab. Each table name, followed by the names of associated objects, is listed in the order referenced in the Table Map. The utility identifies objects that already exist at the destination and assigns an appropriate status.

Object List

Select Include the object in Create or Drop processing or display in the Browse SQL dialog. An object that does not exist at the destination is selected automatically. Clear the check box to exclude an object, as required.

To select or unselect all objects listed on the tab, select **Select All Objects** or **Deselect All Objects** from the **Tools** menu or the shortcut menu.

Status The status of the object, as follows:

blank Object does not exist in the destination database and can be created.

Conflict
Source and destination conflict. A message describes the conflict, which must be resolved before the object can be created. In some cases, the object exists but does not match the attributes in the source file (for example, a column precision mismatch).

The method used to resolve a conflict varies by type of conflict. For example, if the name of an object matches that of an object in the target database, you can drop the object in the target or rename the object to be created. However, a column mismatch for a primary key or relationship may require research or analysis to determine the appropriate action.

Created
Object was created earlier and can be dropped.

Exists Object existed prior to opening the Create dialog and can be dropped.

Invalid Name
Object name in the Create dialog was modified and the modified value is invalid.

Not Needed
Object is created implicitly.

Pending
The object can be created after a prerequisite object has been created. For example, a foreign key (FK) is not created until both tables in the relationship are created.

Unsupported
Object not supported in the database.

To update the status of listed objects, select Refresh Status from the Tools menu. For example, if you change the setting for DB2 OS/ 390 Current Rules in Personal Options while the Create dialog is displayed, you can Refresh Status to reflect the new setting. For more information, see "Create Tab" on page 440.

Object Type
The type of object definition as:

Alias	PK (DBMS)
Aux Table	PK (Optim)
Default	Private Synonym
Ext Rel	Rule
FK	Rel
Function	Synonym
Index	Sequence
Package	Trigger
Package Body	Table
Procedure	LOB Tablespace
User Defined Type (Datatype)	View

Note: DDL objects are indented beneath each owning, parent, or superior object.

Object Name
Name of the object. You can modify the names of most objects except those mapped in the Table Map (e.g., tables). To modify names of tables and certain other objects (e.g., Defaults, Rules, UDTs), you must open the Table Map Editor.

Alloc Perc
Percentage by which storage-related SQL parameters for the source object are adjusted (for Oracle and DB2 MVS). The allowable range is 0 to 999. Specify a value to override defaults for an object or zero (0) to use the target system default. (See "Defaults Dialog" on page 418 for additional information regarding defaults.)

Tablespace
Name of the Tablespace, Segment, Filegroup, or DBSpace for certain objects. The utility populates Tablespace when the Create dialog opens. If an object exists at the target, the tablespace name is displayed and cannot be modified. If the object does not exist, the initial value is:

- The name of the tablespace for the source table, if it matches a target tablespace.
- The target system default, identified as <DEFAULT>, unless overridden by a tablespace name provided on:
  - The **Table Defaults** tab of the DB Alias editor (see "Table Defaults" on page 197), unless overridden by a Personal Options setting.
  - The **Create** tab of **Personal** options (see "Create Tab" on page 440), unless overridden by a Defaults dialog setting.
  - The Defaults dialog accessed from the **Defaults** menu of the Create dialog. See "Defaults Dialog" on page 418 for details.

To change the setting, click a **Tablespace** grid cell and select:

<DEFAULT>
Use the target system default.

<SOURCE>
Use the source.

name Use the designated tablespace.

Note: If you change a Tablespace value to <SOURCE>, and the source tablespace does not exist at the target, the Create Process will fail.

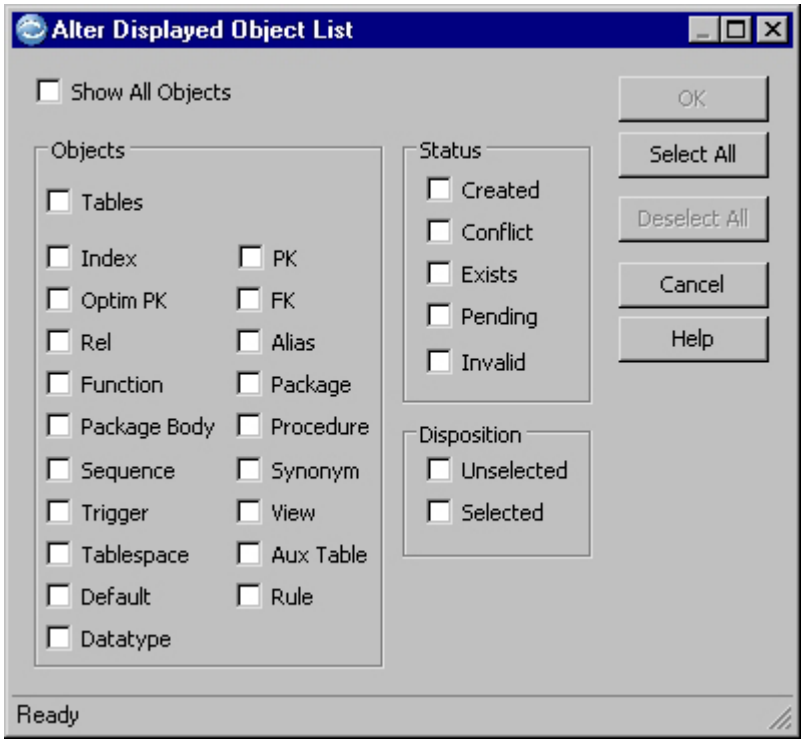
Tablespace/ Buffer pool
For DB2 MVS, the label is **Tablespace/Buffer pool**, rather than **Tablespace**.

- If the object is a table, this value is the tablespace name. If the table exists, the tablespace name is displayed and cannot be modified. If the table does not exist, the initial value is provided as described for **Tablespace** and can be modified in the same way.
- If the object is an index, this value indicates the buffer pool for the index. If the index exists, the buffer pool name is displayed and cannot be modified. If the index does not exist, the initial value is provided as described for **Tablespace** and can be modified by clicking the grid and selecting from the list.

Managing the Display

You can manage a large list of objects on the Create dialog according to object type, status or a combination of criteria. From the **Tools** menu, select **Alter Display**.

(Table definitions are always listed, regardless of selections on this dialog.)



Show All Objects

Clear the check box to select criteria for objects listed on the Create dialog. Select the check box to list all objects (default).

Objects

List objects of the indicated type or types.

Status

List objects of the indicated status.

Disposition

List the objects selected (Selected) or cleared (Unselected) on the Create dialog.

Command Buttons

For convenience, you can **Select All** and clear criteria check boxes for objects to be eliminated from the list. Similarly, use **Deselect All** to clear all criteria check boxes and selecting those that apply.

Convert DBMS and Optim Primary Keys and Relationships

By default, DBMS primary keys and relationships are created as DBMS objects, while Optim primary keys and relationships are created as Optim objects. You can choose to change the type, however, converting certain Optim primary keys and relationships to DBMS objects and analogous DBMS objects to Optim objects.

Only Optim relationships identified as Rel, can be converted to create DBMS objects. Optim extended relationships, identified as Extended Rel, cannot be created as DBMS relationships. (Extended relationships emulate application-managed relationships and are unique to Optim; see Chapter 9, "Relationships," on page 213 for details on extended relationships.) DBMS foreign keys can be saved as Optim objects, however.

To convert an object, right-click the name and select **Switch to DBMS Key** or **Switch to Optim Key**.

Note: If you select **Switch to Optim Key** for a DBMS relationship whose name is greater than 64 bytes, Optim will truncate the name to 64 bytes (the Optim relationship maximum).

Create Objects

You can create any table or other object for which the status is blank (i.e., the object is not in Conflict, Exists, Pending, etc. status). For an object that is subordinate to a table, the generated SQL must create both the associated table and the object or the table must already exist. Generally, the best method for creating an object depends upon the number of objects you want to create and the associations among them.

- To create a single object, right-click the name and select **Create Object** from the shortcut menu. The **Select** check box setting is ignored.
- To create a table and subordinate objects, right-click the table name and select **Create All Selected Table Objects** form the shortcut menu. The **Select** check box setting is ignored for the table but governs the subordinate objects to be created.
- To create several unrelated objects listed on the tab, select the appropriate **Select** check boxes before selecting **Create All Selected Objects** from the **Tools** menu.
- To create all objects listed on the tab, use the **Select All Objects** shortcut menu command before selecting **Create All Selected Objects** from the **Tools** menu.

Note: A generic Optim primary key or relationship is automatically created as an explicit object.

Drop Objects

There are different ways to drop objects, depending on whether you want to drop a single object or drop all selected objects. Using the Create Utility, you can drop any target object for which the Status is Conflict, Created, or Exists.

- To drop a single object, right-click the name and select **Drop Object** from the shortcut menu. By dropping a table, you also drop any subordinate objects. The **Select** check box setting is ignored.
- To drop a table and subordinate objects, right-click the table name and select **Drop All Selected Table Objects**. The **Select** check box setting is ignored for the table but governs the subordinate objects to be dropped.
- To drop several unrelated objects listed on the tab, select the appropriate **Select** check boxes before selecting **Drop All Selected Objects** from the **Tools** menu.
- To drop all objects listed on the tab, use the **Select All Objects** shortcut menu command before selecting **Drop All Selected Objects** from the **Tools** menu.

Browse SQL for Objects

Generally, the best method for displaying generated SQL depends on the number of objects you want to review and the associations among them.

Note: When browsing generated SQL, you cannot edit it. To edit the generated SQL, you must select **Review SQL** from the **Options** menu and select **Create Object** from the shortcut menu, or **Create All Selected Objects** from the **Tools** menu.

- To review the SQL for a single object, right-click the name and select **Browse Object SQL** from the shortcut menu. You are not required to select the **Select** check box.
- To review the SQL for a table and the subordinate objects, right-click the table name and select **Browse All Selected Table Objects**. You are not required to select the **Select** check box for the table, but must select it for any subordinate objects to be created.
- To browse the SQL for several selected objects listed on the tab, select the appropriate **Select** check boxes before selecting **Browse All Selected Objects** from the **Tools** menu.
- To browse the SQL for all objects listed on the tab, use the **Select All Objects** shortcut menu command before selecting **Browse All Selected Objects** from the **Tools** menu.

Defaults Dialog

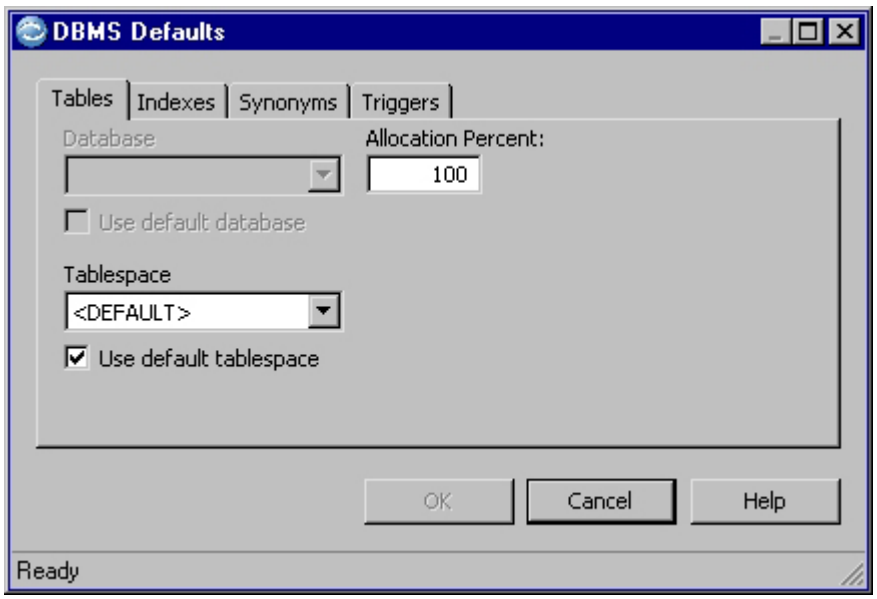
You can establish as many as three layers of default settings for the creation of database objects, in addition to target system defaults. The default settings determine the values displayed in the Object List for the Create Utility and can be overridden at the object level by editing the list.

At the broadest level, DB Alias settings establish defaults for creating objects in the associated database. If desired, you can provide Personal Options settings for a user or group of users that override some or all DB Alias settings. A third level of optional defaults apply at the processing level to override Personal Options and DB Alias settings. Use the Defaults dialog to establish this level of default settings for creating objects. To open a Defaults dialog, select the DB Alias name from the **Defaults** menu on the Create dialog.

The Defaults dialog provides tabs that allow you to define defaults by object type. The display of tabs and options on the Defaults dialog varies according to the DBMS associated with the selected DB Alias. When you close the dialog, the utility validates any settings and, if indicated, resets values in the Create dialog.

Tables Tab

Use the **Tables** tab to provide defaults for creating database tables and subordinate objects. **Database** is enabled for a DB Alias associated with DB2 MVS, while **Tablespace** (or **Segment**, **Filegroup**, or **DBSpace**) is enabled for Oracle, DB2 UDB, DB2 MVS, Sybase ASE, SQL Server, or Informix with the appropriate label.



Database
The default database (for DB2 MVS only). Click the down arrow to select:

<DEFAULT>
Use the default database for the target system.

<SOURCE>
Use the source database.

databasename
Use the designated database.

Use default database
Select **Use default database** to enable the **Database** setting.

Clear the **Use default database** check box to ignore the **Database** setting and create tables and subordinate objects in the target database that matches the source.

- If the name of a target database does not match the source name, tables and subordinate objects are created in a database designated by the *databasename* value, if any, in **Database**.
- If the name of a target database does not match the source name or *databasename* value in **Database**, the target system default applies.

Tablespace
The default Tablespace, Segment, Filegroup, or DBSpace (the label is appropriate to the DBMS). Click the down arrow to select:

<DEFAULT>
Use the target system default.

<SOURCE>
Use the source.

name Use the designated Tablespace, Segment, Filegroup, or DBSpace.

Use default tablespace

Select **Use default tablespace** to enable the **Tablespace** setting.

Clear **Use default tablespace** to ignore the **Tablespace** setting and create tables in the target tablespace that matches the source.

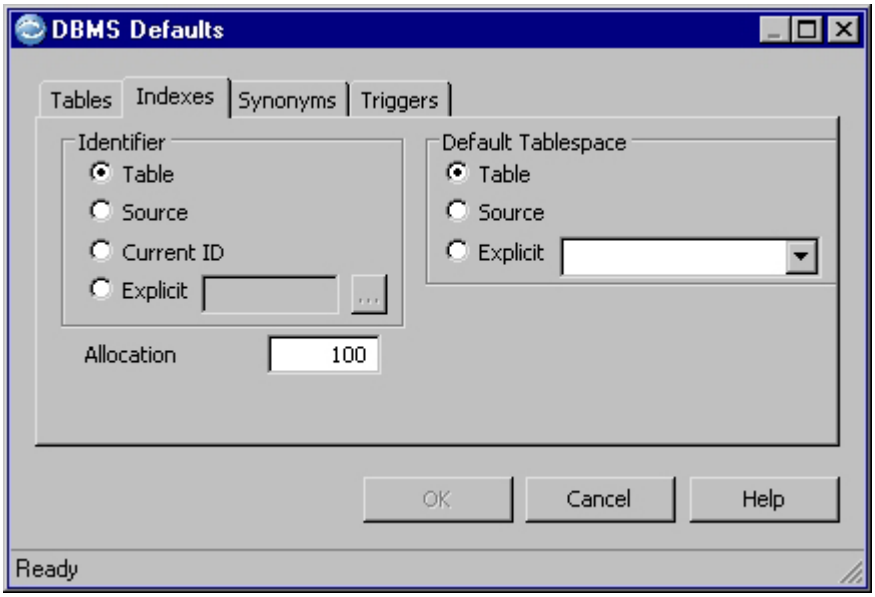
- If the name of a target tablespace does not match the source name, tables and subordinate objects are created in a tablespace designated by the *name* value, if any, in **Tablespace**.
- If the name of a target tablespace does not match the source name or *name* value in **Tablespace**, the target system default applies.

Allocation Percent

Percentage by which storage-related SQL parameters for the source object are adjusted (for Oracle and DB2 MVS). The allowable range is 0 to 999. The default value is 100. Specify zero (0) to use the target system default.

Indexes Tab

Use the **Indexes** tab to provide defaults for creating database indexes.



Identifier

The default Identifier for new indexes. This value is used to assign a unique name to the index at the destination and provides a convenient way to group and identify indexes. Select:

Table Use the source table.

Source

Use the Identifier for the source index.

Current ID

Use the current DBMS user ID for the destination index.

Explicit

Use an explicit Identifier (1 to 64 characters). Type the identifier in the associated box.

Allocation

Percentage by which storage-related SQL parameters for the source object are adjusted (for Oracle and DB2 MVS). The allowable range is 0 to 999. The default value is 100. Specify zero (0), to use the target system default.

Default Tablespace
A Tablespace, Segment, or FileGroup (the label is appropriate to the DBMS) is required for Oracle, Sybase ASE, or SQL Server indexes. Select:

Table Use the Tablespace, Segment, or FileGroup for the source table.

Source
Use the Tablespace, Segment, or FileGroup for the source index.

Explicit
Use an explicit Tablespace, Segment, or FileGroup. Click the down arrow to select from a list of tablespaces for the target database or select **<DEFAULT>** to use the target system default.

Buffer Pool
The default Buffer Pool (for DB2 MVS only) used to create an Index. You can type an explicit value (e.g., BP1) or click the down arrow to select:

<DEFAULT>
The target system default buffer pool.

<SOURCE>
The buffer pool for the source index.

bpname
An index buffer pool specified for the DB Alias or as a Personal Option.

Aliases Tab (DB2)

Use the **Aliases** tab to provide defaults for creating DB2 Aliases (for DB2 UDB, or DB2 MVS).

Identifier
The default Identifier for new aliases. This value is used to assign a unique name to the alias at the destination and provides a convenient way to group and identify aliases. Select:

Object
Use the identifier for the underlying source object.

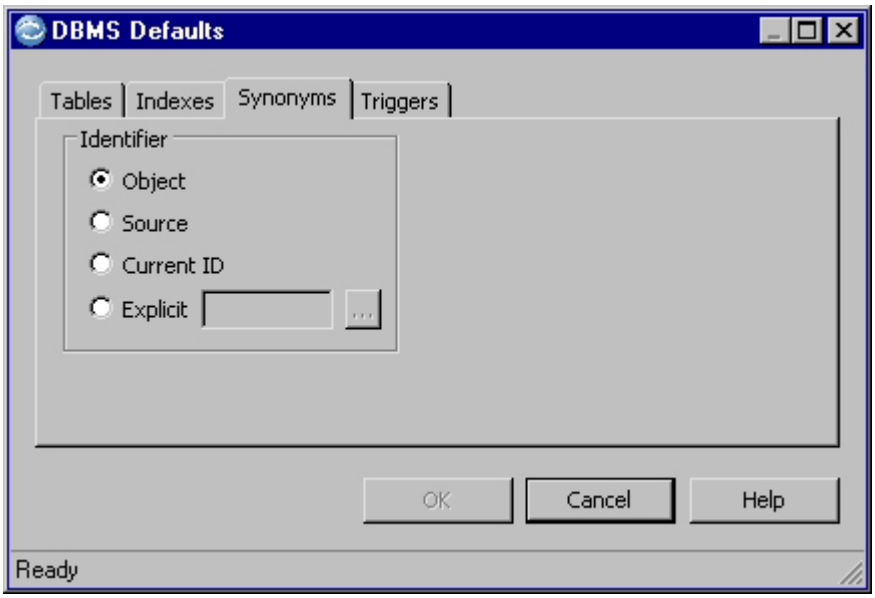
Source
Use the identifier for the source alias.

Current ID
Use the current DBMS user ID for the target alias.

Explicit
Use an explicit Identifier. Type the identifier to be used (1 to 64 characters).

Synonyms Tab (Oracle/Informix)

Use the **Synonyms** tab to provide defaults for creating Synonyms (for Oracle or Informix).



Identifier
The default Identifier for new synonyms. This value is used to assign a unique name to the synonym at the destination and provides a convenient way to group and identify synonyms. Select:

Object
Use the Identifier for the underlying source object.

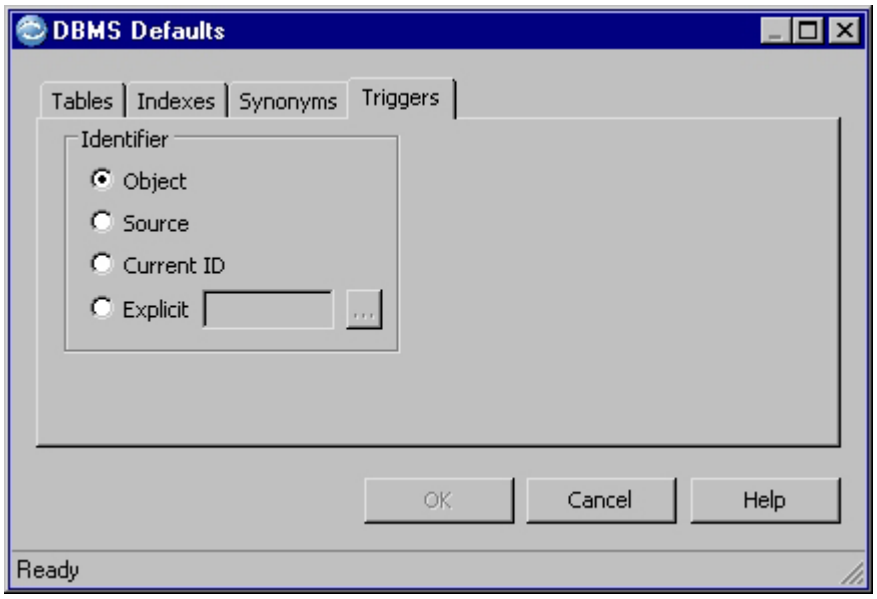
Source
Use the Identifier for the source synonym.

Current ID
Use the current DBMS user ID for the target synonym.

Explicit
Use an explicit Identifier. Type the identifier to be used (1 to 64 characters).

Triggers Tab

Use the **Triggers** tab to provide defaults for creating Triggers (for DB2 UDB, Oracle, Sybase ASE, SQL Server, or Informix).



Identifier
The default Identifier for new triggers. This value is used to assign a unique name to the trigger at the destination and provides a convenient way to group and identify triggers. Select:

Object
Use the Creator ID for the owning source object.

Source
Use the Creator ID for the source trigger.

Current ID
Use the current DBMS user ID for the target trigger.

Explicit
Use an explicit Identifier. Type the identifier to be used (1 to 64 characters).

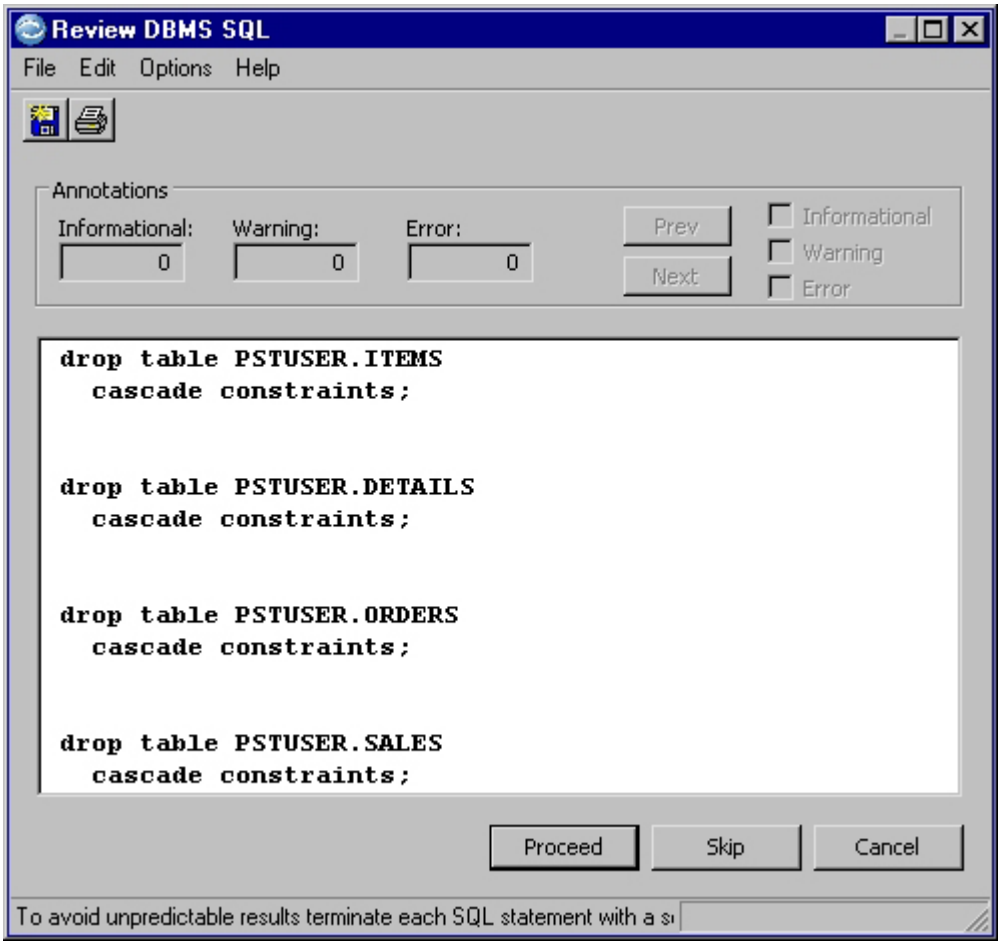
Process the SQL

When you select **Create All Selected Table Objects**, **Drop All Selected Table Objects**, **Create Object**, or **Drop Object** from the shortcut menu, the Create Utility generates the SQL needed to carry out the command. The SQL for each DB Alias is generated separately.

Review and Edit SQL

From the **Options** menu, select **Review SQL** to display the generated SQL prior to submitting it for execution.

You can edit the generated SQL before it is submitted, but the utility does not validate your changes. Most DDL statements end with a semicolon. You must begin any added DDL statement that has embedded semicolons with a BEGIN_SQL statement and end it with an END_SQL statement.



In this example, a table named TEST.CUSTOMERS is to be created. The SQL is displayed in the Review dialog. Note that the DB Alias is included in the title. (In this case the DB Alias is DBMS).

Annotations

The annotations at the top indicate conditions identified while generating the SQL. These include informational, warning and error messages. Use the check boxes at the right to designate the types of message to target when you scroll by clicking **Prev** and **Next**. For example, select the Warning check box to scroll to the warning messages.

Although you may want to review informational and warning messages, only error messages prevent the SQL from executing.

Highlighted Object Names

At times, you may modify the Creator ID and name for a target object in the Table Map or the Create dialog, typically when a DDL type of object is targeted to a database different from the source. If the object is retrieved from the source DBMS catalog in text format (for example, a View or an Oracle Procedure), an SQL parser replaces Creator IDs and object names from the source with the appropriate target Creator IDs and object names. Any replacement text is highlighted to help you review and evaluate the changes. The default colors are blue and yellow, but you can modify the defaults by changing settings on the **Create** tab in Personal Options.

- When the SQL parser is **confident** of a change, the object name is highlighted in the default color **blue**.
- If the SQL parser is **uncertain** of a change, the object name is highlighted in the default color **yellow**.

You can right-click highlighted object names and choose commands from the shortcut menu as follows:

- To display the object name from the source file, select **Show Source Object Name**.
- To browse the source SQL for an object, select **Browse Source Object SQL**.

Command Buttons

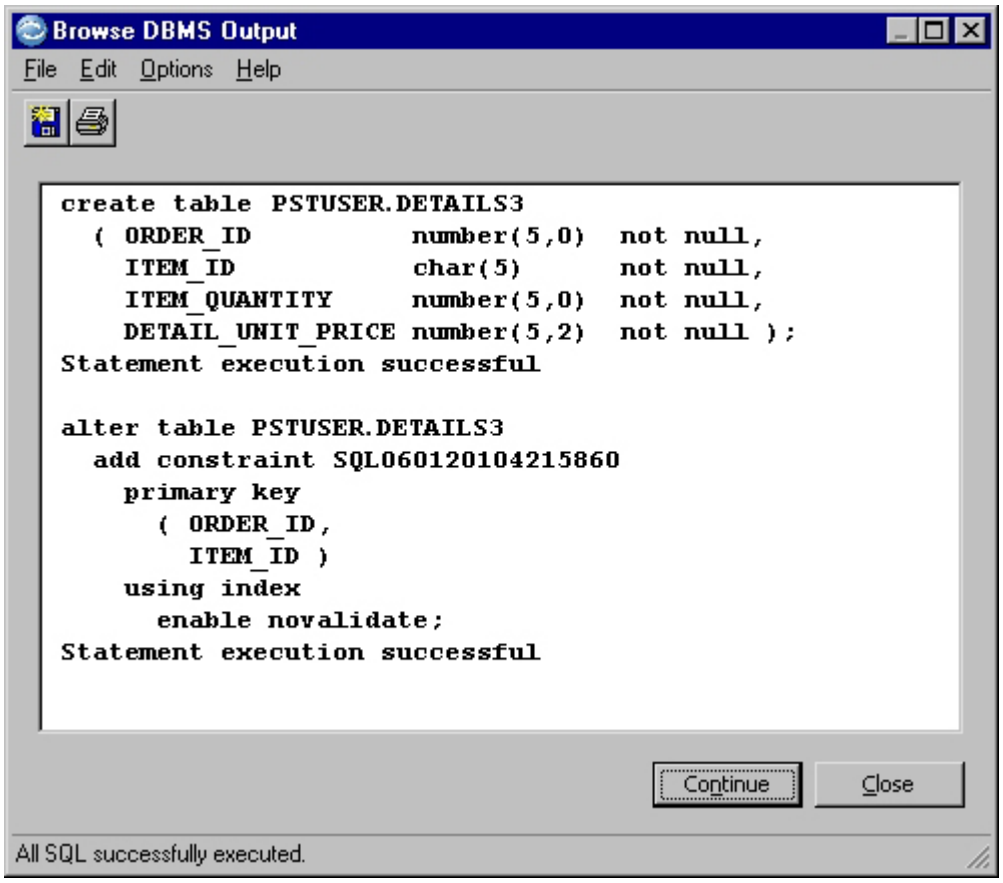
Proceed
Submit the SQL. The Browse Output dialog opens to display the information generated by the database as a result of executing the SQL.

Skip Do not submit the SQL for the current DB Alias and display the SQL for the next. If there is no SQL for other DB Aliases, return to the Create dialog.

Browse Output Dialog

If, before processing, you select **Browse SQL Output** from the **Options** menu to display the executed SQL and any resulting DBMS completion messages in the Browse Output dialog. This option is set by default.

If you clear this option, the Browse Output dialog is opened for SQL execution errors only. The Browse Output dialog displays the executed SQL and DBMS error messages.



Continue

If execution is unsuccessful, the Create Utility opens the Review SQL dialog for editing. If execution is successful and SQL for another DB Alias remains to be executed, it is displayed. If processing is successful and no SQL for other DB Aliases remains, you are returned to the Create dialog.

When you finish viewing the report, close the dialog to return to the Create dialog, where you can create, drop, or browse other objects, or exit the Create Utility.

Chapter 18. Personal Options

You can use Personal Options to customize Optim for use at each workstation.

For example, you can:

- Specify default data directories, set user-defined database logon and password information, and select other options to customize display features and message text.
- Provide defaults for the Schedule and Create Utilities, and the Load, Edit, and Browse Processes.

Personal Options are recorded in the Windows Registry of the workstation.

Note: Site-level Product Option settings supersede any conflicting user-level Personal Option settings. For more information, refer to the *Installation and Configuration Guide* .

Open the Personal Options Editor

You can use the **Configuration** program to configure Personal Options, or you can set options from within Optim.

Using the Configuration Program

To configure Personal Options using the Configuration program:

1. Open the Configuration program.
2. In the main window, select **Configure Options** from the **Tasks** menu.
3. Specify an Optim Directory and click **Proceed** to open the Specify Product Configuration File dialog.
4. Select **Create New File** or **Use Existing File**, verify the name of the Configuration File, and click **Proceed**.
5. On the Modify Product Options dialog, click **Proceed** to open the Modify Personal Options dialog.
6. Click the **Personal Options** button to open the Personal Options dialog.
7. Specify the necessary details on each tab in Personal Options.
8. Choose one of the following:
   - To close the Personal Options dialog without saving your changes, click **Cancel**.
   - To save your changes and continue using the Personal Options dialog, click **Apply**.
   - To save your changes and close the Personal Options dialog, click **OK** to return to the Modify Personal Options dialog.
9. Click **Proceed** to complete the process.

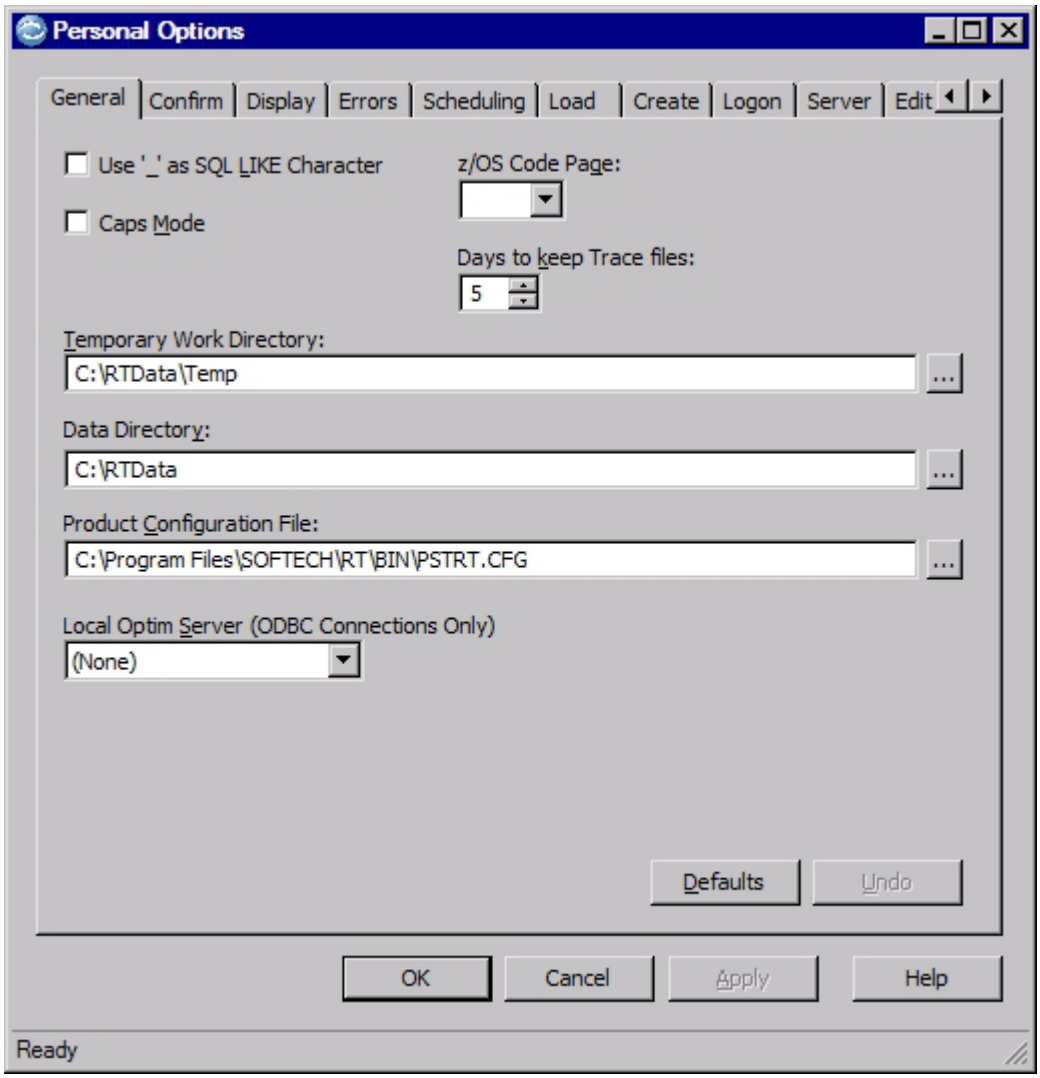
Within Optim

To change the settings within Optim:

1. In the main window, select **Personal** from the **Options** menu.
2. On the Personal Options dialog, provide the necessary information on each tab.
3. Choose one of the following:
   - To close the Personal Options dialog without saving your changes, click **Cancel**.
   - To save your changes and continue using the Personal Options dialog, click **Apply**.
   - To save your changes and close the Personal Options dialog, click **OK**.

Using the Editor

The Personal Options dialog allows you to customize Optim for the workstation.



Tabs

The tabs in the Personal Options dialog are described briefly in the following paragraphs. Detailed information is provided in each section of this chapter.

General

Identify the Temporary Work Directory, Data Directory, and the directory for the Product Configuration File used by the workstation. Options allow you to use the underscore as an SQL LIKE character, select the uppercase or lowercase default for selection criteria, establish the number of days Trace Files are retained, specify a default code page value, and elect to warn a user when a cascading delete or update may occur.

Confirm

Choose to display a confirmation prompt before definitions are deleted, files overwritten, or DDL lost.

Display

Choose options for data displays when editing data or using Point and Shoot, including the

maximum number of rows fetched and the maximum number of entries in history and menu file lists. Also, set options to determine the default settings for Large Objects for the Access Definition.

Errors Select font characteristics for informational, warning, and error messages and the number of lines to display in the message bar.

Scheduling
Select options for using the Scheduling Monitor.

Load Specify the complete path and name of the executable used to access each DBMS Loader.

Create Specify default options for creating database objects.

Logon Review logon information for each DB Alias in a selected Optim Directory. Connection and password information is stored in the workstation registry.

Server Specify User ID, password, and domain information to be used by the Optim Server (Server) when running actions remotely.

Edit Select options that apply when you browse or edit database tables.

Browse
Select display options for browsing Archive, Extract, Control, or Compare Files. Set defaults for emphasizing differences between rows when browsing Compare Files.

Archive
Specify the default Archive Directory and Archive Index Directory. Select options that apply when you archive data.

Removable Media
Specify default segment size values.

Actions
Set options for displaying tabs in action request editors, printing Column Map procedures, and retaining reports.

Printer
Set printer, font, and language preferences for printing a request or definition.

Database
Select options for handling multi-byte round trip conversion errors, cascade deletes, and database connections. (For Sybase ASE, you can also specify whether to run in Unchained mode.)

Notify Specify default options and a list of addresses used for sending an email message when a request is processed, or only if a request succeeds or fails.

General Tab

Use the General tab to identify the Temporary Work Directory, Data Directory, and the directory for the Product Configuration File used by the workstation, and other defaults.

Use '_' as SQL LIKE Character

Select to use the underscore character to represent any single character in a pattern. For example, if **Use '_' as SQL LIKE Character** is selected, you can type PSTDEMO.M _ _ P in the pattern box on the Open an Access Definition dialog to list only Access Definitions with a name beginning with M and ending with P for the Identifier PSTDEMO. If you clear this check box, the underscore represents the underscore character.

Caps Mode

Select to translate all string literal characters in selection criteria, relationships, and Column Maps to uppercase. If you clear this check box, characters display exactly as entered.

z/OS Code Page

In an Optim process, you can use an Extract File created using Optim z/OS. Optim uses a code page to convert the mainframe file format. In the **z/OS Code Page** list, select a default value to be used if the Extract File does not contain a code page number.

Days to keep Trace files

Specify the number of days (2 to 30) to retain trace files in the temporary work directory. The default value is 5.

Trace files are useful for tracking processing. Each trace file name begins with PR0 and ends with a numeric extension (for example, PR0TOOL.123). The second part of the name identifies the type of trace file. Trace files are sequentially numbered .001 through .999, followed by .A00 through .Z99, as necessary. If more than 3,599 trace files of a type are created and stored within the specified number of days, file names are reused, beginning with the first.

Note: Consider storage space limitations when deciding the number of days to retain the files.

Temporary Work Directory

Specify the complete path to the default directory in which you want to store internal work files and trace files. To select from your system directories, click the browse button.

Data Directory

Specify the complete path to the default directory in which you want to store Archive, Extract, Control, Compare, and Export Files, and other process files. To select from your system directories, click the browse button. The Data Directory serves as the default; you can specify a different directory path and file name on any process request.

Product Configuration File

Specify the complete path to the Product Configuration File. The file name has a **.cfg** extension. To select from your system directories, click the browse button. The Product Configuration File is created when you install Optim.

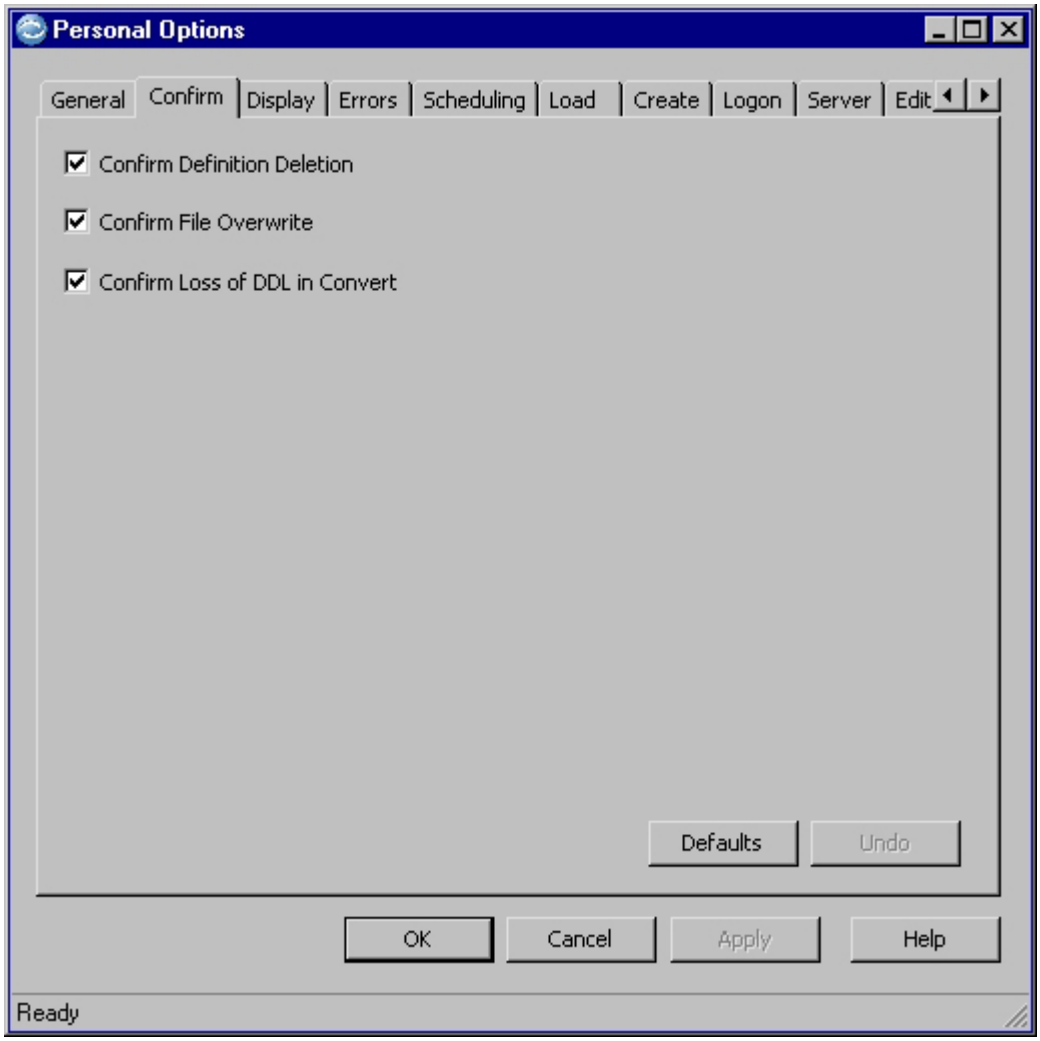
Local Optim Server (ODBC Connections Only)

To improve ODBC response times, select the name of the Optim Server that runs on this machine. The **ODBC Server** will run locally and the Server will not be contacted, if an ODBC connection specifies this Server name, or if the ODBC interface selects an Archive File with this Server name in its Archive Directory entry.

Note: Do not use this setting if accessing archived data on a backup device.

Confirm Tab

Use the Confirm tab to choose whether a confirmation dialog is displayed before the execution of a process that results in loss of data.



Confirm Definition Deletion

Select to display a confirmation dialog before you delete a definition or process request from the Optim Directory. Selecting this check box helps prevent deleting a request or definition accidentally.

Confirm File Overwrite

Select to display a confirmation dialog before you overwrite an existing file (Archive File, Extract File, Control File, Compare File, or Export File).

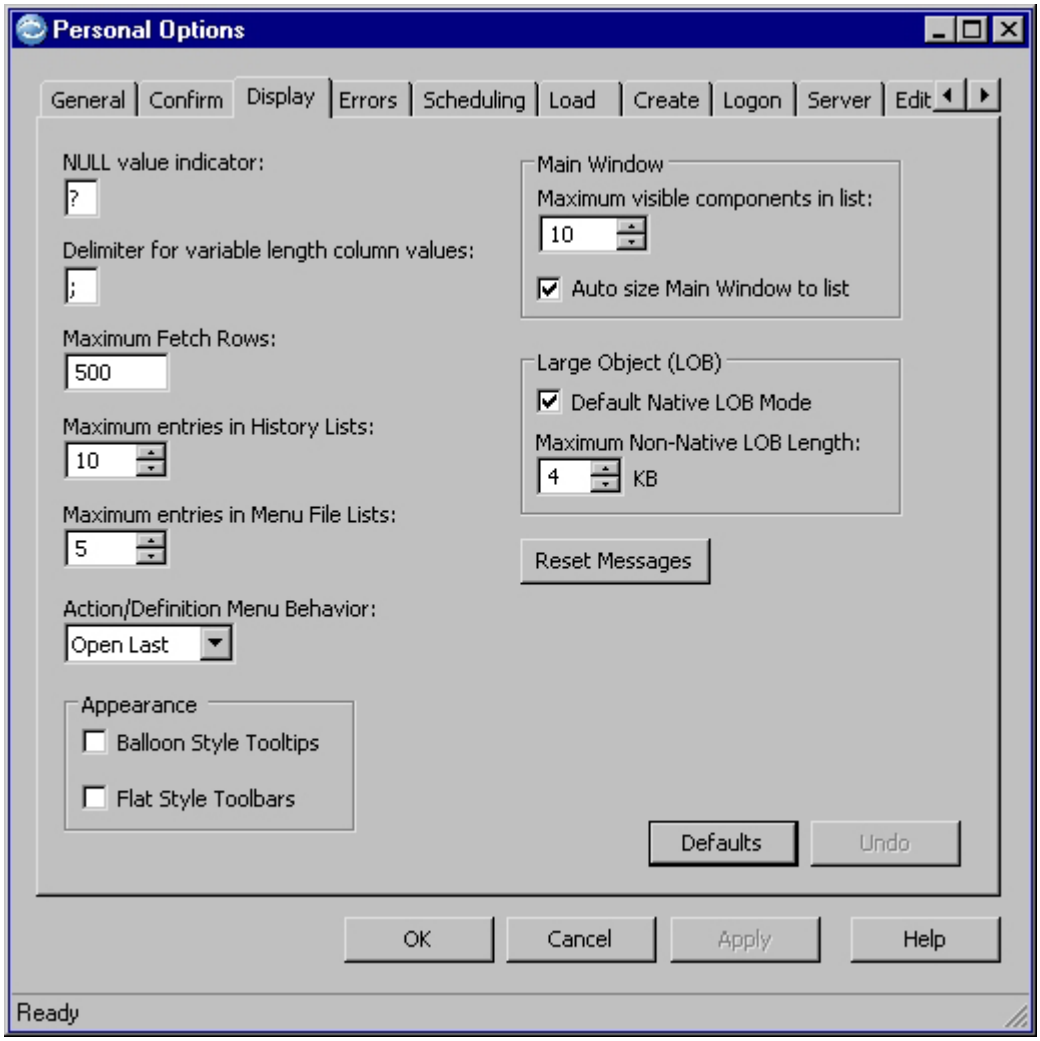
Note: If you select this check box, a confirmation dialog opens before you run a process. However, this confirmation does not display for a process request that is scheduled or run using the Command Line Interface.

Confirm Loss of DDL in Convert

Select to display a confirmation dialog during a Convert Process that can result in the loss of DDL statements. The confirmation dialog is not displayed for a scheduled process request.

Display Tab

Use the **Display** tab to select preferences for displaying information.



NULL value indicator

Enter a character to represent a NULL value. The question mark (?) is the default. Although you can choose any character to represent a NULL value, an infrequently used character is best.

Delimiter for variable length column values

Enter a character to delimit variable length column values that have trailing blanks. The semi-colon (;) is the default. Although you can choose any character as a delimiter, an infrequently used character that differs from the NULL value indicator is best.

Maximum Fetch Rows

Specify the maximum number of rows from a single table that can be displayed when you browse or edit table data or use Point and Shoot. You can enter a number from 1 through the site maximum, specified on the Product Options dialog (see the *Installation and Configuration Guide*). The default value is 500.

Maximum entries in History Lists

Specify the maximum number of recently selected items (1 to 20) displayed in a drop-down list. For example, the Extract File box in a request editor, where you can click the down arrow to view a list of the most recently used Extract Files.

Maximum entries in Menu File Lists

Specify the maximum number of items (1 to 20) displayed below the **File** menu commands in an editor. For example, a list of the most recently opened definitions appears on the **File** menu in the Access Definition Editor.

Action/Definition Menu Behavior

Click the down arrow to select the default mode for opening the action or definition editors from the **Actions** or **Definitions** menu in the main window.

Open Last
Opens the last edited request in the selected editor. This is the default.

Open New
Opens a new, untitled request in the selected editor.

Appearance

Options for displaying tooltips and toolbars on dialogs and editors in Optim.

Balloon Style Tooltips
Controls the appearance of the tooltips. Select the check box to choose balloon-style. Clear the check box to choose flat-style (default).

Flat Style Toolbars
Controls the appearance of the toolbars. Select the check box to choose flat-style. Clear the check box to choose 3D style (default).

Main Window

Options for displaying the list of active dialogs and editors in the main window. As you open each editor or dialog, the main window expands to list the active editors and dialogs. To recall an active editor or dialog, double-click the item in the list.

Maximum visible components in list
Specify the maximum number of active editors or dialogs (5 to 99) displayed in the main window. The default value is 10. If the number of active editors and dialogs exceeds the maximum, you can scroll the list. (If fewer editors or dialogs are in use, the bottom of the list is blank.)

Auto size Main Window to list
Select to automatically resize the main window as editors and dialogs are added to the list. If you clear this check box, the main window is sized to display the maximum number of editors and dialogs that can be listed.

Large Object (LOB)

Default Native LOB Mode
Select this check box to set the default for the **Native LOB Mode** check box on the **Columns** tab of the Table Specifications dialog, available through the Access Definition Editor. You can use the check box on the Table Specifications dialog when preparing an Access Definition for use with

Edit. When editing, the check box designates whether to start the native application associated with an LOB, or to process as a VarChar or VarBinary data type.

Maximum Non⌂Native LOB Length
Specify the maximum length of data to retrieve from a database for an LOB processed as a VarChar or VarBinary data type when using the Table Editor in **Edit**. Select a value from 1 KB through 32 KB.

Note: The setting of the **Native LOB Mode** check box on the **Columns** tab of the Table Specifications dialog (available through the Access Definition Editor) determines whether an LOB is processed as Native or Non-Native.

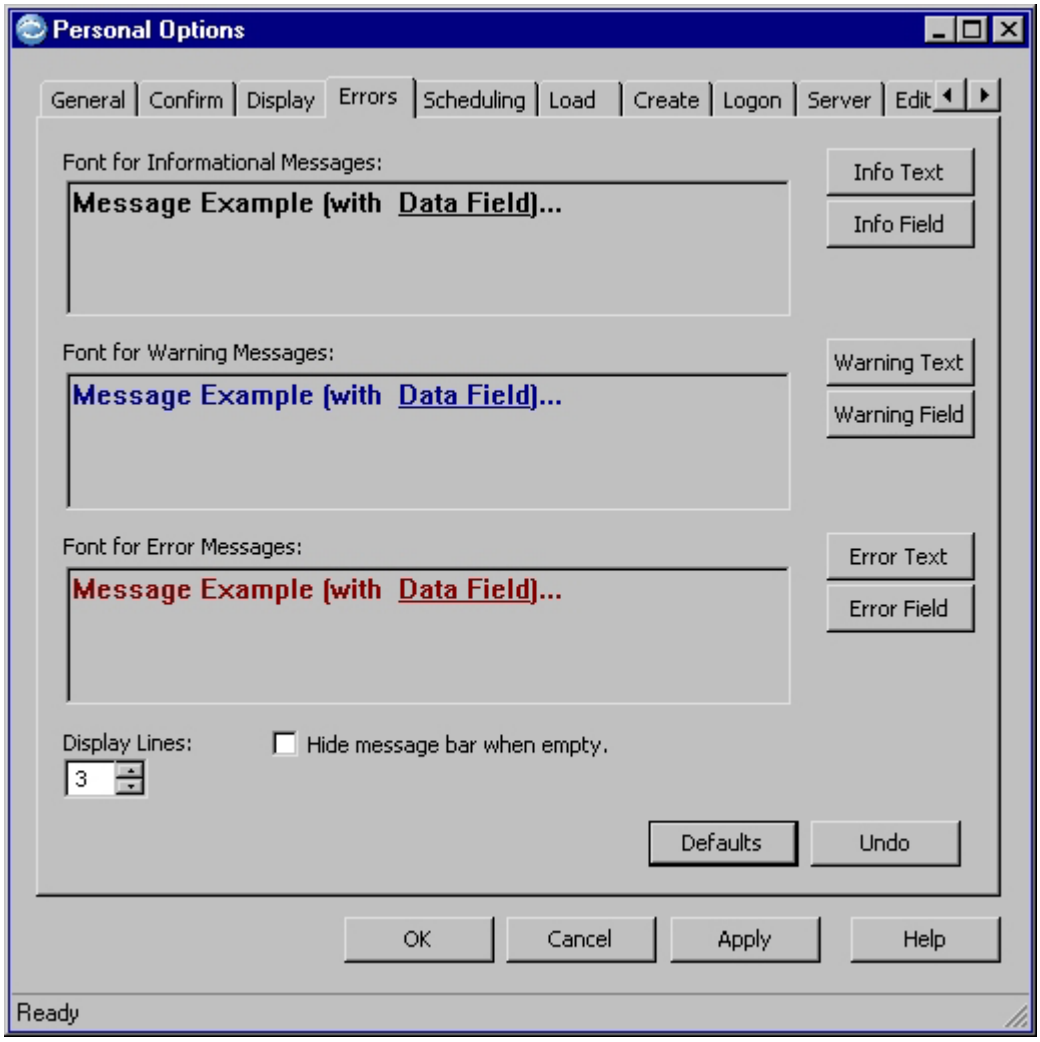
An LOB truncates when the size of the LOB exceeds the Maximum Non-Native LOB Length setting, and appears in the Table Editor as a protected cell with a cross-hatched pattern.

Reset Messages

Click **Reset Messages** to reset system messages. Message dialogs provide information or warnings. You can also choose *not* to display the message again. **Reset Messages** resets the option to display these message dialogs, when appropriate.

Errors Tab

Use the options on the **Errors** tab to set preferences for the display of error messages.



The current font for messages is shown in each font message box. To open the Windows Font dialog to modify font attributes, click the appropriate command button.

Font for Informational Messages

Informational messages are not critical; for example, messages that ask whether information should be saved when a dialog is closed are informational.

Info Text
Select font characteristics for the informational message text. The default is System, 10 point, Bold, Black.

Info Field
Select font characteristics for the data referenced in an informational message. The default is System, 10 point, Bold, Underline, Maroon.

Font for Warning Messages

Warning messages indicate serious, but not critical conditions. A warning message does not interrupt an action, but may indicate that you should reevaluate the current action.

Warning Text
Select font characteristics for the warning message text. The default is System, 10 point, Bold, Maroon.

Warning Field
Select font characteristics for the data referenced in a warning message. The default is System, 10 point, Bold, Underline, Maroon.

Font for Error Messages

Error messages indicate critical conditions and interrupt the current action. A problem presented in an error message must be addressed before the attempted action can proceed. Error messages may be displayed in pop-up dialogs, but are usually displayed in the message bar at the bottom of the dialog.

Error Text
Select font characteristics for the error message text. The default is System, 10 point, Bold, Navy.

Error Field
Select font characteristics for the data referenced in an error message. The default is System, 10 point, Bold, Underline, Navy.

Display Lines

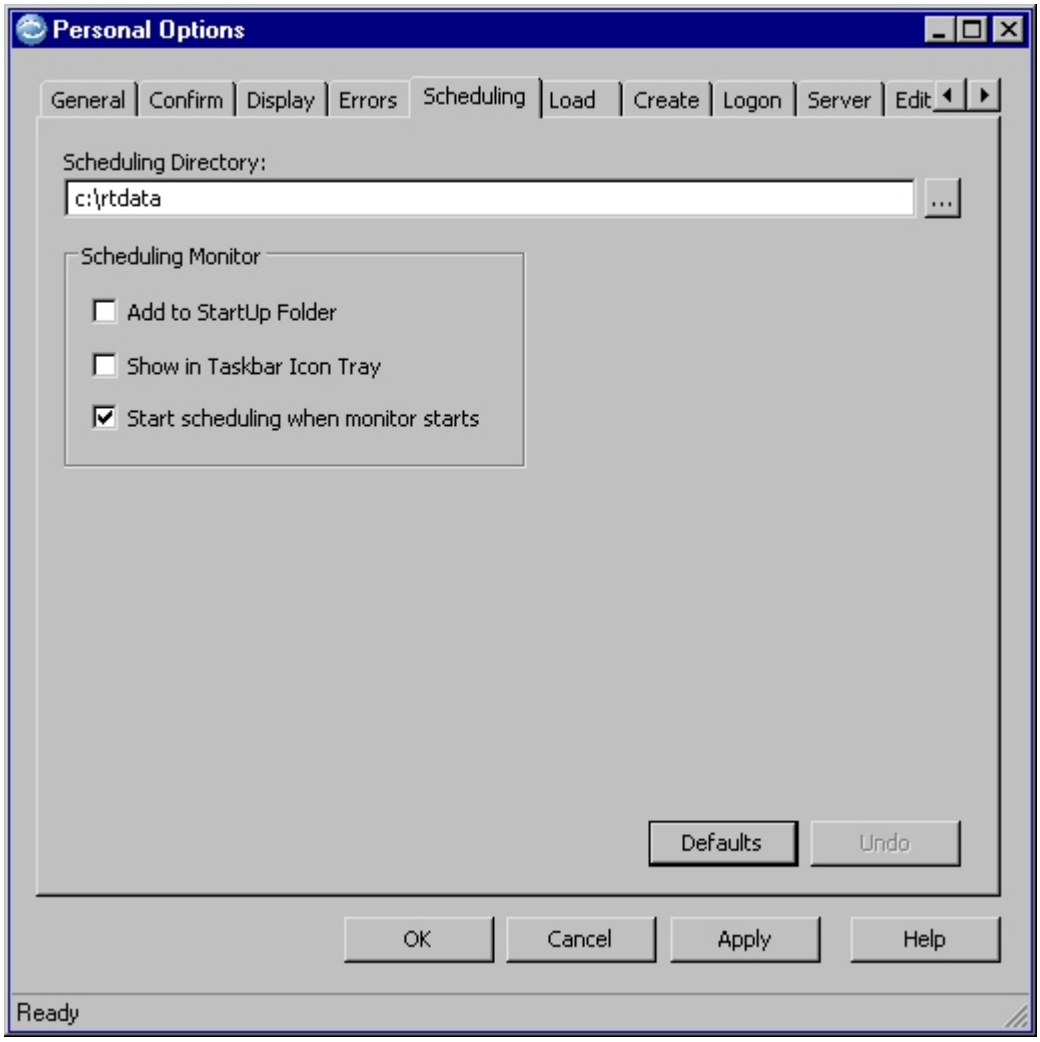
The maximum number of lines (3 to 9) to display in the message bar for any type of message.

Hide message bar when empty

Select to hide the message bar when no messages are displayed. If you clear the check box, the message bar remains at the bottom of each editor or dialog at all times.

Scheduling Tab

Use the options on the **Scheduling** tab to specify a default directory for Optim process requests, and to set options for the Scheduler.



Scheduling Directory

Specify the complete path to the default directory where you want Optim to store schedule files. To select from your system directories, click the browse button. Each user should have a unique directory for storing schedule files.

Scheduling Monitor

Select options for using the Scheduling Monitor. The Scheduling Monitor must be active for scheduled jobs to be processed.

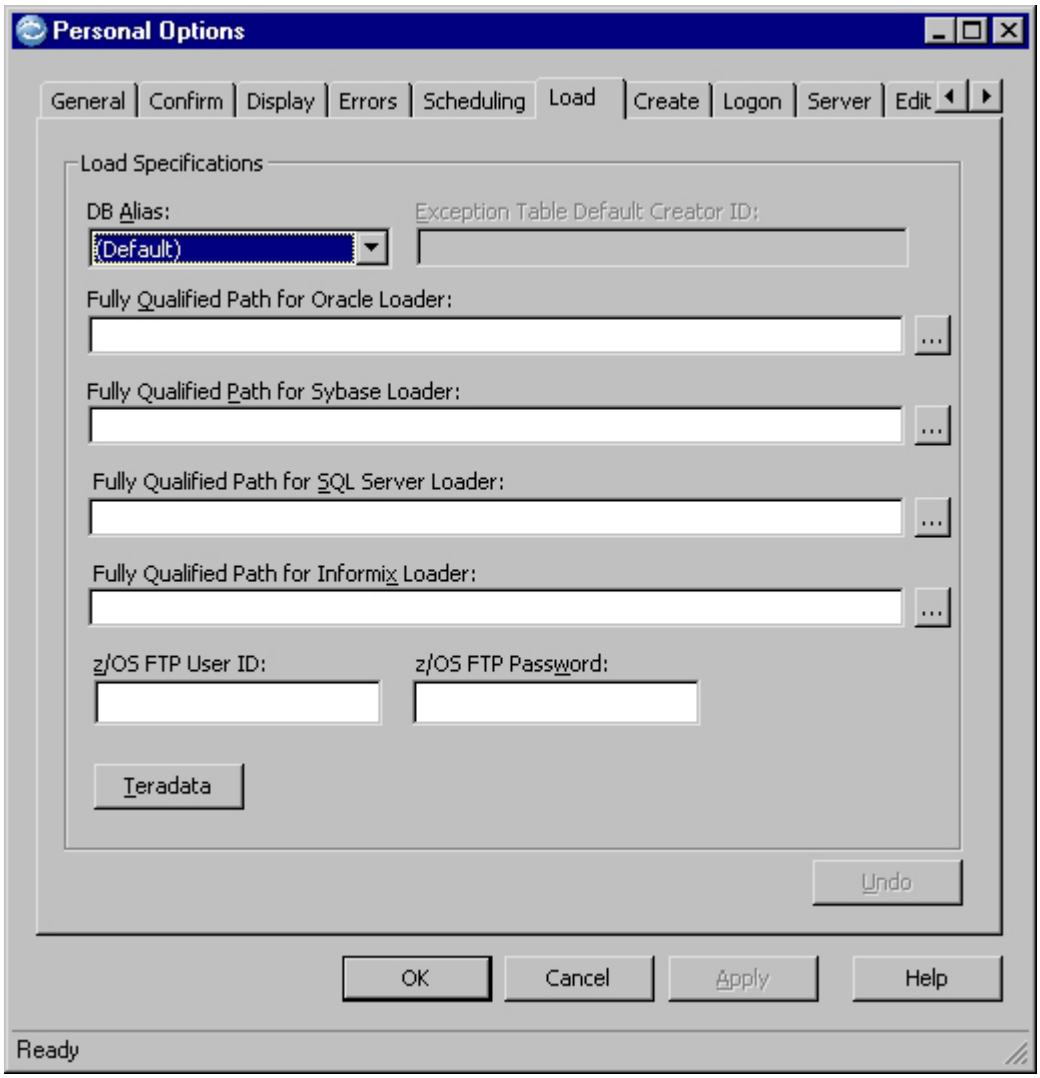
Add to StartUp Folder
Select this check box to add the Scheduler to the group of programs that start automatically when you start Windows.

Show in Taskbar Icon Tray
Select this check box to display the Scheduler icon in the Windows task bar icon tray, instead of as a button on the taskbar.

Start scheduling when monitor starts
Clear this check box to prevent starting scheduled jobs immediately when the Schedule Monitor starts.

Load Tab

Use the options on the **Load** tab to set specifications for a Load Process.



Load Specifications

Provide the complete path and name of the executable to access each DBMS Loader that can be used with a Load Request.

DB Alias

Select **(Default)** to enter the path and name of the Loader executable file for each DBMS type.

If you have more than one version of a particular DBMS type, you can enter the unique loader specification for each version. Click the down arrow to select the specific DB Alias, then enter the appropriate path and name of the Loader executable file for the particular DBMS version.

Exception Table Default Creator ID

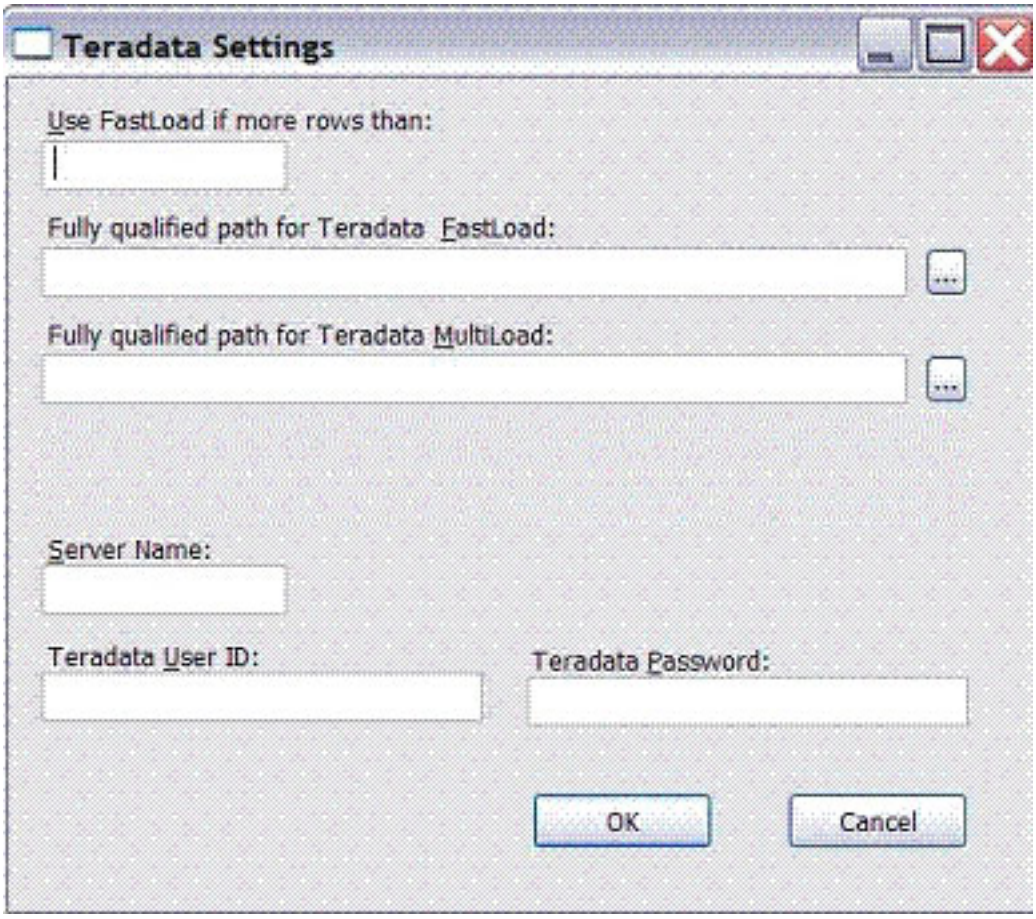
Provide a default Creator ID for exception tables (DB2 and Oracle) or violation tables (Informix). This field is available only when you select a DB Alias for a DB2, Oracle, or Informix database.

Fully Qualified Path for DBMS Loaders
Provide the directory path and program name for the specific DBMS Loader. Consult your DBMS documentation for the name of the loader program. To select from your system directories, click the browse button.

z/OS FTP User ID
Provide the default user ID for the z/OS FTP server, used when uploading files to the z/OS machine during the Load Process. This option is available when you select the (Default) or a DB2 **DB Alias**. You can override the user ID in the Load Request.

z/OS FTP Password
Provide the password for the default z/OS FTP server user ID. This option is available when you select the (Default) or a DB2 **DB Alias**. You can override the password in the Load Request.

Teradata
Select to provide settings for the Teradata loader. This option is available when you select a DB2 **DB Alias** for a Teradata database. The Teradata Settings panel displays:



Use FastLoad if more rows than
Row count to determine whether FastLoad or MultiLoad is used. Allowable values are 0 - 999,999,999. If you specify 0 or do not specify a value, MultiLoad is used. For any other value, FastLoad is used if the row count of the load file is greater than the value you specify for **Use FastLoad if more rows than**.

Fully Qualified Path for Teradata FastLoad
Provide the directory path and program name for the Teradata FastLoad. Consult your Teradata documentation for the name of the loader program. To select from your system directories, click the browse button.

Fully Qualified Path for Teradata MultiLoad
Provide the directory path and program name for the Teradata MultiLoad. Consult your Teradata documentation for the name of the loader program. To select from your system directories, click the browse button.

Server Name
Name of the Teradata server.

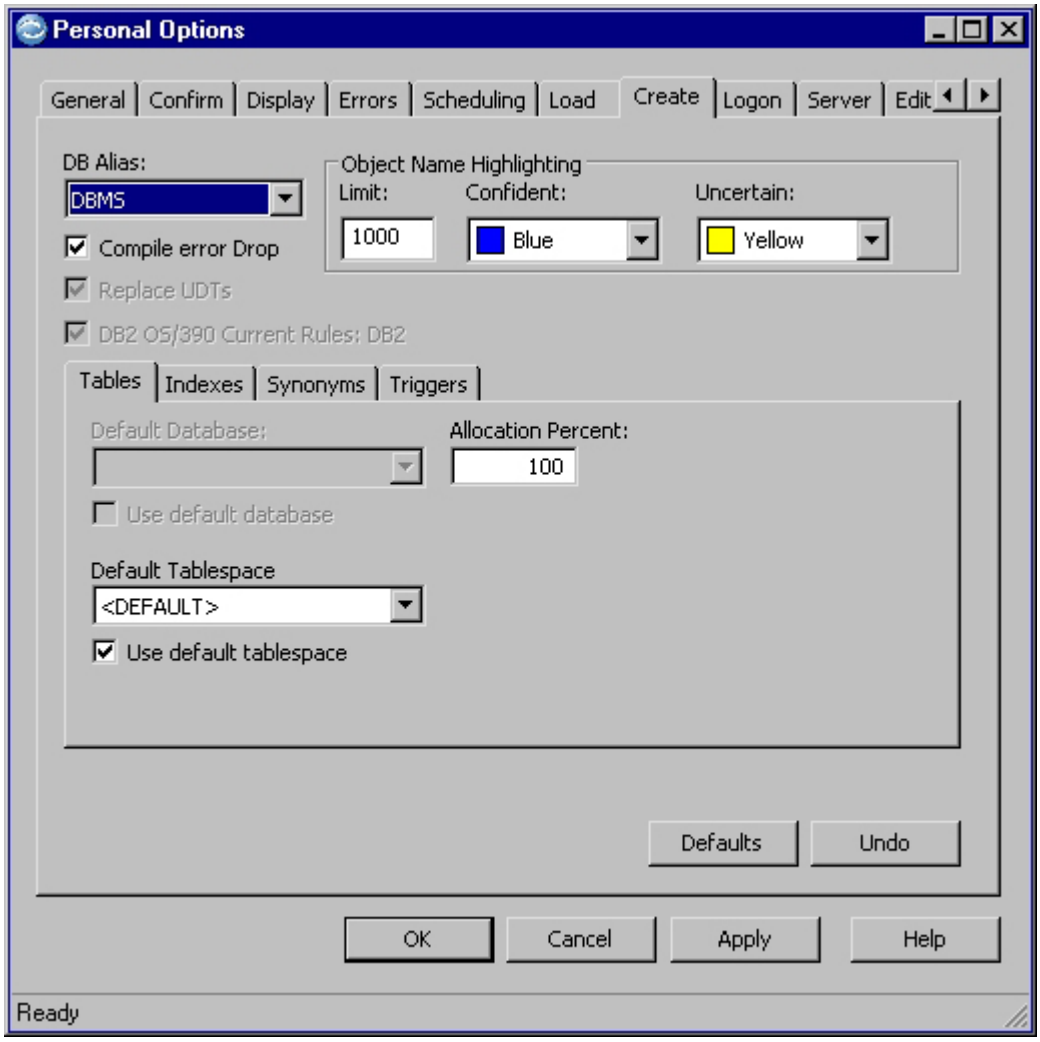
Teradata User ID
Teradata User ID for the user creating the Load Request.

Teradata Password
Teradata password for the user creating the Load Request.

Create Tab

Use the options on the **Create** tab to set defaults for creating objects. Note that you can establish as many as three layers of default settings for the creation of database objects, in addition to target system defaults. The default settings determine the values displayed in the Object List for the Create Utility and can be overridden at the object level by editing the list.

At the broadest level, DB Alias settings establish defaults for creating objects in the associated database. If desired, you can provide Personal Options settings, as described in the following text, for a user or group of users that override some or all DB Alias settings. A third level of optional defaults apply at the processing level to override Personal Options and DB Alias settings. Use the options on the **Create** tab to set second-level defaults for the Create Utility.



DB Alias

Specify the DB Alias that identifies the database in which you want to create database objects. To select from a list, click the down arrow. Specify default options for creating different types of database objects on each corresponding tab.

Compile error Drop

Select this check box to automatically drop any Oracle object that causes a compile error during the Create Process. If you clear this check box and compile errors occur, you must interrupt the Create Process to drop the object before continuing.

This feature applies to Oracle compile errors that may occur on certain database objects: functions, packages, package bodies, procedures, triggers, and views. (The Create Process can create these objects, but they may not be functional.) Select this check box to correct possible problems in the Review SQL dialog before performing the Create Process.

Replace UDTs

Select this check box to replace table-type column references to User Defined [data] Types with base column data types in any generated DDL. When you clear this check box, references to UDTs are preserved in generated DDL. (This check box is available only when you select a DB Alias for a DB2, Sybase ASE, or SQL Server database.)

Note: Clear this check box if you want UDT references in the generated DDL.

DB2 OS/390 Current Rules: DB2

Select this check box to require the user (i.e., Create and DDL) to create and delete LOB tablespaces, AUX tables, and unique Indexes. When you clear this check box, DB2 OS/390 automatically creates and deletes LOB tablespaces, AUX tables, and unique Indexes. This check box is selected by default.

Object Name Highlighting

Select a font color to highlight object name changes in the SQL statements shown on the Review SQL dialog before creating those objects in the target database. During the Create Process, object names (specified in the Table Map) are translated to be appropriate for the target database. This feature applies to creating text type database objects: functions, packages, package bodies, procedures, triggers, and views.

Limit When creating a large number of objects, highlighting object names in color can affect the speed of the process. Specify the maximum number of created objects to highlight in color (i.e., if the number of objects to create exceeds the limit you specify, colorization is not used). The default is 1000.

Confident
Select a font color to highlight object name changes that are reasonably confident. Accept the default color (blue) or click the down arrow to select a different color.

Uncertain
Select a font color to highlight object name changes that may require verification because of the way different DBMSs use object names. Accept the default color (yellow) or click the down arrow to select a different color.

Create — Tables Tab

Use the **Tables** tab to specify the default database (for DB2 MVS). Specify a default tablespace (segment, filegroup, or dbspace) for creating database objects. Specify an allocation percent to adjust SQL storage related parameters (for Oracle and DB2 MVS).

Default Database

Enter the name of the default database for creating tables. To select from a list, click the down arrow. This option is available only if you are using DB2 MVS. A single DB Alias in Optim can identify more than one database in DB2 MVS.

Use default database

Select this check box to use the default database for creating tables. If you clear this check box, the Create Utility attempts to use the source database from the Source File. However, if the source database does not exist on the target system, the Create Utility uses the default database.

Default Tablespace

Enter the name of the default tablespace (segment, filegroup, or dbspace) for creating tables. To select from a list, click the down arrow. If you select <Default>, the default set in the database is used.

Use default tablespace

Select this check box to use the default tablespace (segment, filegroup, or dbspace) for creating tables. If you clear this check box, the Create Utility attempts to use the tablespace (segment, filegroup, or dbspace) in the Source File. However, if the source does not exist on the target system, the Create Utility uses the default.

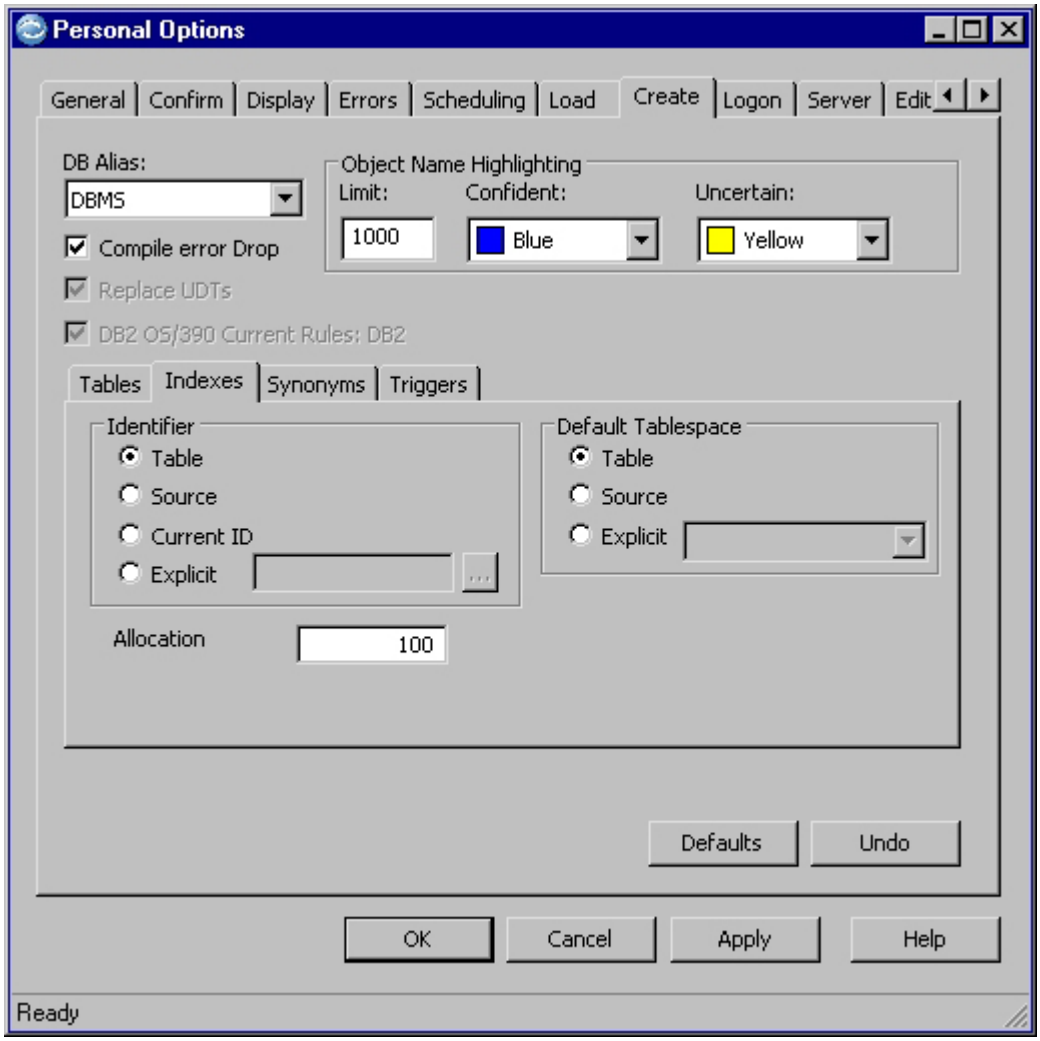
Allocation Percent

Enter a percent (0 to 999) to adjust SQL storage-related parameters for the Create Utility. The default is 100. Allocation percent is available for creating tables and indexes in Oracle and creating indexes in DB2 MVS.

Target SQL is generated based on the values of the objects in a Source File. If you specify zero (0), the storage-related clause in the SQL statement is omitted. Using a value other than zero results in a percentage of the source value being used in the target clause.

Create — Indexes Tab

Use the **Indexes** tab to select a default identifier for creating new indexes. Specify an allocation percent to adjust SQL storage-related parameters (for Oracle and DB2 MVS). Specify the default tablespace (segment, filegroup, or dbspace) for creating indexes.



Identifier

Specify the default identifier for new indexes based on the identifier from one of the following:

Table Use the identifier from a corresponding target table as the default for new indexes.

Source
Use the identifier from the source index as the default for new indexes.

Current ID
Use the current SQLID (User ID) as the default for new indexes.

Explicit
Use an explicit identifier as the default for new indexes. If you select this option, you must specify an explicit identifier (1 to 64 characters). To select from a list, click the browse button.

Allocation

Enter a percent (0 to 999) to adjust SQL storage-related parameters for the Create Utility. The default is 100. Allocation percent is available for creating tables and indexes in Oracle and creating indexes in DB2 MVS.

Target SQL is generated based on the values of the objects in a Source File. If you specify zero (0), the storage-related clause in the SQL statement is omitted. Using any value, other than zero, results in a percentage of the source value being used in the target clause.

Default Tablespace

Specify a default tablespace (segment, filegroup, or dbspace) for creating new indexes, based on one of the following:

Table Create an index in the same tablespace (segment, filegroup, or dbspace) as the owning table.

Source
Create an index in the same tablespace (segment, filegroup, or dbspace) as the index referenced in the Source File.

Explicit
Create an index in a particular tablespace (segment, filegroup, or dbspace). If you select this option, you must specify the appropriate default. If you select <Default>, the default set in the database is used.

Buffer Pool

The buffer pool (e.g., BP1) that is to be used when creating an Index. You can enter a specific value for the buffer pool or select a value from the list. (**Buffer Pool** is displayed only for DB2 MVS.)

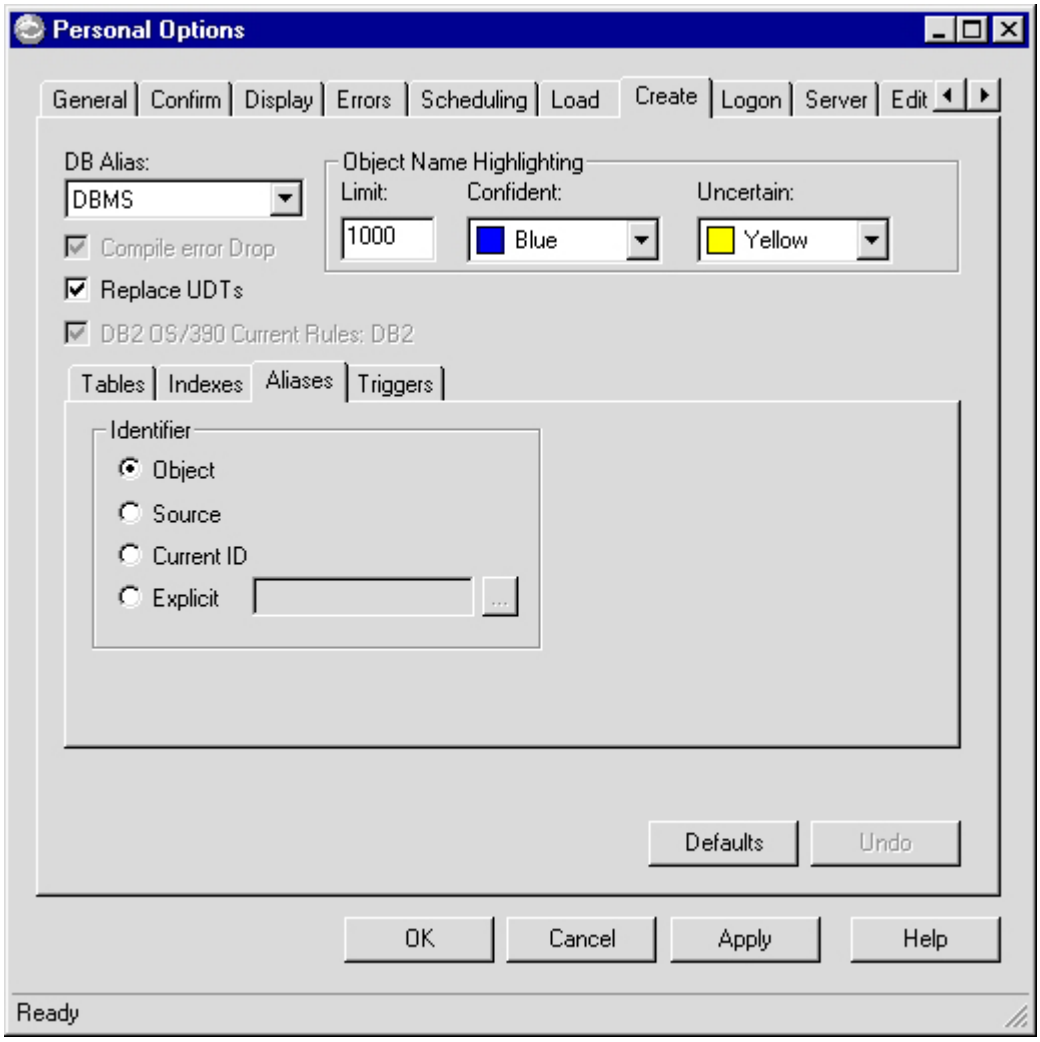
The list displays any index buffer pools already specified for this DB Alias (i.e., on the **Index Defaults** tab of the DB Alias Editor), as well as the following:

<DEFAULT>
Select to use the default buffer pool specified by DB2 MVS. When creating an index, Optim does not generate a BUFFERPOOL clause in the Create statement.

<SOURCE>
Select to use the same buffer pool as the index for the source Archive or Extract File.

Create — Aliases Tab

Use the **Aliases** tab to select default options for creating new aliases. You can specify a default alias when you use DB2 CS, DB2 UDB, or DB2 MVS.



Identifier

Specify the default identifier for new aliases based on the identifier from one of the following:

Object
Use the identifier from the corresponding target object as the default for new aliases. For aliases, the corresponding target object is the table, view, or alias referenced in the alias.

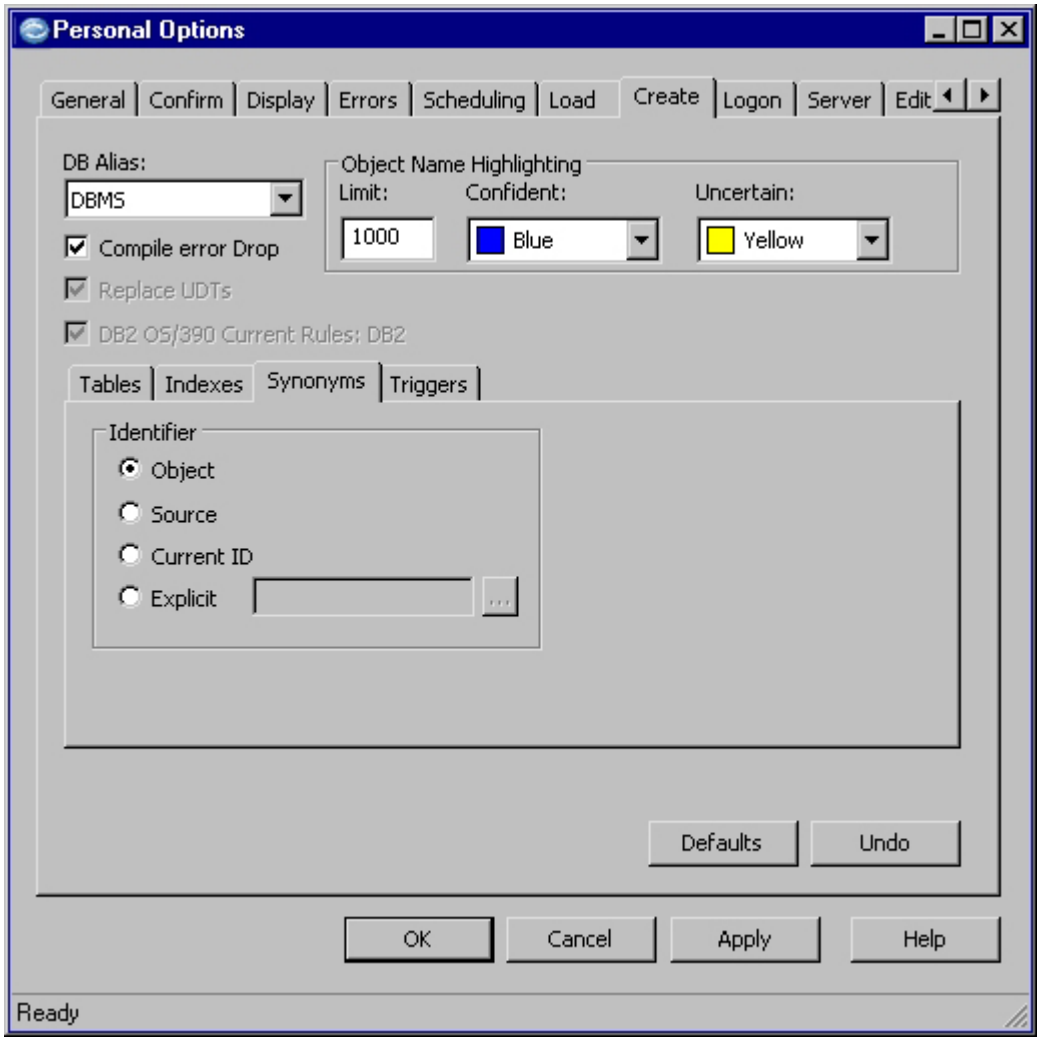
Source
Use the identifier from the source alias as the default for new aliases.

Current ID
Use the current SQLID (User ID) as the default for new aliases.

Explicit
Use an explicit identifier as the default for new aliases. If you select this option, you must specify an explicit identifier (1 to 64 characters). To select from a list, click the browse button.

Create — Synonyms Tab

Use the **Synonyms** tab to select default options for creating new synonyms. You can specify a default synonym when you use Oracle or Informix.



Identifier

Specify the default identifier for new synonyms based on the identifier from one of the following:

Object
Use the identifier from a corresponding target object as the default for new synonyms. For synonyms, the corresponding target object is the table, synonym, function, package, package body, procedure, sequence, trigger, or view referenced in the synonym.

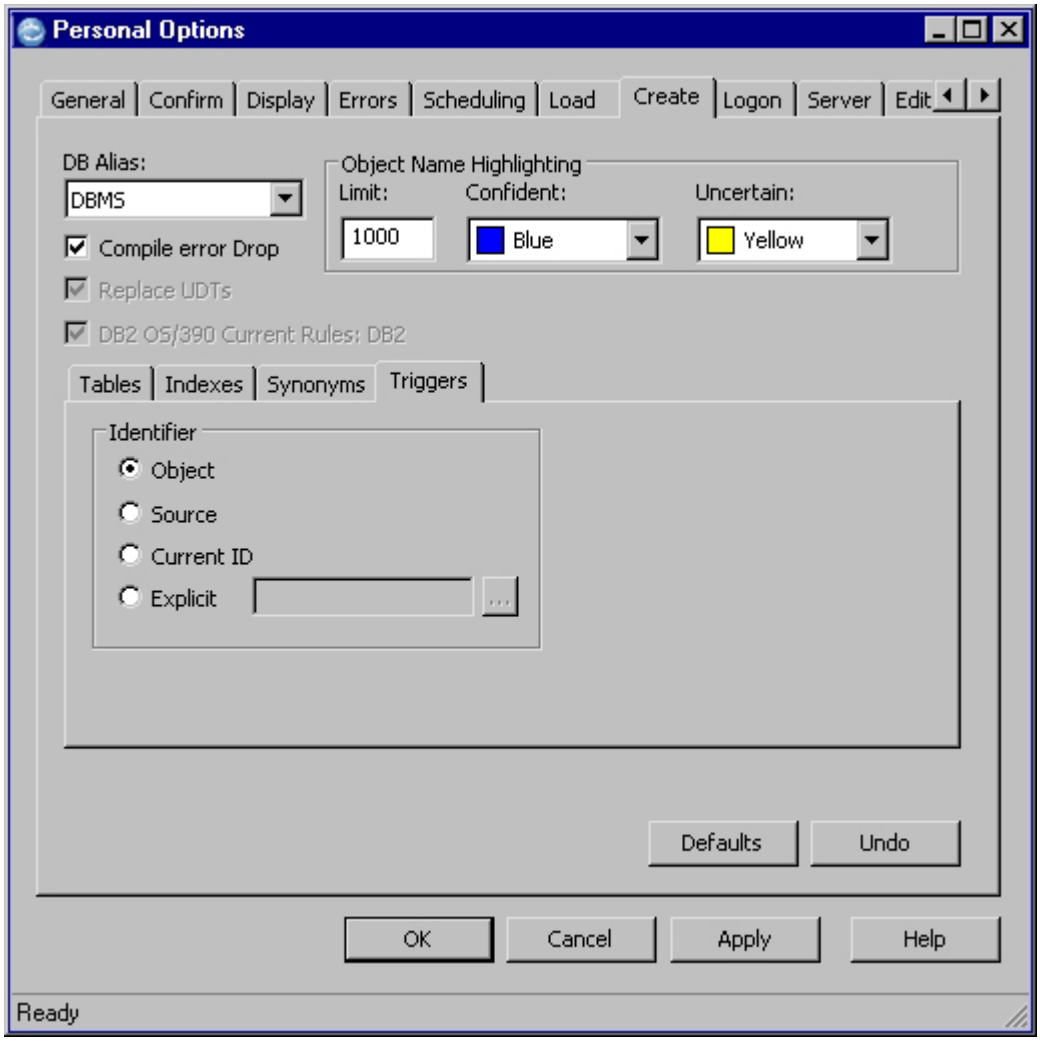
Source
Use the identifier from the source synonym as the default for new synonyms.

Current ID
Use the current SQLID (User ID) as the default for new synonyms.

Explicit
Use an explicit identifier as the default for new synonyms. If you select this option, you must specify an explicit identifier (1 to 64 characters). To select from a list, click the browse button.

Create — Triggers Tab

Use the **Triggers** tab to select default options for creating new triggers. You can specify a default trigger when you use DB2 UDB, Oracle, Sybase ASE, SQL Server, or Informix.



Identifier

Specify the default identifier for new triggers based on the identifier from one of the following:

Object
Use the identifier from a corresponding target object as the default for new triggers. For triggers, the corresponding target object is the table referenced in the trigger.

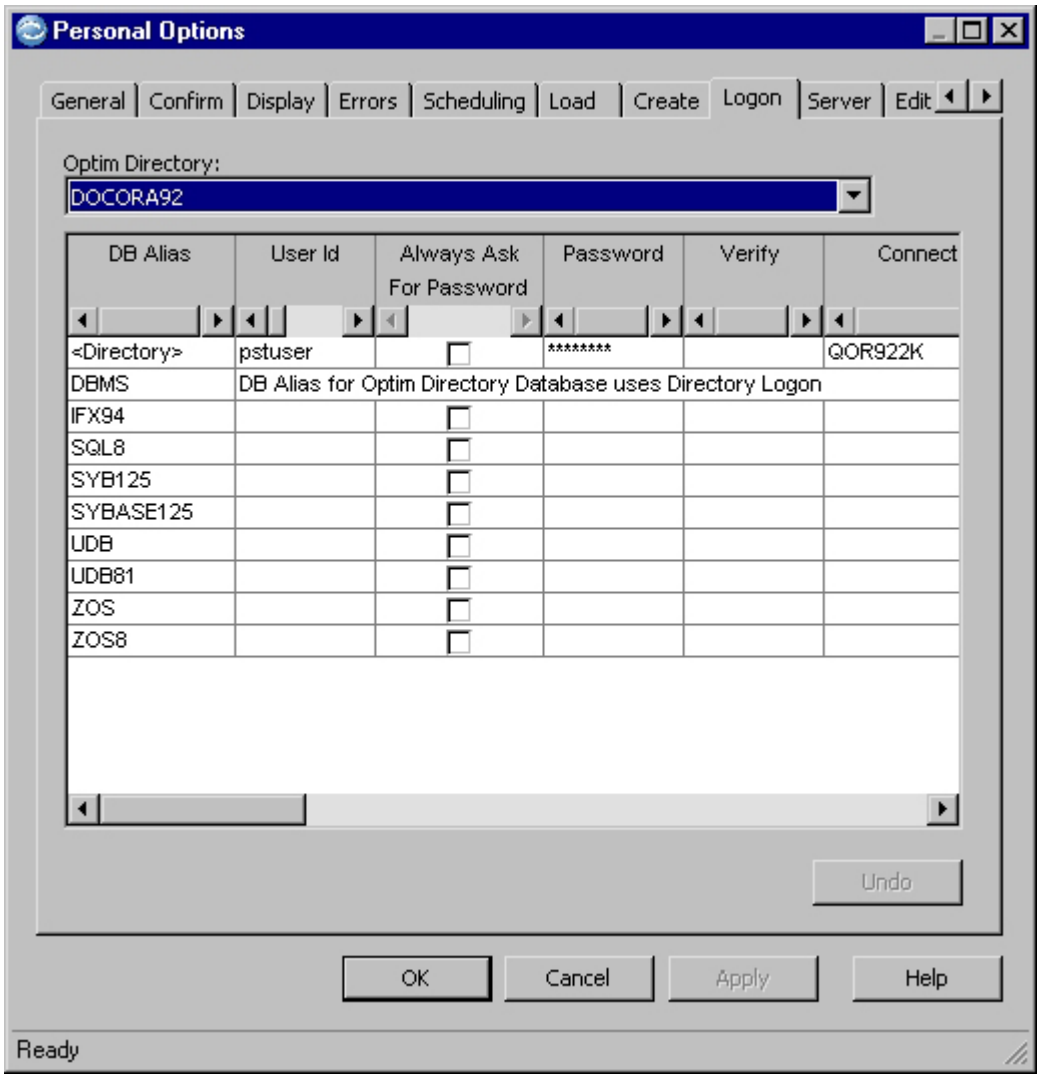
Source
Use the identifier from the source trigger as the default for new triggers.

Current ID
Use the current SQLID (User ID) as the default for new triggers.

Explicit
Use an explicit identifier as the default for new triggers. If you select this option, you must specify an explicit identifier (1 to 64 characters). To select from a list, click the browse button.

Logon Tab

Use the options on the **Logon** tab to set logon and password preferences.



Note: The maximum length for User IDs and Passwords vary depending on the DBMS you are using.

Optim Directory

Select the name of the Optim Directory to display the corresponding logon information. If you have access to more than one Optim Directory, click to select from a list.

Grid Details

The logon information corresponding to the selected Optim Directory includes the following details:

DB Alias
List of DB Aliases.

User ID
Identifier (1 to 30 characters) that allows you to use a DB Alias. User IDs are usually assigned and maintained by the database administrator.

Note: If you are using Informix, specify the User ID in upper case for an ANSI database and in lower case for a non-ANSI database.

Always Ask For Password
Select to display the Logon dialog every time you access a different Optim Directory or DB Alias. If selected, the **Password** and **Verify** entries are not required.

If you clear the check box, the Logon dialog appears the first time you access a different Optim Directory or DB Alias. After you provide a password for a Optim Directory or DB Alias, it is not necessary to provide a password again. (This check box also appears on the Logon dialog.)

Password
Enter a password (1 to 30 characters) that allows you to access a particular database using the specified DB Alias.

Verify Re-enter the password for verification.

Connection String
Connection string Optim uses to access a database using the specified DB Alias.

Always Fail Connection
Select to automatically cancel the logon prompt for a DB Alias that you are not authorized to access or that you choose not to access.

If you clear this check box, a logon dialog is displayed any time you do not have immediate access to a particular DB Alias. (You cannot modify the check box associated with the Optim Directory.)

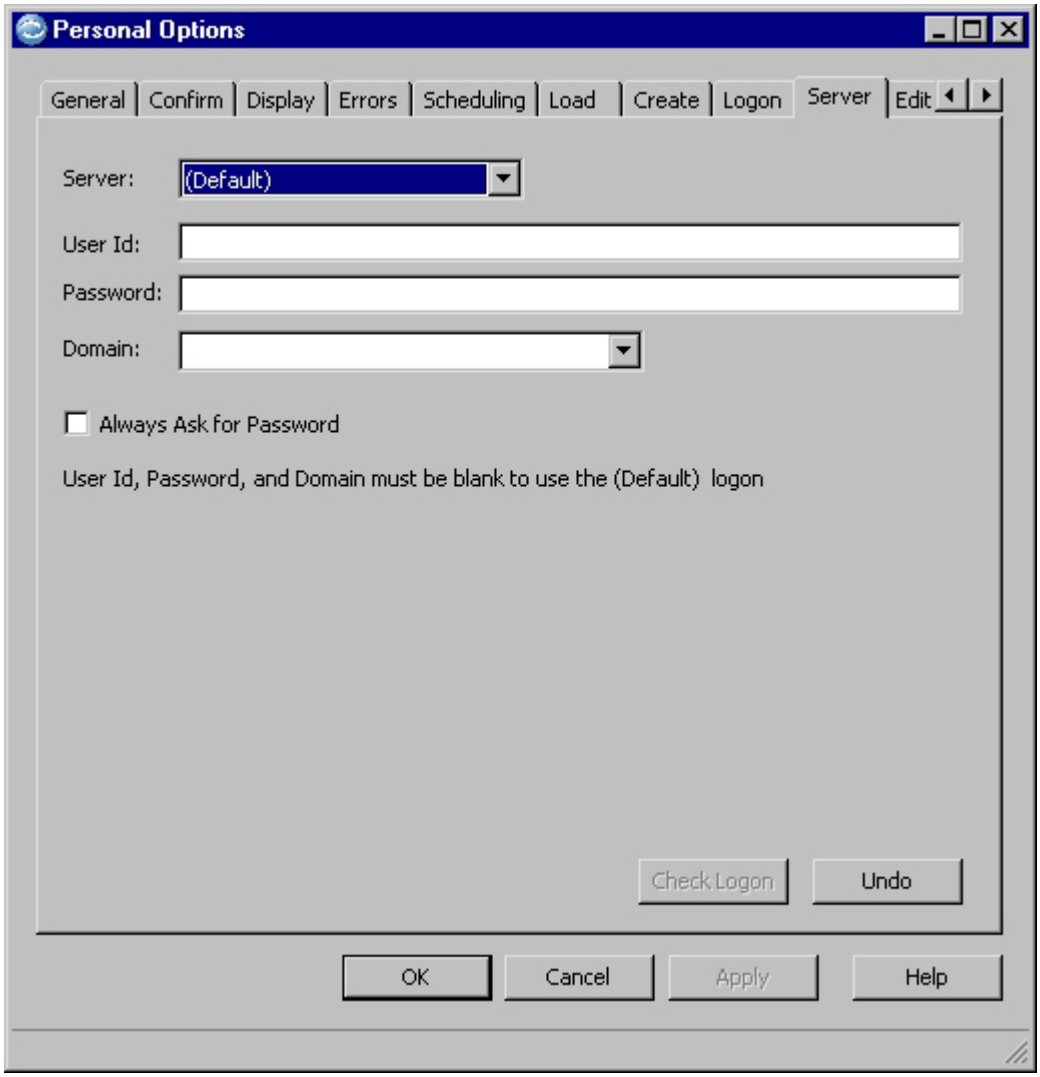
Description
Text that describes or explains the purpose of the logon record.

Test the Connection

You can test the DB Alias connection to verify the validity of the DB Alias logon information. To perform the test, right-click in a grid cell and select **Test Connection** from the shortcut menu. A message displays in the Status bar at the bottom of the dialog indicating the success or failure of the test. If you selected the **Always Ask For Password** check box, you are prompted to enter the password.

Server Tab

Use the **Server** tab to provide credentials that may be used when the optional Optim Server is enabled and tasks are delegated to the Server. The Server can be configured to use these credentials for access to the Optim Directory, certain DB Aliases, or the working files.



Server

Select the name of the Server for which to enter User ID, Password, and Domain information. Click the down arrow to select from the list of Optim Servers configured in Product Options, or select **[Default]** to use the same information for all Servers.

User ID

Enter the User ID used by the Server when performing tasks.

Note: The User ID must have SeBatchLogonRight privileges, or be a member of a "well-known group" with the appropriate authority. This privilege must be granted at the local level for each Server machine.

Password

Enter the password corresponding to the specified User ID.

Domain

Enter the Domain name used to run actions remotely, or for remote input/output files.

Always Ask for Password

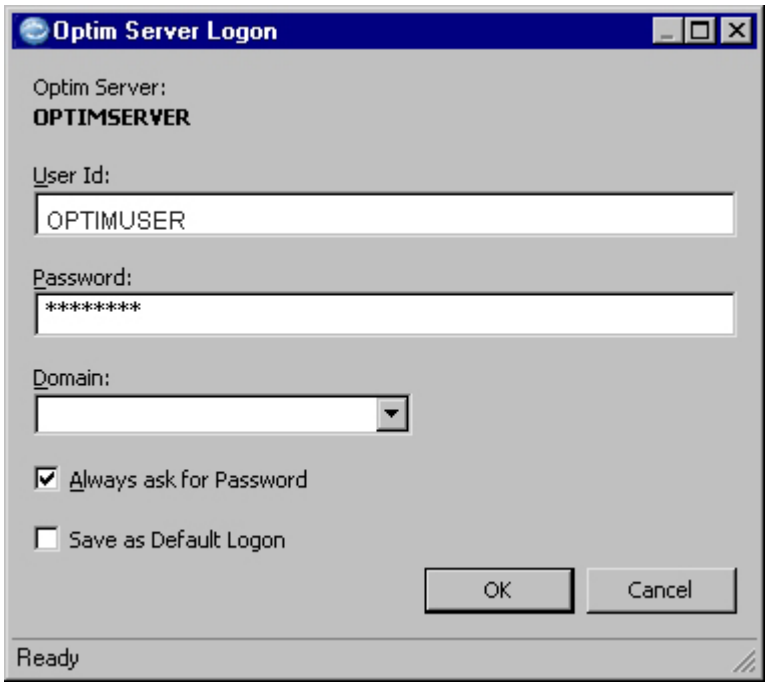
Select to display the Server Logon dialog whenever the Server is used for remote processing calls. When selected, **Password** is not available.

Check Logon

Click **Check Logon** to verify that the Server can log on with the information provided.

Optim Server Logon Dialog

If you select the **Always Ask for Password** check box, or if the default logon information is incorrect, the Optim Server Logon dialog is displayed from a request editor when you specify an Optim Server.



Always Ask for Password

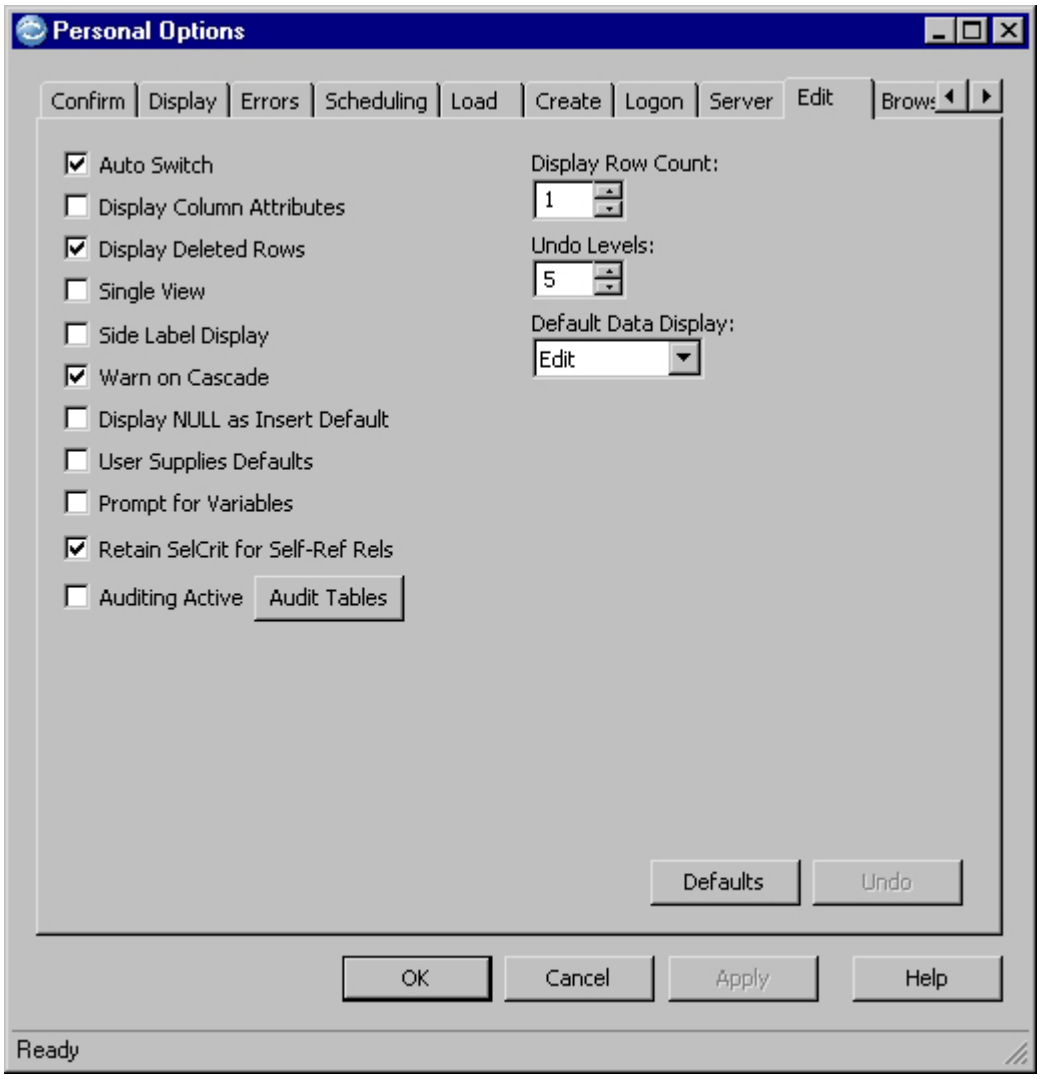
Select to display the Optim Server Logon dialog whenever the **Server** is used for remote processing calls.

Save as Default Logon

If you select the **Save as Default Logon** check box, the information you enter in this dialog overrides the default settings specified on the **Server** tab in Personal Options.

Edit Tab

Use the options on the **Edit** tab to set preferences for browsing and editing data.



Auto Switch

Select to automatically switch subordinate tables in a *stack* of two or more joined tables to display related rows.

When you select a row in a table and no related rows exist in the subordinate table, **Edit** automatically switches to display the next table in the stack that has a related row.

Display Column Attributes

Select to display column attributes (data type, length, and nullable attribute) for all columns in a selected table. Column attributes are useful when you insert a row or modify column data in the Table Editor.

Display Deleted Rows

Select to display rows that you delete (in Deleted status) in the Table Editor. Deleted rows appear dimmed. To hide deleted rows, clear this check box.

Single View

Select to disable the Join capability when the first item in the Table Editor is a view. Browsing and editing is more efficient using single view mode because relationship information is bypassed. However, to browse or edit related data, you must clear the check box.

Side Label Display

Select to show column names and values side by side for a single row. To show column names and values for multiple rows (Columnar Display), clear this check box.

Warn on Cascade

Select to display a warning that rows in other tables may be deleted, or column values set to NULL, when you delete rows in a table. The Delete Confirmation dialog displays the names of affected tables, including tables that are not shown in the Table Editor. Column values may be set to NULL if the relationship between the tables is using the SET NULL delete rule.

Note: Be certain you want to disable this feature before clearing this check box.

Display NULL as Insert Default

Select to specify NULL as the default value for nullable columns when you insert a new row. If you clear this check box, **Edit** provides a value based on the column data type. Other than NULL, possible values include blank, zero, current date, current time, and current timestamp. To specify the character for the NULL value indicator, use the **Display** tab on the Personal Options dialog.

Note: Site management may set Product Options to restrict the use of this feature.

User Supplies Defaults

Select to require a user-supplied value for every column that cannot accept a default value. If you clear this check box, **Edit** provides a value based on the column data type. Possible values include blank, zero, current date, current time, or current timestamp.

Note: Site management may set Product Options to restrict use of this feature.

Prompt for Variables

Select to request a prompt for default values associated with substitution variables in an Access Definition. You can use substitution variables in selection criteria to specify data to browse or edit.

Retain SelCrit for Self-Ref Rels

Select to apply selection criteria each time a table is self-referenced in the Table Editor. Clear the check box to ignore selection criteria when a table is self-referenced. The default value can be changed on the Specify Edit Preferences dialog.

Note: A table can be self-referenced only when the Table Editor is in Browse mode.

Auditing Active

Select to activate the Audit option for tracking database changes when you edit data. If you select this option, click **Audit Tables** to specify the tables you want to audit. If you prefer not to use this option, clear the check box.

Note: Site management may set Product Options to restrict the use of this option.

Audit Tables

Click to open the Audit Tables dialog on which you can specify a list of tables to audit.

Note: Auditing is available in Personal Options only if the **Auditing Status** in Product Options is set to **Active/User**.

Display Row Count

Specify the default number of rows to display for each joined table in the Table Editor. Use row count to manage the display area when you join several tables in the Table Editor.

Undo Levels

Specify the default number of times (1 to 20) you can undo a commit to any row in the Table Editor. You can specify the maximum number of undo levels (or commit points) per row.

Use undo commands to restore data in the Table Editor to a prior commit point. For example, if you change three columns in a row and commit that row, you can undo the changes using one undo level. If you set Undo Levels to 5 and you commit 7 times on a particular row, you can undo only the last 5 committed changes to that row or return to the original row.

Default Data Display

Select the default mode for adding tables to the Table Editor. Click the down arrow to select **Browse**, **Browse Only**, or **Edit** mode for each table opened or joined in the Table Editor.

In **Browse** or **Edit** mode, a table can appear only once in the Table Editor. In **Browse Only** mode, a table can appear more than once in the Table Editor.

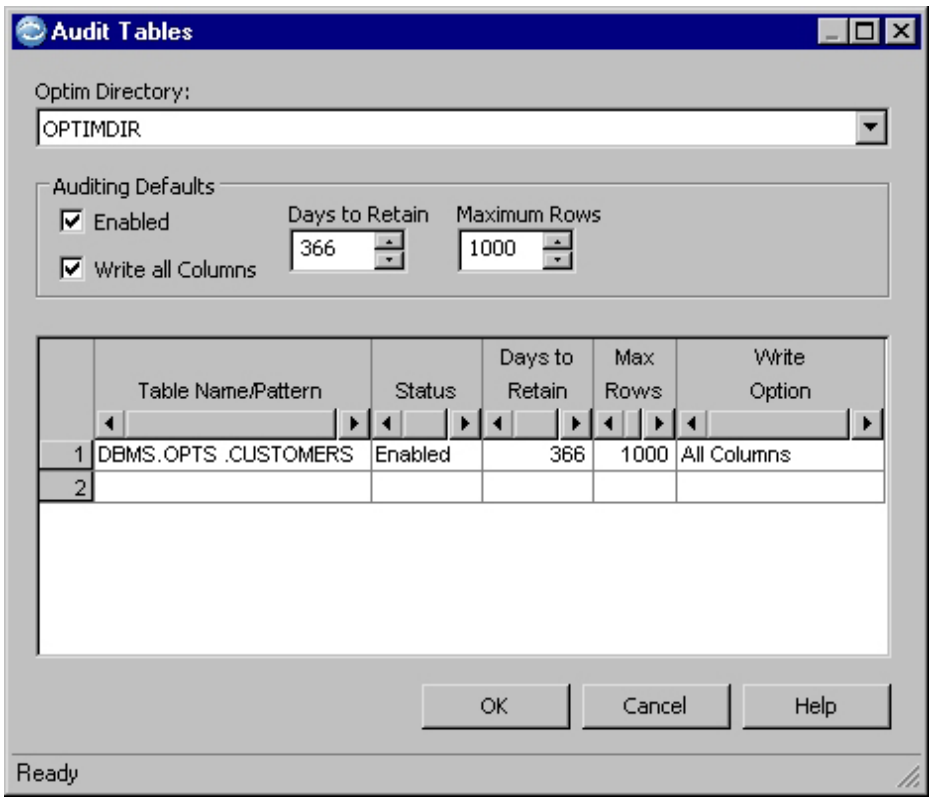
You can switch from **Browse Only** mode to **Browse** or **Edit** mode by unjoining any duplicate tables, selecting **Preferences** from the **Tools** menu on the Table Editor, and selecting **Browse** or **Edit** mode.

Note: If the **Force Browse Only** check box on the **Edit** tab in Product Options is selected, the controls pertaining to editing data are unavailable.

Audit Tables Dialog

If you select the **Auditing Active** check box on the **Edit** tab, click **Audit Tables** to display the Audit Tables dialog. You can specify a Personal Options list of tables to audit.

Note: Auditing is available in Personal Options only if the **Auditing Status** in Product Options is set to **Active/User**. (See the *Installation and Configuration Guide* .)



Optim Directory

Select the Optim Directory associated with the tables to audit. If you have access to more than one Optim Directory and want to specify the tables to audit for those directories, click the down arrow.

Audit results are stored in the PSTAUDIT table, which is one of the Optim Directory tables created when you install Optim. If you are authorized, you can browse or edit the PSTAUDIT table in the same way you browse or edit any other database table. However, **Auditing Status** in Product Options must be set to **Active/User**, and you must have database SELECT authority for the PSTAUDIT table.

You can specify a Personal Options list of tables to audit. However, the list specified in Product Options takes precedence over the list that you specify in Personal Options.

Table Name/Pattern
Enter the name of the database table or pattern that identifies the tables to audit. Table names consist of *dbalias.creatorid.tablename*.

When you specify a pattern for like-named tables, you can use percent (%) to represent one or more characters. Use underscore (_) to represent a single character.

Note: An option on the **General** tab in Personal Options allows you to use the underscore as an SQL LIKE character.

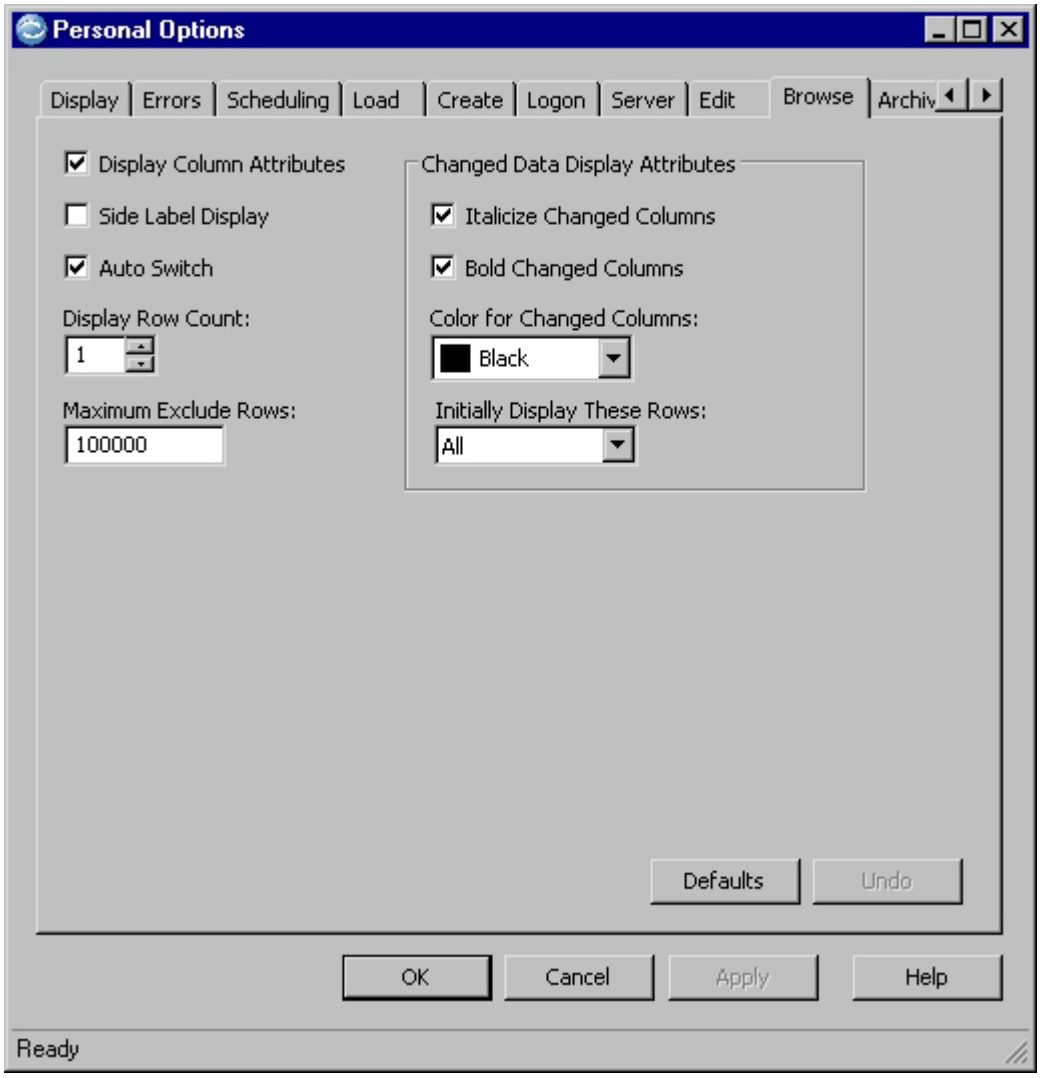
Status Enable or disable the Audit option for individual tables. Click to display a down arrow and select **Enabled** or **Disabled** for each table in the list.

If the status indicates **Superseded by Product List**, the table is ignored because of a conflict with the parameters set in Product Options. You cannot enable or disable the audit option for that table, but you can modify the table name or remove it from the list.

The Edit Process audits tables based on a number of specific parameters, beginning with the parameters specified on the **Edit** tab in the Product Options dialog.

Browse Tab

Use the options on the **Browse** tab to set preferences for browsing data and to select font characteristics for browsing a Compare File to quickly identify changed data.



Display Column Attributes

Select to display the data type, length, and nullable attributes for all columns in a selected table when browsing an Extract File, Archive File, or a Compare File. To display only column names, clear the check box.

Side Label Display

Select to display rows one at a time, with column names and values side-by-side. To display multiple rows in a columnar format, clear this check box.

Auto Switch

Select to automatically switch subordinate tables in a stack of two or more joined tables to display related rows.

When you select a row in a table and there are no related rows in the subordinate table, the display is automatically switched to the next table in the stack that has a related row.

Display Row Count

Specify the default number of rows to display for each joined table. Use row count to manage the display area when you join several tables.

Maximum Exclude Rows

Specify a row limit to improve performance when browsing an Extract, Archive, or Control File that contains a large number of rows. When browsing, the **Exclude Rows** and **Only Show Rows in Error** features are unavailable for tables that exceed the specified row limit. (When an Extract, Archive, or Control File is first opened for browsing, system resources are allocated for creating a cache for temporary storage of excluded rows and rows in error. Therefore, browsing a very large file can consume a large amount of system resources.) This limit is reevaluated when a table is joined to another table, because the resulting subset may contain less rows.

The default value is 100,000 rows.

When a table is browsed that contains rows in excess of the Maximum **Exclude Rows** value, a message displays to remind you of the specified limit.

Note: If you never expect to use the **Exclude Rows** or **Only Show Rows in Error** features, set this limit very low to optimize system performance.

Changed Data Display Attributes

Select options to identify changed data when browsing a Compare File.

Italicize Changed Columns
Select to *italicize* data that differs between Source 1 and Source 2.

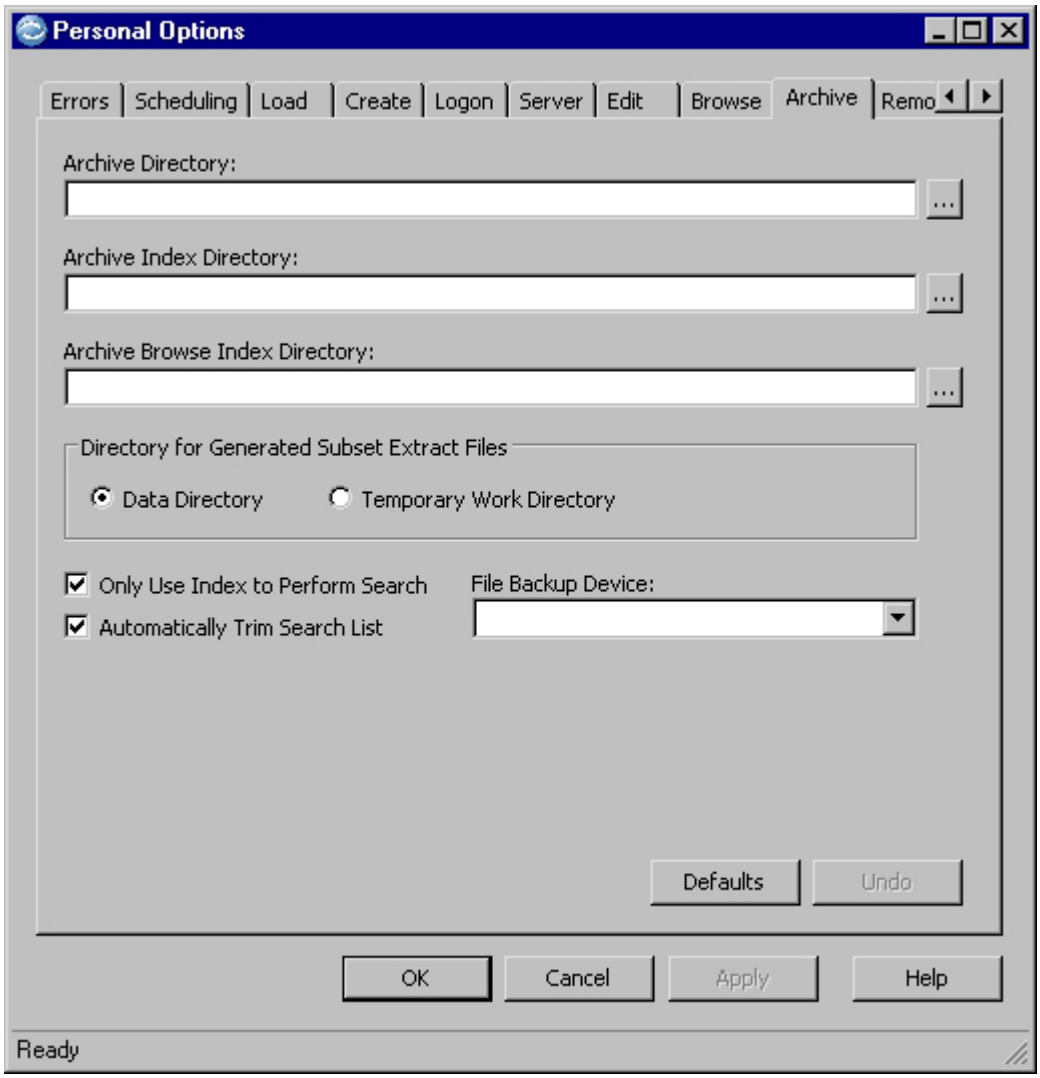
Bold Changed Columns
Select to display the data that differs between Source 1 and Source 2 in **bold** type.

Color for Changed Columns
Click the down arrow to select a color to identify data that differs between Source 1 and Source 2.

Initially Display These Rows
Select rows to display by default when browsing a Compare File: All, Different, Duplicate, Equal, or Only (unmatched rows).

Archive Tab

Use the options on the **Archive** tab to set preferences for an Archive Process.



Archive Directory

Specify the complete path to the default directory where you want to store Archive Files. To select from your system directories, click the browse button. If you do not specify a directory, the Data Directory specified on the **General** tab is used by default.

Archive Index Directory

Specify the complete path to the default directory where you want to store Archive Index Files. To select from your system directories, click the browse button. If you do not specify a directory, the Archive directory is used by default.

Archive Browse Index Directory

Specify the complete path to the default directory where you want to store Archive Index Browse Files. To select from your system directories, click the browse button. If you do not specify a directory, the Archive directory is used by default.

An Archive Index Browse File is created automatically whenever you join tables while browsing an Archive File. The Archive Index Browse File stores primary key and foreign key information to expedite the retrieval of data, and has an *.abf* extension, by default. Archive Index Browse Files are dynamically updated. For this reason, it is advisable to select a directory accessible to any user that may browse an Archive File.

Directory for Generated Subset Extract Files

Select either **Data Directory** or **Temporary Work Directory** to specify the location in which to store automatically generated subset Extract Files.

Note: Both directories are specified on the **General** tab.

Only Use Index to Perform Search

When search criteria is used to locate Archive Files containing specific data, Archive initially searches for matching rows in the corresponding Archive Index Files. If a match for the search criteria cannot be determined from the Archive Index information (or index information is not defined), Archive must check the Archive File to resolve the search. This check box is cleared by default to direct the Archive Process to automatically search Archive Files when a match cannot be determined from Archive Index Files.

When you select this check box, you direct the Archive Process to search Archive Index Files only. Use shortcut menu commands **Resolve** and **Resolve All** to complete the search when a match cannot be determined from Archive Index information.

Automatically Trim Search List

Select to automatically exclude all files other than possible matches when you use the Search command to locate and display Archive File names that contain specific data.

File Backup Device

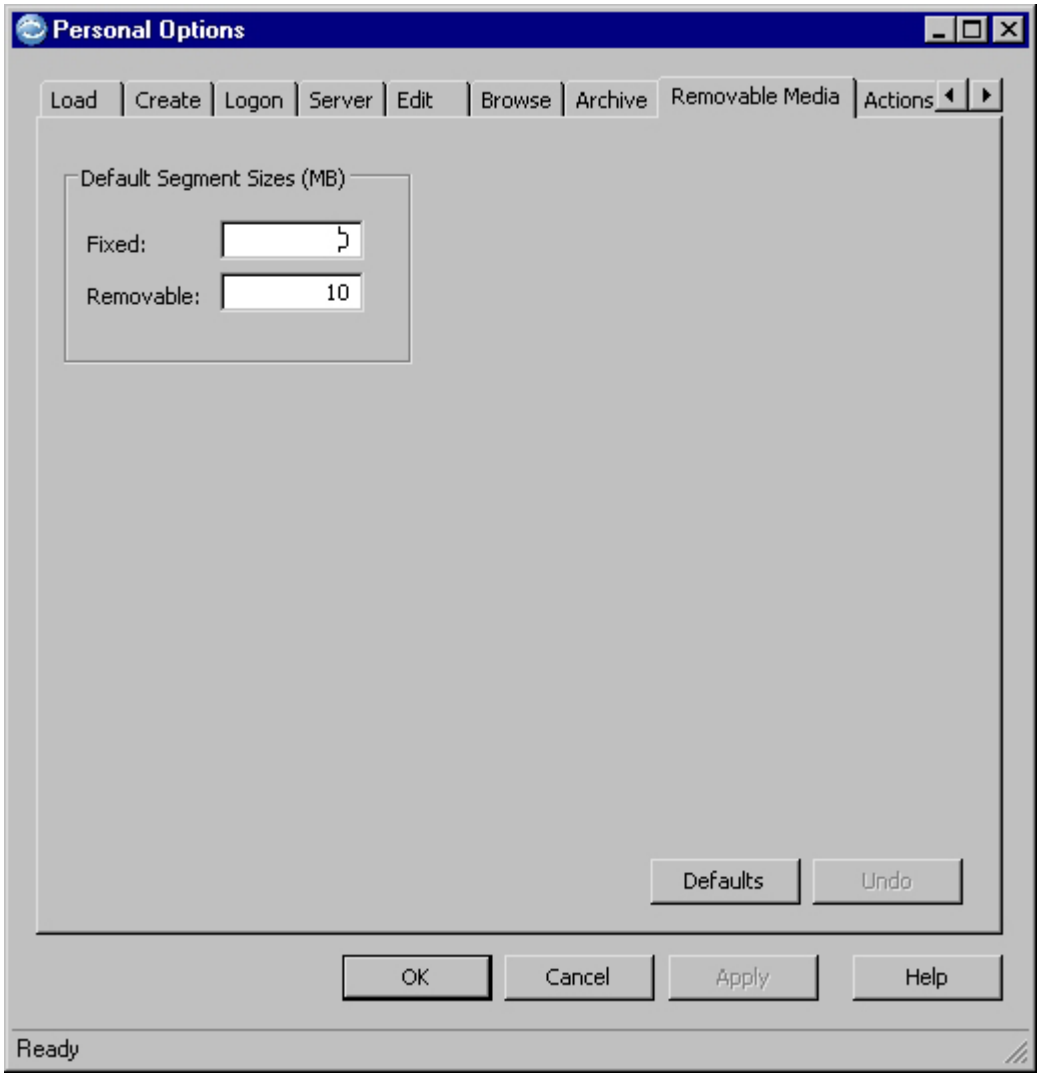
Click the down arrow to select the default backup device from a drop⌂down list of available backup devices.

Note: The drop-down list includes the backup devices selected on the **Archive** tab in Product Options (see the *Installation and Configuration Guide*). If no backup devices are selected in Product Options, this option is unavailable.

The default backup device is automatically selected when you create a new Storage Profile.

Removable Media Tab

Use options on the **Removable Media** tab to set default values for segment size when creating Archive Files, whether or not using a Storage Profile Definition, and Extract Files. When an Archive or Extract File is larger than the space limitation for the target media, the file must be segmented to span more than one volume.



Default Segment Sizes (MB)

Fixed Enter the segment size (0 - 9999 MB) to use when the target destination is a fixed drive (i.e. hard disk). To specify no limit, enter a value of 0 (zero).

Note:

- This value also applies to Archive Files copied to a backup device, because Archive Files are created on disk before being copied to a backup device.
- The maximum segment size when the target is a Centera Server is 2 GB.

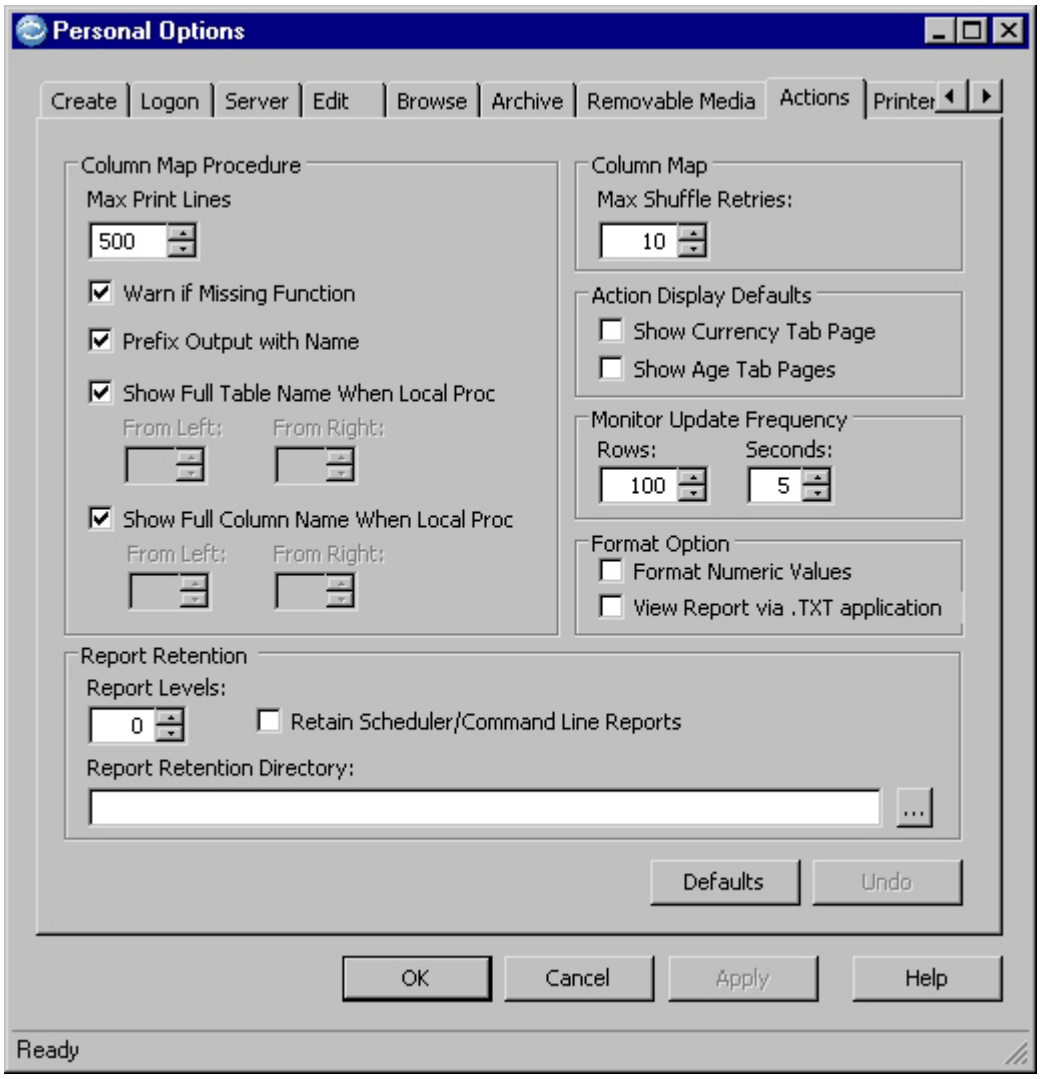
Removable
Enter the segment size (1 - 9999 MB) to use when the target destination is a removable device (e.g., zip drive).

Note: The naming convention for a segmented file is: *filename_1.ext,filename_2.ext, (...filename_n.ext), filename.ext*. The name of the last segment is the file name specified in the process request. For example, when creating an Archive File named c:\arch\archtest.af that requires three segments, the segments are named as follows:

```
c:\arch\archtest_1.af (segment 1)
c:\arch\archtest_2.af (segment 2)
c:\arch\archtest.af (segment 3)
```

Actions Tab

Use the options on the **Actions** tab to select preferences for printing Column Map procedures, displaying tabs in action request editors, updating the Progress dialog, formatting numeric values, and retaining process reports.



Column Map Procedure

Max Print Lines

Specify the maximum number of lines to route to a process report. If the number of lines exceeds this maximum, a warning message indicates the output is incomplete.

Warn if Missing Function

Clear this check box to suppress the warning message generated in the process report when a Load, CmStartTable, CmEndTable, or UnLoad function is omitted from a Column Map Procedure.

Note: The CmTransform function must be included in a Column Map Procedure.

Prefix Output with Name

Select the check box to include the name of the Column Map procedure with the print output (default).

Note: When you choose to include the name of the Column Map procedure with the print output, and a Local (i.e. unnamed) Column Map procedure is used, a name for the Local Column Map procedure is automatically generated. The name is generated using the corresponding table name, column name, and a unique number as follows: ***tablename.columnname.n***

The following options allow you to modify parts of the generated name for a Local Column Map procedure (the full table name and column name are used by default).

Show Full Table Name When Local Proc
Select this check box to include the full table name in the generated Local Column Map procedure name.

If you clear this check box, use the **From Left** and **From Right** controls to specify a subset of the table name. **From Left** indicates the number of bytes to use from the beginning of the table name. **From Right** indicates the number of bytes to use from the end of the table name. (For example, if the table name is CUSTOMERS, and you specify 4 for **From Left** and 2 for **From Right**, the subset of the table name used is CUSTRS.)

Show Full Column Name When Local Proc
Select this check box to include the full column name in the generated Local Column Map procedure name.

If you clear this check box, use the **From Left** and **From Right** controls to specify a subset of the column name. **From Left** indicates the number of bytes to use from the beginning of the column name. **From Right** indicates the number of bytes to use from the end of the column name. (For example, if the column name is SALESMAN_ID, and you specify 0 for **From Left** and 2 for **From Right**, the subset of the column name used is ID.)

Note: You can specify 0 for **From Left** and 0 for **From Right** to indicate that no part of the name is used. However, you must use part of either the table name or the column name.

Column Map

Max Shuffle Retries
Default number of times the Column Map Shuffle Function will refetch a replacement value until a value that does not match the source row is found (a "retry"). The Shuffle Function **retry** parameter overrides this default.

Enter a value from 0-1000. Enter 0 to allow a replacement value to match the source. The default is 10.

Note: Using a high retry value with columns that contain many duplicate values will increase the processing time. For these columns, it may be best to use a retry value of zero.

Action Display Defaults

Show Currency Tab Page
Select to display the **Currency** tab in the Convert, Insert, and Load Request Editors.

Show Age Tab Pages
Select to display the **Age Function** and **Global Aging** tabs in the Convert, Insert, and Load Request Editors.

Note: You can override these selections with commands available from the **Options** menu in each action request editor.

Monitor Update Frequency

Allows you to specify the frequency with which a progress dialog is updated during processing. (Note that increasing this value may enhance performance.)

Rows Specify the number of rows (100 to 5000) to process before updating the status message on the progress dialog. The default value is 100.

Seconds
Specify the number of seconds (5 to 60) to pass before updating the process time on the progress dialog. The default value is 5.

Format Options

Format Numeric Values
Select this check box to format numeric values displayed on progress dialogs and in process reports for all actions. Clear this check box to display numeric values without formatting (e.g., 99888).

For example, if you select this check box and run the Extract Process, the Extract Request Progress dialog would display 99,888 for the total number of rows extracted, depending on the numeric format defined for Windows.

To view the numeric format for your workstation, select **Regional Options** from the Control Panel and review the **Numbers** tab.

View Report via .TXT application
This check box is selected by default. It formats the report as a .TXT file and opens automatically using an application such as Notepad.

Report Retention

Report Levels
Specify the maximum number of reports you can retain for each type of process. You can specify a value from 0 through 200. A value of 0 (default) disables the report retention feature.

When the number of retained reports for a particular type of process exceeds the maximum, the oldest report is deleted and the current report is saved.

Retain Scheduler/ Command Line Reports
Select to retain reports generated by processes invoked using the **Scheduler** or the Command Line Interface. This check box is cleared by default.

Report Retention Directory
Specify the complete path to the default directory in which you want to store reports. Leave blank (default) to use the Temporary Work Directory specified on the **General** tab. To select from your system directories, click the browse button.

Note: It is recommended that each user specify a private directory for storing reports.

Printer Tab

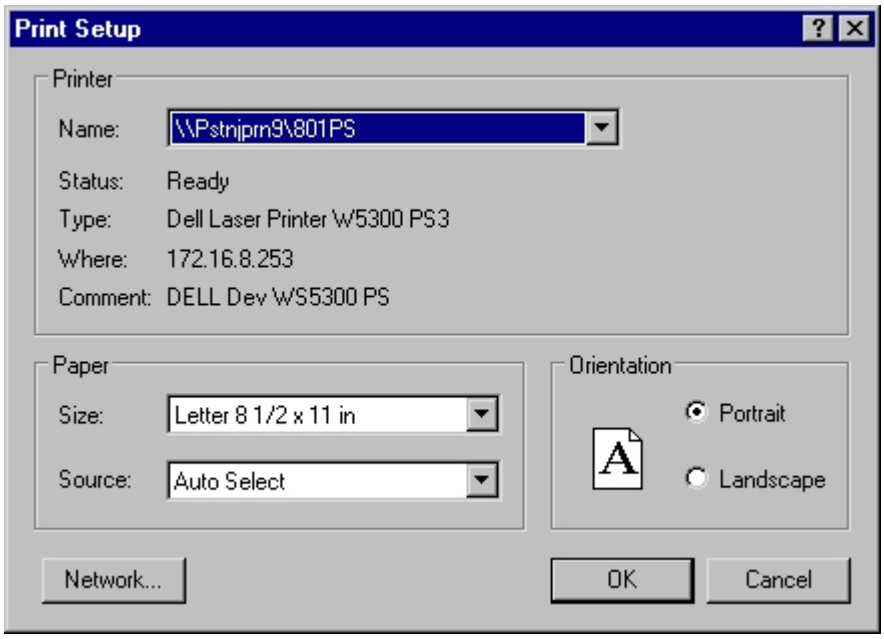
Use the **Printer** tab to set printer, font, and language preferences for printing. Note that font and language settings appropriate to the character set used for a report will ensure that text prints correctly if the Local settings for your computer do not match settings for the computer used to create the report.

The default printer and font information are shown in each of the message boxes. To open the Windows Print Setup dialog to select a printer, click **Set Printer**. To open the Windows Font dialog to select font attributes and specify the desired language script, click **Set Font**.

Printer

Set Printer

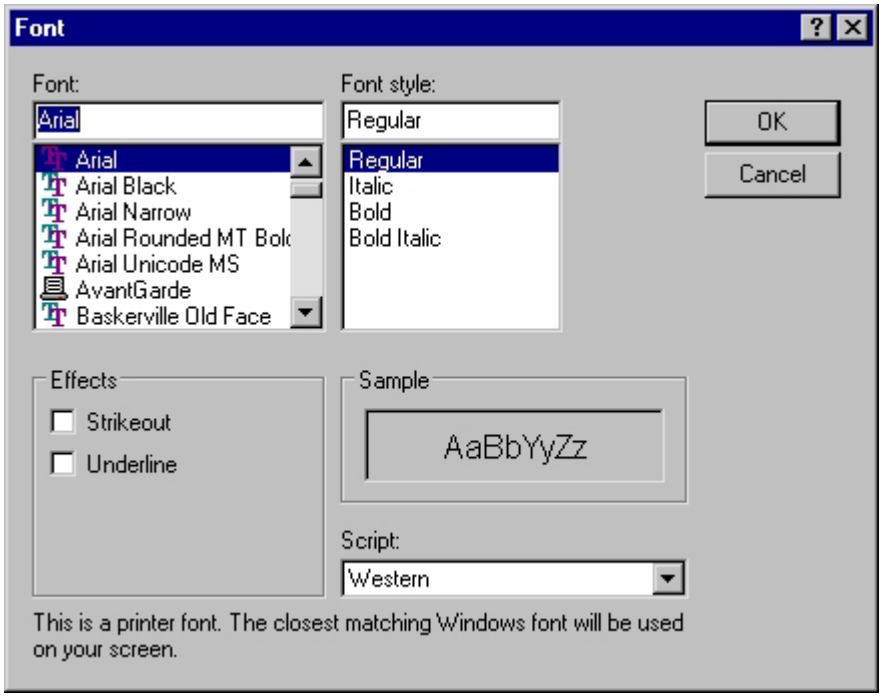
Click to display the Print Setup dialog. Select a printer to use as the default.



Field Font

Set Font

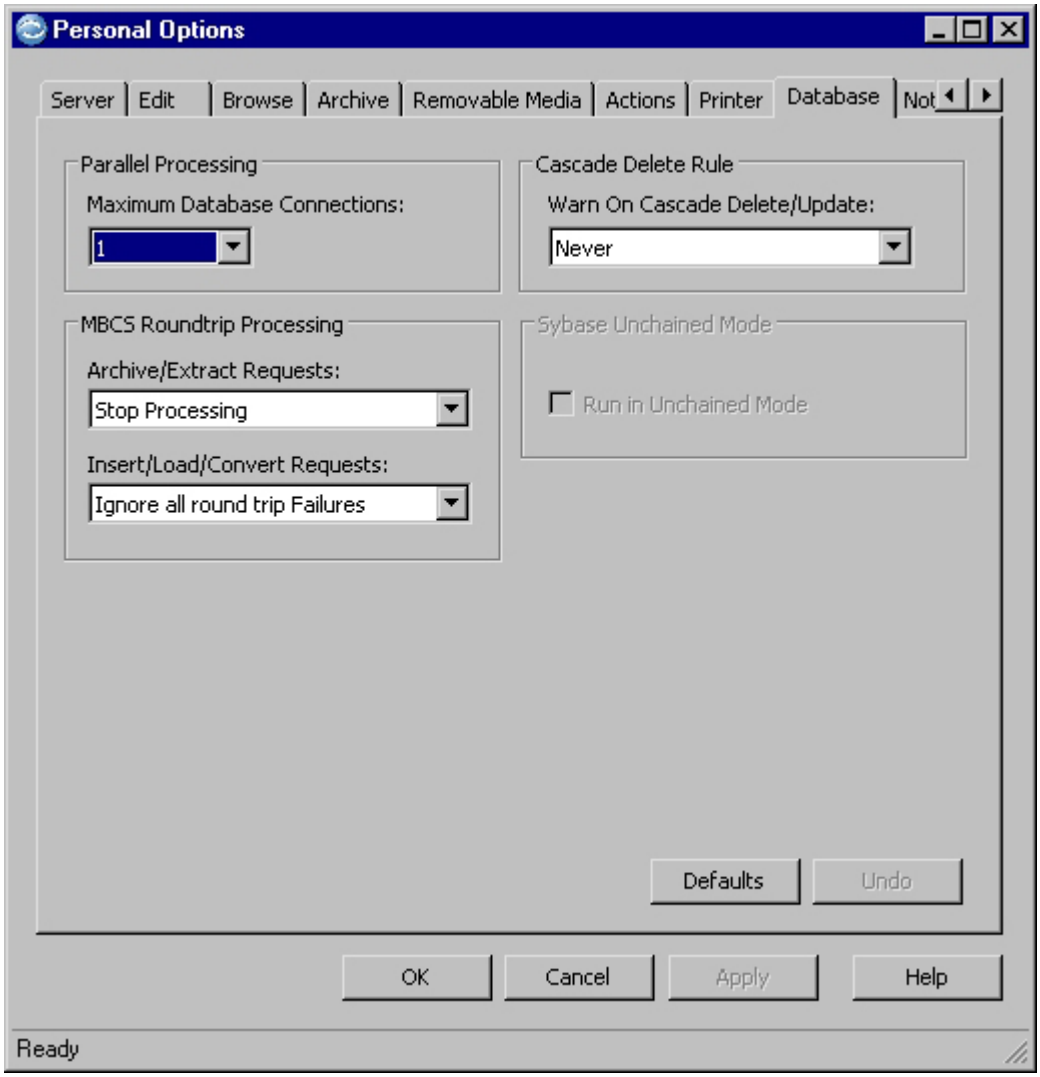
Click to display the Font dialog. Select font characteristics and a language script to use.



Note: The available languages in the **Script** drop-down list are determined by the selected font.

Database Tab

Use the options on the **Database** tab to set preferences for handling multi-byte round trip conversion errors, cascade deletes, and database connections. For Sybase ASE, you can also specify whether to run in Unchained mode.



Parallel Processing

Maximum Database Connections

Specify the default number of concurrent database connections for an Archive, Delete, or Extract Process. Increasing database connections improves performance when processing large quantities of data by allowing multiple threads to process rows in parallel.

To increase the maximum number of connections, select an even number from 2 through the site maximum as specified in Product Options.

Note: For performance reasons, you can only select an even number of maximum database connections.

MBCS Roundtrip Processing

Options for handling characters that could cause round trip conversion issues in a multi-byte Optim Directory or DB Alias. The availability of these options is governed by the **MBCS Roundtrip Processing** settings on the Product Options **Database** tab.

Optim uses the Unicode character set in dialogs and to process data. In some multi-byte character sets (such as Oracle JA16SJIS), multiple characters are mapped to the same Unicode character and may not be

converted correctly into Unicode. When these characters are converted from Unicode back to multi-byte (a round trip), the original character may not be returned.

Archive/Extract Requests
Select an option for handling round trip conversion issues during Archive or Extract processing:

Stop Processing
Stop processing when a multi-byte character is encountered that could cause an incorrect round trip conversion.

Ignore all round trip Failures
Continue processing when a multi-byte character is encountered that could cause an incorrect round trip conversion. (Default.)

Insert/Load/Convert Requests
Select an option for handling round trip conversion issues during Insert, Load, or Convert processing:

Stop Processing
Stop processing when a multi-byte character is encountered that could cause an incorrect round trip conversion.

Ignore all round trip Failures
Continue processing when a multi-byte character is encountered that could cause an incorrect round trip conversion. (Default.)

Select the **Ignore all round trip Failures** option if the database does not contain data with characters that could cause round trip errors or if columns used to manipulate data in a Column Map (e.g., a function is used) and columns for which selection criteria are defined do not contain characters that could cause round trip errors.

Cascade Delete Rule

Warn On Cascade Delete/Update
Display a warning if a cascading delete or update may occur to a table that is not explicitly included in an Access Definition or a process.

Runtime
Display a cascade delete/update warning only at run time of a process.

Saving Access Definition
Display a cascade delete/update warning only when saving the Access Definition.

Always
Display a cascade delete/update warning at run time of a process and when saving the Access Definition.

Never Do not display a cascade delete/update warning. This is the default setting.

The **Warn on Cascade Delete/Update** setting in Product Options affects the availability of this option for user input. If available, you can click the down arrow to choose when to display a warning. If unavailable, the value specified for **Warn on Cascade Delete/Update** in Product Options is displayed and cannot be modified. (Refer to the *Installation and Configuration Guide* .)

Sybase Unchained Mode

Run in Unchained Mode
Optim normally runs in chained mode. When a trigger in a Sybase ASE table will be fired as a result of an Insert or Delete Process, and the trigger calls a stored procedure that must run in unchained mode, the connection must be in unchained mode for the procedure to work.

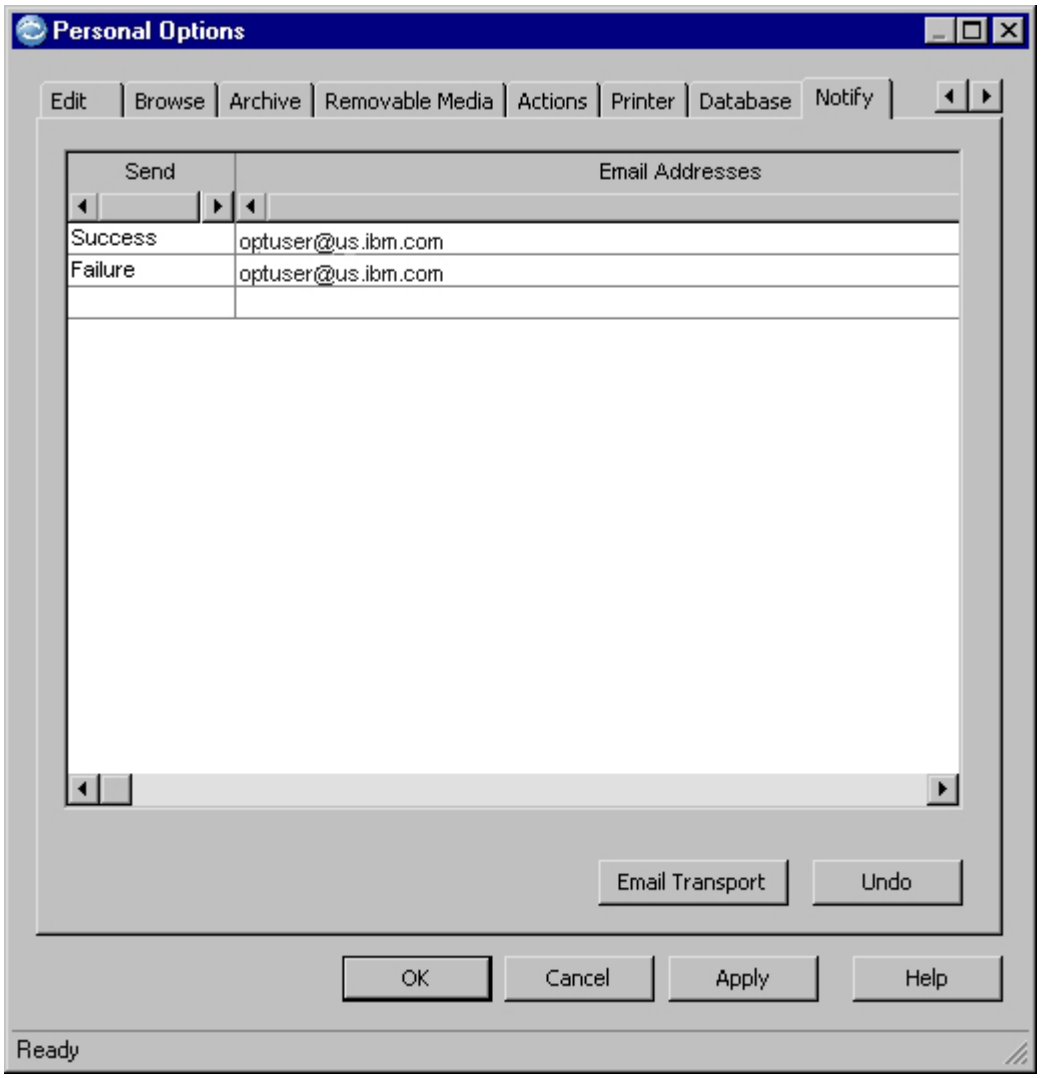
The **Sybase Unchained Mode** setting in Product Options enables or disables this check box. If enabled, select the check box to run Insert and Delete Processes in Unchained Mode, or clear the check box to run all actions in normal mode. (See the *Installation and Configuration Guide* .)

Notify Tab

Use the **Notify** tab to specify default options and addresses for automatic email notification of the success or failure of a process. The process report generated when a process completes is automatically sent as an attachment.

Note: Before using email notification, the desired email program must be installed. For Windows, the email client must be defined as the default, and set up to interface with MAPI. For UNIX or Linux, a valid copy of SENDMAIL must be configured correctly.

In an action request editor, you can click **Get Site Defaults** on the **Notify** tab to populate it with the defaults specified on the **Notify** tab in Personal Options.



Grid Details

The **Notify** tab contains the following details:

Send For each email address you list, click to select an option to send a message as determined by the outcome of the process. You can select **Always**, **Success**, or **Failure**.

Email Addresses
Enter an email address to which notification is sent with a process report at the completion of the process. Enter one address per line.

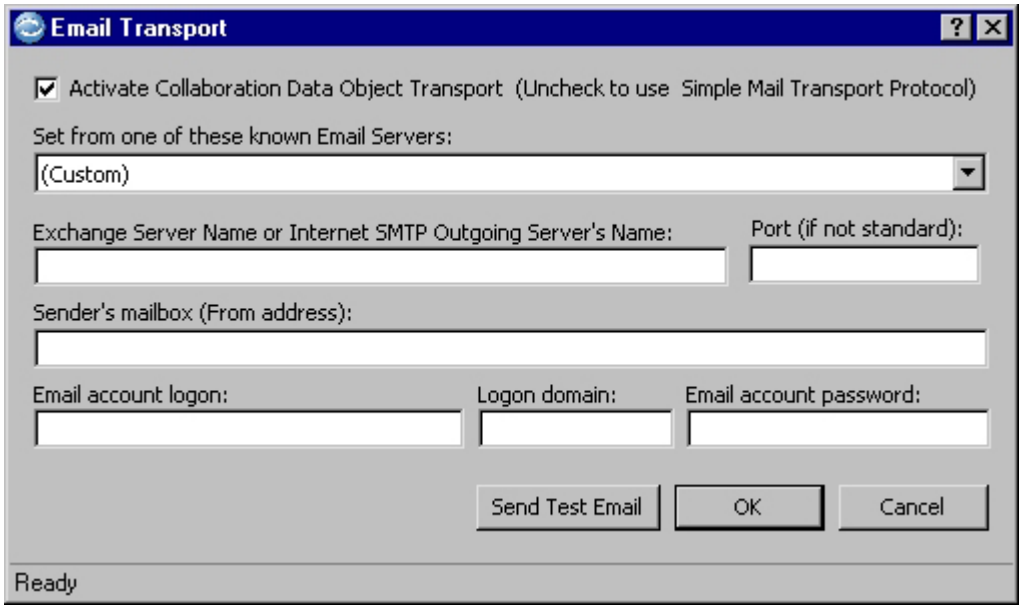
Send Test eMail

Right-click a grid row and select **Send Test eMail** to validate the email address.

Email Transport

Click this button to display the Email Transport dialog so that you may activate and configure the Collaboration Data Object (CDO) transport to send email. If you don't click this button, email is sent using the Simple Mail Protocol Transport (SMTP).

Note: You must use the CDO transport if the email client on the server requires logon credentials to send an unattended message or requires user input when SMTP is used to send a message. Also, select the CDO option if the server uses a Microsoft Outlook client (version 2000 or later) to send messages through a Microsoft Exchange server.



Activate Collaboration Data Object Transport

If you use the SMPT email transport, keep this check box cleared (default) and select **OK**. A popup will ask you if you want to connect without entering a password. If you use the CDO email transport, select this check box to enable the dialog and continue entering information.

Set from one of these known Email Servers

Click to select an account from the list and populate the remainder of the Email Transport dialog with information for the selected account.

Exchange Server Name or Internet SMPT Outgoing Server's Name

Type an exchange server name or internet address.

Port (if not standard)

Type a port name or leave blank (default port).

Sender's mailbox (From address)

Type the sender (From) email address.

Email account logon

Type the email logon name.

Logon domain

Type the domain name.

Email account password

Type the password. A blank password is valid if the account allows it; a prompt will confirm that you want to connect without entering a password.

Send Test Email

Click this button to send a test email to your mailbox.

Note: It is recommended that you send a test email to ensure that the information you entered is sufficient to send an email. If you do not receive the test email, make the necessary corrections to the information you entered.

Appendix A. Enhanced File Names

Optim allows you to automatically generate file names for: Extract Files, Archive Files, Archive Index Files, Compare Files, Control Files.

In addition, for processes invoked using the Command Line Interface, you can generate Output File names via a parameter file.

Using Enhanced File Names

Use macros to dynamically create unique file names for Optim processes. A macro is resolved when a process is run. When specified, a macro remains in the request, so that a new name is generated each time the request is executed.

For example, during an Extract Process, type one or more macros in the Extract File box. The name of a unique Extract File is generated each time the Extract Request is run.

Specify a macro name as follows:

prefix< macro1[,literal1][,macro2][,literal2]...[,macro*N*][,literal*N*] >suffix

where:

prefix A character prefix name (for example, 'Extract').

macro1-n
An Optim-defined macro or a Windows environment variable name (for example, '$USER'). All macros after macro1 are optional.

literal1-n
An optional text string that is inserted in the file name, unchanged (for example, 'workstation').

suffix A character suffix name (for example, '.XF').

Rules and Guidelines

When creating a file name macro, follow these rules and guidelines:

- The total length of the resolved file name cannot exceed 255 characters.
- Any characters (letters, numbers, spaces, or symbols) that are valid for a file name are permitted for the prefix and suffix.
- Macros and literals can be separated by spaces or commas.
- All macros in a file name must be enclosed in a single set of angle brackets (i.e., < $MM $DD $YY >)
- A file name can include an Optim Macro or a Windows environment variable. All Optim macros must begin with a $ symbol (e.g., $USER), and all Windows environment variable macros must be enclosed in % symbols (e.g., %NETUSER%).
- Macro names are not case sensitive.
- The number of macros and literals permitted is limited only by the length of the resulting file name.
- Only the prefix and suffix are validated at the time a process request is edited. When the request is executed, the name is generated and validated. If the resulting file name is invalid, the process will fail.

Optim Macros

Use the following Optim macros to generate unique file names:

Macro	Result
$DAY	The current day of the week in text format (i.e., Sunday through Saturday).
$DD	The day of the month in numeric format (i.e., 01 through 31).
$DE	The day of the week in European numeric format (i.e., 1 through 7, where 1 = Monday).
$DW or $D	The day of the week in numeric format (i.e., 1 through 7, where 1 = Sunday).
$DY or $DDD	The day of the year in numeric format (i.e., 01 through 366).
$HH	The hour in numeric format (i.e., 00 through 23).
$HHMM	The hour and minute in numeric format (i.e., 0000 through 2359).
$MIN	The minute in numeric format (i.e., 00 through 59).
$MM	The month in numeric format (i.e., 01 through 12).
$MON	The month in abbreviated text format (i.e., Jan through Dec).
$SEQ and $SEQn	A numeric Optim Directory Sequence Number (i.e., 0 through 2,147,483,647). The Optim Directory holds a sequence number that is used exclusively by the $SEQ and $SEQn macros. This sequence number is incremented by a single digit each time the macro generates a number. The $SEQn macro generates a number of *n* places, including leading zeros, if needed. For example, if the Optim Directory Sequence Number is 9 when the $SEQ4 macro is used to generate a file name, the macro returns the value 0010: **Macro:** EXTRACT<$MON $YYYY SEQ $SEQ4>.XF **File Name:** C:\ProgramFiles\Softech\RT\DATA\EXTRACTNov2000SEQ0010.XF
$SS	The second in numeric format (i.e., 00 through 59).
$USER	The operating system User ID.
$WE	The week of the year in European numeric format, where Monday is the first day of the week (i.e., 00 through 52).
$WW	The week of the year in numeric format, where Sunday is the first day of the week. (i.e., 00 through 52).
$YY	The current year in two-digit numeric format (e.g., 00 through 99).

Appendix B. Exit Routines for Column Maps

When you create a process request to Convert, Create, Insert, Load, or Restore data, you can specify a Table Map that includes one or more Column Maps to derive the appropriate values for destination columns.

(For details on defining Column Maps, refer to "Create a Column Map" on page 122.)

Optim offers several ways to specify a column value in a Column Map. One way is to specify an exit routine as the source; the exit sets values that could not otherwise be defined for destination columns. Another way is to use exit routines to exclude rows from processing. You can use three types of exit routines:

Standard Exit

The **Standard Exit** routine is called to derive the value for a destination column in a Column Map. This type of exit routine is useful when you want to perform data transformations that are beyond the scope of Column Maps. For example, an exit can change an employee department number for selected rows according to a complex algorithm, or select specific rows to be processed and discard all others.

A standard exit can get a substring segment of a Source LOB column. To insert a new LOB value in a Destination LOB column, an exit can create a file and pass the file name back to the Column Map processor.

Source Format Exit

The **Source Format Exit** is called to format the source column in an Age Function that would otherwise not be supported in a Column Map. This exit routine examines the source date in a character or integer column and converts it into a date format usable as input to the Age Function.

Destination Format Exit

The **Destination Format Exit** is called to format the destination column in an Age Function that would otherwise not be supported in a Column Map. This exit routine converts a date into one of four different destination formats. The data type of the destination column determines the format.

Exit in a Column Map

To use an exit routine in a Column Map, you must specify one of the following in the appropriate source column:

Standard Exit
EXIT *dllname[(parm[[,]parm]...)]*

Source Format Exit
AGE(SRCEXIT=*dllname*)

Destination Format Exit
AGE(DSTEXIT=*dllname*)

The process calls the Column Map exit routine, once for each data row processed and passes a termination call after the last row is processed. Optional parameters specified with a standard exit routine are passed to the exit, and must be string (enclosed in single quotes) or numeric literals (limit 8).

Writing Exit Routines

You can write exit routines in any programming language; however, calls to subroutines must conform to conventions used in the C programming language.

Header Files

To define the parameters, structures, and return codes used in the exit routine, an exit routine must include two C program header files, PSTEXIT.H and, depending on the character format of the metadata (table names, column names, etc.), either PSTCMXIT.H or PSTCMWXT.H:

PSTEXIT.H
Specifies the data types, return codes, and structures for Optim-defined data types.

PSTCMXIT.H
For metadata in single-byte (e.g., ASCII) format.

Provides the prototypes for Column Map call⌂back routines and specifies the *defines* and *structure definitions* for the Column Map exit parameters.

PSTCMWXT.H
For metadata in UTF-16 (WCHAR) format.

Provides the prototypes for Column Map call⌂back routines and specifies the *defines* and *structure definitions* for the Column Map exit parameters.

These header files are located in the same directory with the Optim application files.

Note: If a parameter has a single-byte form and a UTF-16 form, this chapter provides the UTF-16 form in parentheses.

Using DLLs

You must compile and link each exit routine as a separate DLL. Optim loads each DLL dynamically at run time. A DLL can contain only one occurrence of a particular type of Column Map exit. However, you can include one of each type of Column Map exit routine in the same DLL.

You must use the name of the DLL in the Column Map, and you must write the DLL to name and export the actual functions that implement the exit routine:

Standard Exit
PSTColMapExit
(PSTColMapWExit)

Source Format Exit
PSTColMapAgeSrcExit
(PSTColMapAgeSrcWExit)

Destination Format Exit
PSTColMapAgeDstExit
(PSTColMapAgeDstWExit)

Requirements

Each Column Map exit routine must satisfy the following requirements:

- The exit must generate a value appropriate for the destination column and must not change any other storage areas.

- Optim uses the primary key to build internal work areas. If the exit routine assigns a value to a column that is part of the primary key, and that routine is called several times for the same source row, ensure that the exit routine generates the same output value every time.
- An exit may be called frequently; avoid unnecessary overhead. For example, the exit should initialize processing in the first call and save information for subsequent calls in the work area.

Standard Exit Routine

When you write a **Standard Exit**, you specify the exit function, PSTColMapExit (PSTColMapWExit), and the following parameters:

single-byte

```
PSTColMapExit
   (PST_STRUCT_CM_EXIT_PARM * pInputParms,
    PST_STRUCT_CM_EXIT_COL_LIST * pSrcColList,
     PST_STRUCT_CM_EXIT_COL_LIST * pDstColList)
```

UTF-16

```
PSTColMapWExit
   (PST_STRUCT_CM_WEXIT_PARM * pInputParms,
    PST_STRUCT_CM_WEXIT_COL_LIST * pSrcColList,
     PST_STRUCT_CM_WEXIT_COL_LIST * pDstColList)
```

Parameters

When a process calls a standard exit routine, the process passes the following parameters on every call:

pInputParms
Pointer to PST_STRUCT_CM_EXIT_PARM (PST_STRUCT_CM_WEXIT_PARM). This structure contains information about the source and destination tables, the nature of the current call, number of optional parameters specified with the exit, and pointers to the call⌂back functions, work areas, and optional parameters.

The first field in this structure is the **FuncCode** field identified by either PST_CM_FUNC_TRANSFORM (PST_CMW_FUNC_TRANSFORM) or PST_CM_FUNC_TERMINATE (PST_CMW_FUNC_TERMINATE).

The field **NumParms** contains the number of optional parameters specified with the Column Map exit (0 to 8).

The field **pParm** contains an array of pointers to each optional parameter specified with the Column Map exit.

pScrColList
Pointer to PST_STRUCT_CM_EXIT_COL_LIST (PST_STRUCT_CM_WEXIT_COL_LIST). This structure describes the source columns.

pDstColList
Pointer to PST_STRUCT_CM_EXIT_COL_LIST (PST_STRUCT_CM_WEXIT_COL_LIST). This structure describes the destination columns.

Call-Back Functions

Each time an Optim process calls a standard exit routine, the process passes the addresses of the following call-back functions:

pPSTGetColValue( )
Retrieves data for all source columns and most destination columns in the current data row. In general, the exit routine calls this function only once to retrieve the data for a source column. However, the exit routine can call this function several times to retrieve the data for different columns.

pPSTPutColValue( )
Specifies data for the destination column in the current data row. The exit routine determines the value for the destination column and returns the value to Optim. This function must be called unless the row is rejected or the process is aborted.

Processing

Typical processing for the **Standard Exit** routine is summarized in the following steps:

1. On every call from Optim, the exit routine checks for a first time call. On the first call, the exit performs any initialization tasks and normal processing (step 2). On subsequent calls only normal processing is performed (step 2).
2. Optim does not pass column data to the standard exit routine; however, the exit routine can make calls to the **pPSTGetColValue()** call-back function to obtain data for the source columns. These values are needed to determine the value of the destination column.
3. After the exit routine generates the destination value, the exit either calls the **pPSTPutColValue( )** call-back function to store the value in the destination column or passes an appropriate return code instructing the process to skip the data row or abort.
4. After the last data row is processed, the Optim passes a termination call to the exit routine, identified by a value of PST_CM_FUNC_TERMINATE (PST_CMW_FUNC_TERMINATE) in the **FuncCode** field. This call prompts the exit routine to free any dynamically allocated storage. When final tasks are complete, the exit routine passes a return code to Optim.

Return Codes

The following return codes apply to standard exit routines:

```
PST_CM_EXIT_SUCCESS
(PST_CMW_EXIT_SUCCESS)
```

Destination column is assigned a value or is successfully transformed.

```
PST_CM_EXIT_REJECT_ROW
(PST_CMW_EXIT_REJECT_ROW)
```

Destination column cannot be assigned a value or transformed. Discard the row.

```
PST_CM_EXIT_ABORT_PROCESS
(PST_CMW_EXIT_ABORT_PROCESS)
```

Fatal error. Terminate processing. To return an error message, place the message in the work area and set the unused space to blanks or NULL.

Source Format Exit

When you write a **Source Format Exit** routine for Optim, you specify the exit function, PSTColMapAgeSrcExit (PSTColMapAgeSrcWExit), and the following parameters:

single-byte

```
PSTColMapAgeSrcExit
   (PST_STRUCT_CM_AGE_SRCFMT_PARM * pInputParms,
   PST_STRUCT_CM_EXIT_COL_LIST * pSrcColList,
   PST_STRUCT_CM_EXIT_COL_LIST * pDstColList)
```

UTF-16

```
PSTColMapAgeSrcWExit
   (PST_STRUCT_CM_AGE_SRCFMT_WPARM *
pInputParms,
   PST_STRUCT_CM_WEXIT_COL_LIST * pSrcColList,
   PST_STRUCT_CM_WEXIT_COL_LIST * pDstColList)
```

Parameters

When an Optim process calls a source format exit routine, the process passes the following parameters:

pInputParms
Pointer to PST_STRUCT_CM_AGE_SRCFMT_PARM (PST_STRUCT_CM_AGE_SRCFMT_WPARM). This structure contains information about the source and destination tables, the nature of the current call, and pointers to a call-back function and work areas.

The first field in this structure is the **FuncCode** field identified by either PST_CM_SRCFMT_TRANSFORM (PST_CMW_SRCFMT_TRANSFORM) or PST_CM_SRCFMT_TERMINATE (PST_CMW_SRCFMT_TERMINATE).

pSrcColList
Pointer to PST_STRUCT_CM_EXIT_COL_LIST (PST_STRUCT_CM_WEXIT_COL_LIST). This structure describes the source columns.

pDstColList
Pointer to PST_STRUCT_CM_EXIT_COL_LIST (PST_STRUCT_CM_WEXIT_COL_LIST). This structure describes the destination columns.

Call-Back Function

Each time an Optim process calls a source format exit routine, the process passes the address of the following call-back function:

pPSTGetColValue( )
Retrieves data for all source columns and most destination columns in the current data row. In general, the exit routine does not need to call this function because the data for the source column is provided in the first parameter. However, the exit routine can call this function to retrieve the data for different columns.

Processing

Typical processing for the **Source Format Exit** routine is summarized in the following steps:

1. On every call from Optim, the exit routine checks for a first time call. On the first call, the exit performs any initialization tasks and normal processing (step 2). On subsequent calls only normal processing is performed (step 2).
2. The exit receives the value of the source column as specified in the Age Function defined in the Column Map. Optim does not apply the Age Function before calling the exit routine and stores the raw value in one of the **InputValue** union fields as indicated by the **ValueType** field in the header file.
3. If the exit needs to examine other columns to calculate the value for the destination column, the exit must call the **pPSTGetColValue( )** call-back function to obtain the value for those columns.
4. After the destination value is generated, the exit routine must format the value and place it either in the **OutputTimeStamp** field or **OutputSybDateTime** field. Optim validates this value and applies the Age Function. The exit must pass an appropriate return code indicating the field where the data is saved or instructing the process to skip the data row or abort.
5. After the last data row is processed, Optim passes a termination call to the exit routine, identified by a value of PST_CM_SRCFMT_TERMINATE (PST_CMW_SRCFMT_TERMINATE) in the **FuncCode** field. This call prompts the exit routine to free any dynamically allocated storage. When final tasks are complete, the exit routine passes a return code to Optim.

Abort Modes

There are several ways that the **Source Format Exit** routine can abort processing:

- Process rows with skipped dates or invalid dates. If you select either of these options in a process request, and the source and destination columns have the same attributes, the source column is copied unchanged to the destination column. If you do not select either option, the row is rejected.
- Reject the row. Reject the row regardless of the process options for skipped or invalid dates, based on specifications in the exit routine.
- Abort the whole process, based on specifications in the exit routine.

The exit routine passes a return code indicating which date format or which abort mode to use.

During a process, the exit routine may interrogate any columns from the input row and some of the columns from the destination row. However, the exit routine cannot interrogate a destination column that includes an exit routine and is defined in the Column Map after the current destination column. All other destination columns are available.

Return Codes

The following return codes apply to source format exits:

`PST_CM_SRCFMT_USE_TIMESTAMP`
`(PST_CMW_SRCFMT_USE_TIMESTAMP)`

Destination column is assigned a value in the **OutputTimeStamp** field of the first parameter passed to the exit.

`PST_CM_SRCFMT_USE_SYB_DATETIME`
`(PST_CMW_SRCFMT_USE_SYB_DATETIME)`

Destination column is assigned a value in the **OutputSybDateTime** field of the first parameter passed to the exit.

`PST_CM_SRCFMT_SKIP`
`(PST_CMW_SRCFMT_SKIP)`

Aging is not applied. If you do not select the option to **Process rows with skipped dates**, the row is rejected. Otherwise, the data is copied to the source, as long as the source and destination are compatible. If not compatible, the row is rejected.

`PST_CM_SRCFMT_COL_INVALID`
`(PST_CMW_SRCFMT_COL_INVALID)`

Aging is not applied. If you do not select the option to **Process rows with invalid dates**, the row is rejected. Otherwise, the data is copied to the source, as long as the source and destination are compatible. If not compatible, the row is rejected.

`PST_CM_SRCFMT_REJECT_ROW`
`(PST_CMW_SRCFMT_REJECT_ROW)`

Source column cannot be assigned a value. Reject (discard) the row.

`PST_CM_SRCFMT_ABORT_PROCESS`
`(PST_CMW_SRCFMT_ABORT_PROCESS)`

Fatal error. Terminate. To return an error message, place the message in the work area and set the unused space to blanks or NULL.

Destination Format Exit

When you write a **Destination Format Exit** routine for Optim, you specify the exit function, PSTColMapAgeDstExit (PSTColMapAgeDstWExit), and the following parameters:

single-byte

```
PSTColMapAgeDstExit
   (PST_STRUCT_CM_AGE_DSTFMT_PARM * pInputParms,
   PST_STRUCT_CM_EXIT_COL_LIST * pSrcColList,
   PST_STRUCT_CM_EXIT_COL_LIST * pDstColList)
```

UTF-16

```
PSTColMapAgeDstWExit
   (PST_STRUCT_CM_AGE_DSTFMT_WPARM * pInputParms,
   PST_STRUCT_CM_WEXIT_COL_LIST * pSrcColList,
   PST_STRUCT_CM_WEXIT_COL_LIST * pDstColList)
```

Parameters

When an Optim process calls a destination format exit routine, the process passes the following parameters:

pInputParms
Pointer to PST_STRUCT_CM_AGE_DSTFMT_PARM (PST_STRUCT_CM_AGE_DSTFMT_WPARM). This structure contains information about the source and destination tables, the nature of the current call, and pointers to a call-back function and work areas.

The first field in this structure is the **FuncCode** field, identified by:

PST_CM_DSTFMT_TO_CHAR (PST_CMW_DSTFMT_TO_WCHAR),
PST_CM_DSTFMT_TO_INTEGER (PST_CMW_DSTFMT_TO_INTEGER),
PST_CM_DSTFMT_TO_TIMESTAMP (PST_CMW_DSTFMT_TO_TIMESTAMP),
PST_CM_DSTFMT_TO_SYB_DATETIME (PST_CMW_DSTFMT_TO_SYB_DATETIME), or
PST_CM_DSTFMT_TERMINATE (PST_CMW_DSTFMT_TERMINATE).

pSrcColList
Pointer to a PST_STRUCT_CM_EXIT_COL_LIST (PST_STRUCT_CM_WEXIT_COL_LIST). This structure describes the source columns.

pDstColList
Pointer to a PST_STRUCT_CM_EXIT_COL_LIST (PST_STRUCT_CM_WEXIT_COL_LIST). This structure describes the destination columns.

Call-Back Function

Each time an Optim process calls a destination format exit routine, the process passes the address of the following call-back function:

pPSTGetColValue( )
Retrieves data for all source columns and most destination columns in the current data row. In general, the exit routine does not need to call this function because the data for the aged source column is provided in the first parameter. However, the exit routine can call this function to retrieve the data for different columns.

Formats

The input date is in both a PST_C_TIMESTAMP and PST_C_SYB_DATETIME format. The exit is directed to transform that date into one of the following formats, based on the data type of the destination column.

PST_C_CHAR_SZ
CHAR and VARCHAR destination columns.

PST_C_INTEGER_CHAR_SZ
NUMERIC destination columns.

PST_C_TIMESTAMP
DB2 and Oracle DATE/TIME columns.

PST_C_SYB_DATETIME
Sybase ASE DATETIME columns.

Processing

Typical processing for the **Destination Format Exit** routine is summarized in the following steps:

1. On every call from Optim, the exit routine checks for a first time call. On the first call, the exit performs any initialization tasks and normal processing (step 2). On subsequent calls only normal processing is performed (step 2).
2. The exit receives the value of the source column as specified in the Age Function defined in the Column Map. Optim applies the Age Function before calling the Column Map exit and stores the aged value in both the **InputTimeStamp** and **InputSybaDateTime** fields in the header file.
3. If the exit needs to examine other columns to calculate the value for the destination column, the exit must call the **pPSTGetColValue( )** call-back function to obtain the value for those columns.
4. After the destination value is generated, the exit routine must format the value and place it in one of the fields in the **OutputValue** union. The **FuncCode** field indicates the field in the **OutputValue** union where the value must be placed. The exit must return an appropriate code indicating the field where the data is saved or instructing the process to skip the row or abort.
5. After the last data row is processed, Optim passes a termination call to the exit routine, identified by a value of PST_CM_DSTFMT_TERMINATE (PST_CMW_DSTFMT_TERMINATE) in the **FuncCode** field. This call prompts the exit routine to free any dynamically allocated storage. When final tasks are complete, the exit routine passes a return code to Optim.

Abort Modes

There are several ways that the **Destination Format Exit** routine can abort processing:

- Process rows with skipped dates or invalid dates. If you select either of these options in a process request, and the source and destination columns have the same attributes, the source column is copied unchanged to the destination column. If you do not select either option, the row is rejected.
- Reject the row. Reject the row regardless of the process options for skipped or invalid dates, based on specifications in the exit routine.
- Abort the whole process, based on specifications in the exit routine.

The exit routine passes a return code indicating whether the conversion was successful or which abort mode to use.

During a process, the exit routine may interrogate any columns from the input row and some of the columns from the destination row. However, the exit routine cannot interrogate a destination column that includes an exit routine and is defined in the Column Map after the current destination column. All other destination columns are available.

Return Codes

The following return codes apply to destination format exits:

```
PST_CM_DSTFMT_SUCCESS
(PST_CMW_DSTFMT_SUCCESS)
```

Destination column is assigned a value, as specified in the **FuncCode** field of the first parameter passed to the exit.

```
PST_CM_DSTFMT_SKIP
(PST_CMW_DSTFMT_SKIP)
```

If you do not select the option to **Process rows with skipped dates**, the row is rejected. Otherwise, the source data is copied to the target, as long as the source and target are compatible. If not compatible, the row is rejected.

```
PST_CM_DSTFMT_COL_INVALID
(PST_CMW_DSTFMT_COL_INVALID)
```

If you do not select the option to **Process rows with invalid dates**, the row is rejected. Otherwise, the data is copied to the source, as long as the source and target are compatible. If not compatible, the row is rejected.

```
PST_CM_DSTFMT_REJECT_ROW
(PST_CMW_DSTFMT_REJECT_ROW)
```

Destination column cannot be assigned a value. Reject (discard) the row.

```
PST_CM_DSTFMT_ABORT_PROCESS
(PST_CMW_DSTFMT_ABORT_PROCESS)
```

Fatal error. Terminate. To return an error message, place the message in the work area and set the unused space to blanks or NULL.

Sample Exit Routines

Optim application files on the installation CD include a sample set of commented exit routines. These samples show how to use the data areas available for exit routines and provide examples of the type of processing that can be performed using an exit with the sample database.

Refer to the *Installation and Configuration Guide* for a description of each sample.

Appendix C. Row List Files

Point and Shoot allows you to select individual rows (primary key values) from a Start Table to begin Extract Processing. These selected rows are stored in a Point and Shoot File. However, if you need to extract values from data that does not reside in a database table or resides in a database that is not easily accessible, you can create a Row List File manually or by using a utility appropriate for your data source.

If you create a Row List File manually, the file must conform to the file format generated using Point and Shoot. The following guidelines apply:

- Each record in the file cannot exceed 80 characters.
- Data in the file must conform to database syntax and rules for column data types. In addition, the data type and length of the column data must match the attributes of the primary key column(s) in the Start Table.
- The appropriate file extension is **.pns**.

Example 1

For example, assume that you want to extract specific rows from the DETAILS table using a Row List File you create using a method other than Point and Shoot. The primary key for the DETAILS table consists of two columns, ORDER_ID and ITEM_ID. These columns are defined by database:

Database	Column Name	Data Type
DB2/MVS	ORDER_ID	DEC(5,0)
	ITEM_ID	CHAR(5)
Oracle	ORDER_ID	NUMBER(5,0)
	ITEM_ID	CHAR(5)
Sybase ASE	ORDER_ID	DECIMAL(5,0)
	ITEM_ID	CHAR(5)

The following example shows how the list should be structured. (You can create this list using a text editor. If character data includes international characters, the text editor must be UTF-8 compatible, e.g., Windows *Notepad*.) For each row to be extracted, the value in the ORDER_ID column is followed by the value in the ITEM_ID column. Commas separate the values for each row, and a semicolon separates each row.

```
00123, 'CH001'; 00124, 'CH002'; 00125, 'CH003'; 00126, 'CH004';
00133, 'CH001'; 00134, 'CH002'; 00135, 'CH003'; 00146, 'CH004';
00153, 'CH001'; 00154, 'CH002'; 00155, 'CH003'; 00156, 'CH004';
```

Use the following data formats:

- Separate data elements using a comma followed by one or more spaces.
- Separate the primary key values for each row using a semicolon followed by one or more spaces.

Character Data

Character data must be enclosed in single quotes. Embedded quotes must be in the form of two single quotes.

Character data can be wrapped to the next line. The segments of the data must be individually enclosed in quotes without an intervening colon.

The following is an abbreviated example of character data that is wrapped followed by character data that is not wrapped:

'This is an example' 'of wrapped data.'	: No commas
'This is an example', 'of data that does not wrap.', 'Note the use of commas.',	: Commas

Character data stored in fixed length columns is truncated or padded appropriately to fit the column. Character data stored in variable length columns is truncated, as necessary, but is not padded.

Date/Time

All date and time data must be enclosed in single quotes. Any valid database format for these values is acceptable and is handled appropriately.

Numeric Data

Numeric data is not enclosed in quotes. The decimal can be indicated by either a comma or a period and is handled appropriately.

Partial Primary Key

If you want to extract non-unique values or values that do not correspond to a primary key, you can specify an alternate key or a partial primary key in your Row List File. To indicate that the data in this file contains values for some set of the columns, prefix the file with:

COLUMN-LIST

(List the names of the columns for which data is supplied)

END-COLUMN-LIST

Note: The order of column names in the list indicates how the column data is to be processed.

Example 2

Assume you have a set of ITEMS rows that are not in your database. However, you want to extract the DETAILS rows from your database for specific ITEMS. The primary key for the DETAILS rows is comprised of two columns, ORDER_ID and ITEM_ID. However, you prefer to extract rows based on only the ITEM_ID. You can create a Row List File to extract the several DETAILS rows for each specified ITEM_ID value regardless of the ORDER_ID value.

```
COLUMN-LIST
    ITEM_ID
END-COLUMN-LIST
 'CH001'; 'CH002'; 'CH003'; 'CH004';
 'CH005'; 'CH006'; 'CH007'; 'CH008';
 'CH009'; 'CH010';
```

Example 3

Assume that two columns comprise the partial primary key, ORDER_ID and ITEM_ID. Specify the column values in the order in which they are listed for the column list. For each row, in this example, the values for ORDER_ID are followed by the values for ITEM_ID separated by commas. Semicolons

separate the rows. Because commas and semicolons delimit each value, entries can span multiple lines, and multiple entries can be specified on a single line.

```
COLUMN-LIST
   ORDER_ID
   ITEM_ID
END-COLUMN-LIST
00123, 'CH001'; 00124, 'CH002'; 00125, 'CH003'; 00126, 'CH004';
00133, 'CH005'; 00134, 'CH006'; 00135, 'CH007'; 00146, 'CH008';
00153, 'CH009'; 00154, 'CH010';
```

Using the List

Use the following steps to specify your Row List File in an Extract Request:

1. In the **Actions** menu, select **Extract**.
2. In the **Tools** menu, select **Edit Access Definition**.
3. On the **Point and Shoot** tab, select **File** and specify the name of the Row List File.

The file you name is used to extract rows from the Start Table when the Extract is performed.

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing 2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Software Interoperability Coordinator, Department 49XA
3605 Highway 52 N
Rochester, MN 55901
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

All IBM prices shown are IBM's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. _enter the year or years_. All rights reserved.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks

IBM, the IBM logo, DB2, Informix, Optim, OS/390®, Tivoli, z/OS, and ibm.com® are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, and Windows NT are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, or service names may be trademarks or service marks of others.

Glossary

This Glossary defines some of the terms used with Optim.

Access Control Domain
The Access Control Domain (ACD) is a security definition that serves as the foundation for all levels of Optim Security. Each Optim Directory for which Optim Security is initialized contains an ACD named **(Default)** that cannot be deleted. Depending upon the needs of your facility, you may create additional ACDs or use only the **(Default)** ACD. Each ACD includes a list of roles. Each role represents a logical grouping of user and group accounts in your network.

Access Control List
The Access Control List (ACL) is an Optim object that serves as the basis for Object Security. ACL parameters govern the ability of a role to perform actions (such as read, update, or delete) on both the object and the ACL for the object. Each ACD, File Access Definition, and secured Optim object has a unique ACL.

Access Definition
An Access Definition identifies the Start Table, related tables, relationships, and selection criteria that define the data you want to Archive, Extract, Edit, or Compare. You can save and reuse Access Definitions, which are stored in the Optim Directory.

Access Definition Name
An Access Definition name has two parts: *identifier.name*

identifier
(1 to 8 characters)

name (1 to 12 characters)

Age Function
The Age Function is used in a Column Map to age a source column value before inserting that value at a destination. The Age Function is formatted as: AGE(*parameters*)

For details on using the Age Function in a Column Map, refer to "Age Function" on page 162 in Chapter 5, "Column Maps," on page 121.

Archive
Archive enhances database performance by facilitating the removal of infrequently referenced data. This tool allows you to identify and archive sets of relationally intact data before removing selected data from your database. Archived data is indexed and stored. Using **Archive**, you can browse, search, or restore selected subsets of archived data.

Archive Action
Archive Actions are supplemental SQL statements stored in the Table Specifications of an Access Definition that are to be executed at a selected phase of an Archive, Delete, or Restore Process.

For example, you might use this feature to audit deleted rows by executing an appropriate SQL statement before the start of an Archive Delete Process, or after the deletion of each row.

Archive Directory
A database table in the Optim Directory. When an Archive File is created, an entry is created in the Archive Directory that is used to identify and locate the Archive File and any associated indexes.

Archive Directory Maintenance
A utility that creates, updates, or deletes Archive Directory entries, and creates new or modifies existing Archive File indexes. You can also use it to browse an Archive File, or the Access Definition used to create an Archive File.

Archive File
An Archive File contains the selected, relationally intact data described in the Access Definition, and the object definitions needed to recreate the database, if necessary. Archive Files have

a default .af extension and are stored in the Archive Directory specified in Personal Options.

You can browse or selectively restore the data in an Archive File at any time. You can share and use an Archive File as often as needed.

Archive File Security
Archive File Security allows you to control access to data in Archive Files. For example, you might use Archive File Security to prevent any access to data in a specific table or column for most users while granting access to members of selected roles for the same data. Each secured Archive File is associated with a File Access Definition (FAD), which is a security definition that lists tables and columns for which access privileges are defined and, for each listed role, grants or denies privileges to access the archived data.

Archive Index File
An Archive Index File stores index information for a corresponding Archive File. The Archive Index File facilitates the searching, browsing, and restoring of archived data. The default extension for Archive Index Files is .axf.

Archive Process
An Archive Process copies a set of related rows from one or more tables and stores this data in an Archive File. Initiate the Archive Process with an Archive Request that specifies an Access Definition for the data to be extracted and other process parameters, such as delete options.

Archive Request
An Archive Request defines the parameters for archiving and (if desired) deleting data from source tables, and saving that data to an Archive File. An Archive Request references an Access Definition to define the data to archive and the parameters needed to run the Archive Process.

Use the Archive Request Editor to specify an Archive File, an Archive Index File, and a named or local Access Definition. Also, select explicit database objects to extract, override Point and Shoot List parameters, provide values for Variables specified in the Access Definition, and specify Delete options.

You can save and reuse Archive Requests stored in the Optim Directory.

Boolean Constant
A Boolean constant specifies a TRUE or FALSE value. Use a Boolean constant to specify the value of a Boolean column in a Column Map or Relationship.

Browse Utility
The Browse Utility allows you to review the contents of an Archive, Extract, Compare, or Control File. Use the Browse Utility to:

- Verify the contents of a file.
- Identify discarded rows and review diagnostic information in a Control File, when errors occur during a Convert, Delete, Insert, or Load Process.

Calendar
A calendar consists of general options for adjusting and formatting dates and weekend days, dates that have significant meaning for your site (e.g., holidays), and rules to resolve differences when an aged date occurs on a special day.

Calendar Utility
The Calendar Utility allows you to define the calendar year, dates, and business rules to use for date aging and process scheduling. Use date aging when you Convert, Insert, or Load data. Optim includes sample calendars and business rules.

Child Expression
A child expression is a list of columns in the child table that relate to corresponding columns in the parent table. You can accept, modify, or replace these column names.

Column Map
A Column Map defines the specifications for mapping columns of compatible data between source and destination tables. You can map unlike-named columns, modify data, or exclude columns from a process. You can include one or more Column Maps in a Table Map when you create a request to Compare, Convert, Insert, Load, or Restore data.

When you define a Column Map, you can specify the source column value as an explicit column name, NULL, constant (numeric or Boolean), special register, literal (string, hexadecimal, or date/time), function (Substring, Random, Sequential, Age, or Currency), expression (concatenated or numeric), or exit routine.

Column Map Procedure
A Column Map Procedure is a custom program that is referenced by a Column Map and is used for special processing and data manipulation that is beyond the scope of native Column Maps.

Compare
Compare facilitates comparisons of sets of relationally intact data. Using **Compare**, you can compare data resulting from application tests to the original data with all differences highlighted, allowing you to rapidly analyze the effects of your application software or modifications to it.

Compare File
A Compare File contains the results of the Compare Process, including details about changes, matched and unmatched rows, and non-unique values. You can browse the contents of a Compare File. Compare Files have a .cmp extension and are stored in the Data Directory specified in Personal Options.

Compare Process
The Compare Process retrieves the data from Source 1 and Source 2 specified in the Compare Request, compares the rows in each pair of tables, identifies the differences between the two versions of data, and records the results in the Compare File.

Initiate the Compare Process with a Compare Request. The Compare Process uses the Primary Key or a user⌂defined Match Key to identify rows to compare from Source 1 and Source 2. You can use the Browse Utility to review the results in the Compare File.

Compare Request
A Compare Request specifies the data to compare (Source 1 and Source 2) and the parameters for the Compare Process. Use the Compare Request Editor to specify the comparison mode and the data sources:

- In Single Table mode, you can compare one table to another. The source tables can come from an Extract File or the database, and you can specify a Column Map to focus on data in specific columns.
- In Multiple Tables mode, you can compare two sets of tables. The source tables can come from an Extract File, an Access Definition, or include all tables in the database. You can specify a Table Map to focus on specific data.

You can save and reuse Compare Requests stored in the Optim Directory.

Composite Columns
A composite column contains data that corresponds to similar data stored in two or more columns in another table. You can relate data in composite columns using a concatenated expression or the Substring Function.

Concatenated Expression
A concatenated expression combines column values or combines a column value with another value, using a concatenation operator (CONCAT, ||, or +), when you define a Column Map or relationship:

- A **Character Concatenated Expression** consists of one or more character columns, string literals, or substrings of character columns.
- A **Binary Concatenated Expression** consists of one or more binary columns, hexadecimal literals, or substrings of binary columns.

Note: A concatenated expression **cannot** include a zero-length string literal (‘ ‘), a special register, or the Age Function.

Configuration Program
The **Configuration** program allows you to prepare your system and workstations to use Optim. You can start this program immediately following installation, or you can select the Configuration icon on your desk top. For details on using this program, see the *Installation and Configuration Guide* .

Connection String
A connection string permits a workstation to access a particular database. The DBMS uses this connection string to recognize

the database. The database administrator specifies this connection string when configuring Optim.

Control File
A Control File records process specifications and the success or failure of processing. A Control File is generated automatically during Convert, Delete, Insert, or Load Processing, or if you run an Archive Request that deletes rows after they are archived. The Control File provides details on discarded rows and may provide diagnostic information. Specify the name of a Control File when you create a process request. Use the Browse Utility to review the contents of a Control File, which have a .cf extension.

Convert Process
The Convert Process transforms the contents of an Extract File before you perform an Insert or Load. This process is useful to mask sensitive data or to prepare data for download to another platform. Use a Convert Request to initiate the Convert Process and specify the Source File and other process parameters.

Convert Request
A Convert Request identifies the Extract File containing the data to convert and the parameters needed to run the Convert Process. Use the Convert Request Editor to specify a Source File, a destination file to store the converted data, and a Control File to store processing information. You can also specify Table Map options, a Discard Row Limit, and options for date aging, currency conversion, and reporting.

You can save and reuse Convert Requests stored in the Optim Directory.

Create Utility
The Create Utility allows you to copy object definitions extracted from one database and create like-defined objects in the same or a different database.

Use the Create Utility to clone a production database or create a test database that contains tables identical to those in a production database. You can also create objects within the same database, if you define new names for those objects.

Currency Definition
A Currency Definition contains the following:

- General parameters for currency calculations and display options for currency codes.
- Date ranges and conversion rates for currency calculations.
- Lookup tables to correlate database currency codes with standard and user-defined currency codes.

You can specify a Currency Definition when you use the Currency Function in a Column Map or when you create a Convert, Insert, or Load Process request.

Currency Function
The Currency Function is used in Column Maps to convert a source currency to another type of currency, based on a specified conversion rate, before inserting the result at a destination. Format the Currency Function as: CURRENCY(*parameters*). For details on using the Currency Function in a Column Map, see "Currency Function" on page 165.

Currency Utility
The Currency Utility allows you to create and maintain Currency Definitions, which specify the types of currencies, conversion rates, and optional look-up tables for conditional currencies. Optim includes lists of currencies and sample rates.

Data-Driven Relationship
In a data-driven relationship, the parent table is related to one of several child tables, based on data in a particular column. Define a data-driven relationship using string literals, hexadecimal literals, numeric constants, Boolean constants, or NULL.

Date/Time Literal
A date/time literal specifies a value for a date or date/time column in a Column Map. Optim uses the time and short date formats defined for Windows 95/NT, and enclose the date/time literal in single quotes.

Note: To check your workstation format, select **Regional Options** from the Control Panel and review the **Date** and **Time** tabs.

DB Alias
A DB Alias is a set of specifications that allows Optim to identify, locate, and access a particular database. The DB Alias also qualifies the names of objects referenced, defined, or accessed using Optim. The specifications for defining a DB Alias are usually provided by the database administrator.

Default Qualifier
A Default Qualifier is the prefix for unqualified table names: *dbalias.creatorid*.

dbalias
Alias for the database where a table is defined (1 to 12 characters).

creatorid
Identifier assigned to the table (1 to 64 characters). (Use Creator ID for DB2, Schema for Oracle, and Owner ID for Sybase ASE and SQL Server.)

Delete Process
The Delete Process removes sets of related data from a database after an Extract, Archive, or other process, based on the contents of a Source File (either an Extract or Archive File). Use a Delete Request to initiate the Delete Process. The Source File is not changed during the Delete Process; you can use it again to restore the deleted data.

Delete Request
A Delete Request identifies an Extract or Archive File as the Source File containing the data you want to delete, and specifies the parameters for the Delete Process. Use the Delete Request Editor to specify the Source (Extract or Archive) File, a Control File, and the process options, including the commit frequency, discard row limit, and whether to lock tables during the process. You can save and reuse Delete Requests stored in the Optim Directory.

Destination Format Exit
A Destination Format Exit ages dates based on a destination exit routine. This exit routine is called to format the destination column in an Age Function that would otherwise not be supported in a Column Map. This exit routine converts a date into one of four different destination formats, determined by the data type of the destination column.

You specify this exit in a Column Map source column as: AGE(DSTEXIT=*dllname*), where DSTEXIT is the name of the Destination Format Exit.

Discarded Rows
Rows that cannot be processed successfully are marked as discarded and written to a Control File. When the process terminates, you can browse the Control File to view discarded rows and diagnostic information.

You can specify a Discard Row Limit in any request to Archive, Convert, Delete, Insert, or Load data. This limit terminates the process to prevent accumulating a large number of discarded rows.

Edit
Edit is used to browse and edit sets of relationally intact data in database tables. Using **Edit**, you can edit database data, review logical application paths, and browse data to ensure that application test results are as expected. **Edit** supports your rapid development of applications, allows you to analyze the structure of your database, and facilitates your browsing of precisely defined segments of relational data.

Edit Definition
An Edit Definition stores specifications for browsing and editing a set of related data in the Table Editor. An Edit Definition includes:

- Table names, specifications, and selection criteria in a named or local Access Definition.
- Edit preferences, joined tables, and grid options that define the current state of the Table Editor.

An Edit Definition allows you to save the current state of the Table Editor and reopen the Editor using the same specifications. You can save and reuse Edit Definitions stored in the Optim Directory.

Exception Table
An exception table contains copies of rows that violate unique index or primary key rules during a Load Process, and includes a timestamp and a description of

the violation. The DBMS Loader creates an exception table for every source table.

Use the Load Request Editor to select options for creating exception tables. To avoid duplicating or overwriting table names, you can view and specify the names for exception tables in the Load Request.

Exit Routine
An Exit Routine is a set of instructions written outside of Optim. You may specify an exit routine as the source or use it within the Age Function to define a column value. Exit routines are useful when you want to use special processing, manipulate data, or include string literals that exceed the total length of the selected source column. For details on exit routines, see Appendix B, "Exit Routines for Column Maps," on page 473.

Explicit Column Name
An explicit column name can be used to specify a column value in a Column Map. Column names are case-sensitive in Optim.

Note: When you define a relationship, either the parent or the child column must be an explicit column name.

Explicit Relationship
An explicit relationship is a relationship that can be used by a single pair of related tables. The parent and child tables used to define an explicit relationship must have the same Creator ID. Use the Relationship Editor to create an explicit relationship stored in the Optim Directory, or to convert an explicit relationship to a generic relationship.

Export Utility
The Export Utility allows you to copy selected Optim object definitions from a specified Optim Directory to an external file. This utility is useful for moving Optim objects from one Optim Directory to another. After exporting the data, use the Import Utility to copy the exported object definitions into the destination Optim Directory.

Extract File
An Extract File contains a set of related rows extracted from one or more tables, saved in proprietary format. An Extract File can contain data, object definitions, or both. Extract Files have a default extension of .xf and are stored in the Data Directory specified in Personal Options.

You can browse the contents of an Extract File, and can share and use an Extract File as often as needed.

Extract Process
The Extract Process copies a set of related rows from one or more tables and stores this data in an Extract File. The Extract Process always includes the definitions for tables and columns. You can also choose to extract object definitions, including primary keys, relationships, and indexes.

Initiate the Extract Process with an Extract Request that specifies an Access Definition for the data to be extracted and other process parameters.

Extract Request
An Extract Request specifies an Access Definition to define the data to extract and the parameters needed to run the Extract Process. The Extract Request Editor prompts you to:

- Specify the name of an Extract File and a named or local Access Definition.
- Choose to extract data, objects, or both. If applicable, specify object definitions to extract.
- Choose to override the Point and Shoot List or Variables specified in the Access Definition.

You can save and reuse Extract Requests stored in the Optim Directory.

Fast Load
The Fast Load feature significantly reduces processing time when you rerun a Load Request. When a Load Request completes processing, Optim maintains information about the processing parameters and specifications. When you rerun the same Load Request, a confirmation dialog prompts you to confirm whether to use Fast Load Processing.

Fetch Set
A fetch set is a set of rows that Optim reads from the database for each table or view in the Table Editor. You retrieve a new fetch set each time you:

- Join a table in the active Edit Definition.
- Move the Join Arrow in a selected table to retrieve related rows from subordinate tables.
- Set Table Specifications for a selected table.
- Click the Refetch button in a selected table.

You can set a value for Maximum Fetch Rows in Personal Options, up to the site-specific maximum set in Product Options. If you do not set a limit in Personal Options, the limit set in Product Options is used by default. (If a limit is not specified in Product Options, 1000 rows is the default limit.)

File Access Definition
The File Access Definition (FAD) is the basis for Archive File Security. All Archive Files generated by running an Archive Request that references an FAD are secured by the FAD.

Functional Security
As the most general level of Optim Security, Functional Security allows you to control user access to the interface for functions provided by Optim. For example, for a specialized administrator role that is intended to create process requests and objects needed to run these requests, you can grant unlimited access to functions. For members of a role intended only to run the pre-defined process requests, however, you can grant more limited access to functions.

Generic Primary Key
A generic primary key is a primary key that can be used by a set of tables that have the same base name, key columns, and attributes, but different Creator IDs. Generic primary keys are stored in the Optim Directory and are identified by an asterisk (*) as the Creator ID.

Generic Relationship
A generic relationship is a relationship that can be used by related tables that are identical, but have different Creator IDs. Generic relationships are stored in the Optim Directory and are identified by an asterisk (*) as the Creator ID.

Note: You can convert an explicit relationship to a generic relationship; however, you cannot convert a generic relationship to an explicit relationship.

Group Selection
Group selection defines a sample set of rows to extract from a Start Table. You specify a maximum number of rows for a number of unique values based on a selected column in the Start Table. For example, you can extract five rows of customer data from any ten states.

Hexadecimal Literal
A hexadecimal literal specifies a column value in a Column Map or Relationship when the corresponding column contains binary data. Define a hexadecimal literal using the following characters:

X ‘1234567890ABCDEF' or 0X1234567890ABCDEF

For example, to define a Column Map to mask CUSTOMERS data for a test database, where the CUST_ID column has a VARBINARY data type, specify the source column value for the CUST_ID as: X ‘1234CD'.

Import Utility
The Import Utility copies selected Optim object definitions from an external file (created using the Export Utility) to the current Optim Directory. The Import Utility is useful for moving Optim objects from one Optim Directory to another.

Incremental Aging
Incremental Aging is a feature you can use with the Age Function to age dates based on a specified number of years (Y), months (M), weeks (W), days (D), or any combination of these units. For example: [+ or -] *n*Y.

Insert Process
The Insert Process copies data from a Source File into specified destination tables. Use an Insert Request to initiate the Insert Process and specify one of the following methods to perform the Insert Process:

- Insert adds only new rows to a destination table.
- Update replaces existing rows in the destination table.

- Update/Insert updates existing rows and inserts new rows.
- Mixed allows you to specify Insert, Update, or Update/Insert Processing for any table individually.

Insert Request

An Insert Request specifies a Source File containing the data you want to insert or update, and the parameters needed to run the process. The Insert Request Editor prompts you to:

- Specify a Source File and a Control File.
- Specify Table Map and Delete Options.
- Select insert, update, update/insert, or mixed Process Options.
- Choose options for reporting and date aging, as needed.

You can save and reuse Insert Requests stored in the Optim Directory.

Join When viewing a table using the Table Editor, the Point and Shoot Editor, or the Browse Utility, you can use the Join command to display the related data from other tables.

Large Object (LOB)

The term Large Object (LOB) refers to two data types, BLOB (Binary Large Object) and CLOB (Character Large Object). A LOB typically contains a large amount of data (for example, objects such as audio files, photos, and video files).

You can view LOBs through the Table Editor, the Point and Shoot Editor, or the Browse Utility. LOBs are displayed in Native or Non-Native Mode:

- In Native LOB Mode, a set of three icons displays in a LOB column; you can select the corresponding icon to view LOB data in an associated application, in character mode, or in hexadecimal mode.
- In Non-Native LOB Mode, LOB data displays as normal table data (either VARCHAR or VARBIN).

Load Process

The Load Process transforms the contents of a Source File into the load utility format for a supported database. Initiate the Load Process with a Load Request that specifies the Source File containing the data to load and other process parameters.

Load Request

A Load Request specifies a Source File (either an Extract or Archive File) containing the data you want to load and the parameters needed to run the process for the DBMS Loader you are using. The Load Request Editor prompts you to specify:

- A Source File, Control File, and Table Map options.
- DBMS Loader options.
- Options for reporting and date aging, as needed.

You can save and reuse Load Requests stored in the Optim Directory.

Match Key

A Match Key is used in a Compare Process to match rows from Source 1 and Source 2 specified in a Compare Request. Typically, the Compare Process uses the Primary Key from one of the source tables to process the comparison. However, if neither source table has a Primary Key, you can specify one or more other columns to define a Match Key.

Note: Match Keys are only used in the active Compare Request and are not saved to the Optim Directory.

Move **Move** facilitates the extraction and migration of sets of relationally intact data. Using **Move**, you can create test databases that are referentially intact subsets of a production database or extract sets of related data and transform it as you migrate the data to the test database.

Multi-byte

A multi-byte character has a code point that may require more than a single 8-bit byte to be expressed. Multi-byte character sets may be of either fixed width, in which all code points require the same number of bytes, or variable width, in which different code points may require a different number of bytes. JA16SJIS is a variable width character set, as are UTF8 and AL32UTF8.

Naming Conventions
Optim uses the following naming conventions.

Optim Objects:
identifier.name

Table Names and Primary Keys:
dbalias.creatorid.tablename

Relationships:
dbalias.creatorid.tablename.constraint

NULL NULL indicates a column value in a Column Map or Relationship is unknown. The destination column must be null eligible or defined as NULL.

Numeric Constant
A numeric constant specifies a number for a column value in a Column Map or Relationship. You can use a numeric constant as the source for numeric destination columns.

DB2 INTEGER, SMALLINT, DECIMAL, FLOAT, DOUBLE

Oracle NUMBER, FLOAT

Sybase ASE
TINYINT, INT, SMALLINT, DECIMAL, FLOAT, REAL, MONEY, SMALL MONEY

SQL Server
TINYINT, INT, SMALLINT, DECIMAL, FLOAT, REAL, MONEY, SMALL MONEY

Informix
DECIMAL, INT, SMALLINT, FLOAT, SMALLFLOAT, NUMERIC, REAL, SERIAL, MONEY, DOUBLE PRECISION

Note: The numeric constant must represent a value that fits into the destination column, based on data type, precision, and scale.

Numeric Expression
A numeric expression specifies a value in the source column in a Column Map. The data types for the corresponding source and destination columns must be compatible. A numeric expression consists of:

(Operand 1) Operator (Operand 2)

where *operand* must be a numeric column or a numeric constant, and the *operator* specifies whether to add (+), subtract (-), divide (/), or multiply (*).

Object Definition
An object definition specifies the set of parameters a database requires to create a table, primary key, relationship, index, view, or alias.

Object Security
Object Security allows you to control access to specific objects in the Optim Directory, using an Access Control List (ACL). Any Optim object can be secured by associating it with an ACL. An ACL lists roles and grants or denies privileges for each role to read, update, delete, or execute (where appropriate) the object and the ACL. For example, you might use Object Security to allow members of a role to read and execute, but not edit, a specific Archive Request.

Optim Optim enhances database performance by automating the process of archiving, editing, migrating, and comparing relationally intact data from multiple database tables. Optim includes the following components: **Archive** allows you to identify and archive sets of relationally intact data before removing selected data from your database. **Move** facilitates the extraction and migration of sets of relationally intact data. **Edit** is used to browse and edit sets of relationally intact data in multiple tables. **Compare** facilitates comparisons of sets of relationally intact data.

Optim Directory
The Optim Directory contains a set of tables that store all the product-specific object definitions you create using Optim. Typically, a site uses one shared Optim Directory, regardless of the number of database instances to be accessed or the number of workstations using the product. However, you may create more than one Optim Directory, as needed for your site. Use the Optim Directory Editor to view or modify general, server, and connection settings for an Optim Directory.

Optim Object
An Optim Object is a user-defined object

that is unique to Optim and is stored in the Optim Directory. Optim Objects include Access Definitions, Table Maps, Column Maps, primary keys, relationships, Column Map Procedures, Storage Profiles, various process requests, security definitions, and Archive File registrations.

Oracle Sequence Function
The Oracle Sequence Function is used in a Column Map (for Insert or Load Processing) to assign a value to a destination column based on the Oracle Sequence. This function is formatted as: *schema.seqname*.NEXTVAL(INCL_UPD)

schema
Identifier for the name of the Oracle Sequence.

seqname
Name of the Oracle Sequence that assigns sequential values.

NEXTVAL
Inserts the next Oracle Sequence value into the destination column.

INCL_UPD
Updates a sequence value assigned to a column when rows are updated during an Insert.

Parent Expression
A parent expression is a list of columns in the parent table that relate to corresponding columns in the child table. If the parent table has a primary key, the column names are inserted automatically. You can accept, modify, or replace the column names.

Passwords and User IDs
Passwords and User IDs allow you to connect to a particular database. You can set passwords and User IDs on the **Logon** tab in Personal Options. The maximum length for passwords and User IDs varies depending on the DBMS.

Permission
A permission indicates whether a user is allowed or denied access to Functional and Object Association privileges, Optim objects, and Archive File data.

Functional privilege permissions can be set for each role in the (Default) Access Control Domain (ACD). Object Association privilege permissions can be set for roles in any ACD.

Each secured object, including ACDs and File Access Definitions (FADs), has an Access Control List (ACL) that allows or denies Read, Update, Delete, and Execute permissions for the object to roles defined in the associated ACD. The ACL also allows or denies Read, Update, and Delete permissions for the ACL itself.

Access permissions for Archive File data are defined in a FAD, which is associated with each secured Archive File. The FAD lists tables and columns for which access permissions are defined. For each listed role, the FAD grants or denies permissions to access the archived data.

Personal Options
Personal Options set preferences for using Optim from your workstation. You can set preferences for confirmation prompts, display features, error messages, schedule monitoring, DBMS Loader directories, defaults for using the Create Utility, logon passwords, Server defaults, display options for Edit and Browse, Archive directories, removable media options, defaults for using Actions, and default options for automatic email notification.

To return to the settings in place upon opening the Personal Options dialog, click **Defaults**.

Note: Many site-specific Product Options may override your Personal Options.

Point and Shoot
Point and Shoot allows you to select specific rows from a Start Table to be used in a process request. Point and Shoot is available when you create an Access Definition, or when you use an Extract, Archive, or Restore Request.

When you use Point and Shoot to select rows from the Start Table, the primary keys for these rows are stored in a local Point and Shoot List or an external Point and Shoot File. The process request uses the primary keys to identify the rows to process first.

Point and Shoot List
A Point and Shoot List is a list of primary key values for the rows you want to

include in an Archive, Extract, or Restore Process request. You can create two types of row lists using Point and Shoot:

- A Point and Shoot List saved locally, stored with and available only for the associated Access Definition or process request.
- A Point and Shoot List saved as an external Point and Shoot File, which can be referenced by other Access Definitions and process requests.

Primary Key

A Primary Key defines a column or set of columns that uniquely identify each row in a table. Although you can view primary keys defined to the database, you can edit only those primary keys defined to the Optim Directory.

Privilege

A privilege defines a type of user access to the Optim functions and objects. Optim provides Functional and Object Association privileges, for which permissions may be granted or denied.

Functional privileges control user access to functions and dialogs in Optim. For example, they allow users to create, run, and access requests for actions, definitions, security definitions, and utility definitions. They also allow users to execute a utility from the command line, perform tasks within the Configuration program, edit Product Options, use Security dialogs to secure Optim objects, functions, and Archive Files, export/import secured Archive Files, modify a File Access Definition, and run Security Reports.

Object Association privileges control a user's ability to secure types of objects, using an Access Control List (ACL). Objects include action requests, definition requests, and utility definitions. Any Optim object can be secured by associating it with an ACL. A member of a role that is granted the Object Association privilege for an object type (e.g., Access Definitions) can define an ACL for any object of that type that the member creates. Object Association privileges allow a role to use the roles defined in an Access Control Domain (ACD) as the basis for an ACL that protects objects of the indicated type. An ACL lists roles in the associated Access Control Domain (ACD) and grants or denies permissions for each role to read, update, delete, or execute (where appropriate) the object and its ACL.

Privileges are organized into "classes". For example, the Invoke Action Editors Functional privilege class includes the privileges that make the Action Editors (that is, Archive, Compare, Convert, Delete, Extract, Insert, Load, Report, Restore, Table Editor) available to users. The Associate Action Editors Object Association privilege class includes privileges that allow users to secure requests created by the Action Editors.

Product Options

Product Options are a set of preferences that apply to all workstations using Optim. Specify processing limits, site defaults, database options, the current Product Configuration File, the password to access Product Options, Edit options, Server configuration, Archive options, DBMS Loader parameters, and report formatting defaults.

Many site-specific Product Options override Personal Options. To return to the settings in place upon opening the Product Options dialog, click **Defaults**.

Note: Before you can view or modify Product Options, you must provide a password.

Propagate Function

The Propagate Function is used in a Column Map (for Insert, Load, or Convert Processing) to assign a value to a destination column, and then propagate that value to all related tables. Format this function as: PROP(*value*,*column-name*)

value Value for the destination column.

column-name
Name of the source column containing the value for the destination column that is to be propagated to all related tables.

Random Function

The Random Function specifies a column value in a Column Map by returning a random number within the range

specified by the low and high values. This function is formatted as: RAND(*low,high*)

low Lowest possible random value in the range.

high Highest possible random value in the range.

Relationship
A relationship defines columns in a parent table that relate to corresponding columns in a child table. The data types of the corresponding columns must be compatible. Relationships can be defined to the database or the Optim Directory; however, you can create and edit only those relationships defined to the Optim Directory.

Relationships defined to the Optim Directory do not have to conform to database conventions. You can use these flexible "extended Optim relationships" to relate parent and child columns using column names, NULL, literals, constants, concatenated expressions, and the Substring Function.

Relationship List
The list of relationships (found on the **Relationships** tab of the Access Definition Editor) defined for the tables in an Access Definition. You can specify the traversal path for extracting data from two or more pairs of related tables. Typically, Optim traverses all relationships from parent to child; however, you can specify other traversal options in an Access Definition.

Report Process
A Report Process summarizes the contents of an Archive or Compare File according to your specifications, and stores this data in a Report File. Use the Report Request Editor to select the Archive or Compare File and other specifications.

Report Request
A Report Request contains the specifications needed to collect data from an Archive or Compare File, and send the output to a file, a printer, or both.

The Report Request Editor prompts you to specify the Source File and the tables to be included in the Report Process, layout and formatting options, and whether to send the output to a file, printer, or both.

Reset Object Cache
The Reset Object Cache command is available from the **Utilities** menu to reset current database objects or configuration parameters. (Optim uses a caching scheme to improve performance.)

Restart/Retry
The Restart/Retry Utility allows you to:

- Restart Processing from the last commit point when a Delete, Insert, or Insert/Update Process ends abnormally because of an internal error, resource limitations, or user request.
- Retry the Process for discarded rows when a Delete, Insert, or Insert/Update Process ends normally, but a number of rows could not be processed (due to conflicting data types or referential integrity rules). The process writes these discarded rows to a Control File, which you can browse. Once you resolve any problems, you can retry the process.

Restore Process
A Restore Process selects data from one or more Archive Files and restores the data to the original or a different database. Initiate the Restore Process with a Restore Request that specifies the Archive Files and defines the Insert or Load Request used to restore the archived data.

Restore Request
A Restore Request defines the parameters for restoring data from one or more Archive Files to the original or a different database. Use the Restore Request Editor to specify the Archive File(s) to be restored, global or local selection criteria, an Insert or Load Request to restore the archived data to the database, and a Request Selection Mode to match the Insert or Load Request with the appropriate Archive File.

You can save and reuse Restore Requests stored in the Optim Directory.

Save Saves the current version under the same name, replaces the original version, and displays the current version.

Save As
Saves the current version under a different name, preserves the original version, and displays the newly named

version. Save As is useful for modeling a new item based on an existing item.

Save Copy As
Saves a copy of the current version under a different name, preserves the original version, and displays the current version. Save Copy As is useful if you want to save a copy of an item at a given stage in development and continue editing the current version.

Schedule Utility
The Schedule Utility allows you to create and maintain scheduled processing jobs for one or more process requests to Archive, Compare, Convert, Delete, Extract, Insert, Load, or Restore data. Scheduling details include the start date/time, latest start time, cycle options, and the definition of the process requests. The Schedule Utility interfaces with the Scheduling Monitor to track the progress of scheduled jobs.

Scheduling Monitor
The Scheduling Monitor is an Optim application that tracks each scheduling job. This monitor allows you to view and edit the number of scheduled jobs, monitor active jobs, and review the results of completed jobs.

Segment Size
When an Archive or Extract File is larger than the storage capacity for a target media type, the file must be segmented to span more than one volume. You can specify default segment size values for fixed and removable media storage in Personal Options.

Semantic Aging
Semantic Aging is a feature of Optim that allows you to define a set of rules for date aging. For example, you can define a set of rules to adjust aged dates to result in valid business days.

Sequential Function
The Sequential Function specifies a column value in a Column Map by returning a number that is incremented sequentially by the step value based on the start value. This function is formatted as: SEQ(*start*,*step*)

start Starting value for the sequence.

step Value to increment the sequence.

Set as Default
The Set as Default command allows you to profile the entries you specify on a process request or definition editor as the default entries. If you select **Set as Default** from the **File** menu, the next time you open that editor, your default entries display automatically.

Source File
A Source File is an Extract File or an Archive File used as the source of data for a process. You must specify a Source File when you create a Convert, Delete, Insert, or Load Process request, or when you use the Create Utility to create database objects.

Source Format Exit
A Source Format Exit ages dates based on a source exit routine. This exit routine is called to format the source column in an Age Function that would otherwise not be supported in a Column Map. This exit routine examines the source date in a character or integer column and converts it into a date format usable as input to the Age Function.

Specify the Source Format Exit in a Column Map source column as AGE(SRCEXIT=*dllname*), where SRCEXIT is the name of the Source Format Exit.

Special Registers
A special register stores system information that can be referenced in an SQL statement. You can use a special register to specify a column value in a Column Map. You can use the following special registers:

- CURDATE( )
- CURRENT DATE
- CURRENT_DATE
- CURRENT TIME
- CURRENT_TIME
- CURRENT TIMESTAMP
- CURRENT_TIMESTAMP
- CURRENT SQLID
- CURRENT_SQLID
- CURTIME ( )
- GETDATE ( )
- GETTIME ( )
- NOW ( )

- SYSTDATE ( )
- USER
- WORKSTATION_ID

SQL WHERE Clause
An SQL WHERE Clause specifies criteria for extracting or archiving data. You can use an SQL WHERE Clause when you define Table Specifications as part of your selection criteria in an Access Definition. The statement must conform to SQL syntax.

Standard Exit
A Standard Exit is an exit routine that is called to derive the value for a destination column in a Column Map. Specify the Standard Exit in a Column Map source column as: EXIT *dllname*.

Start Table
The Start Table is the first table to use in processing an Archive or Extract. You can specify any table in the Access Definition as the Start Table, except a reference table. If you do not explicitly specify a Start Table, the first table in the table list is the Start Table.

Storage Profile
A Storage Profile allows you to define storage parameters for creating an Archive File on fixed or secondary media. You can use a Storage Profile to automatically create a duplicate Archive File, or to copy an Archive File to a supported backup device. You can also use a Storage Profile to override default values defined in Personal Options for segment size and the default value defined in Product Options for a minimum retention period (for supported devices). Use a Storage Profile by specifying the name of the Storage Profile in an Archive Request.

Storage Profile Utility
The Storage Profile Utility allows you to create and maintain Storage Profiles. Access the Storage Profile Utility from the **Utilities** menu on the main window. To edit a Storage Profile referenced in an Archive Request, select **Edit Storage Profile** from the **Tools** menu on the Archive Request Editor.

String Literal
A String Literal specifies a column value in a Column Map or Relationship when the corresponding column contains character data. To define a string literal, you can use any characters, enclosed in single quotes. For example: '90210' or 'CA'.

Subset Extract File
The Subset Extract File is generated during a Restore Process when you define selection criteria to restore a subset of an Archive File. That subset is automatically extracted from the Archive File and stored in a Subset Extract File, which is then restored to the selected database. You can specify whether to save or delete the Subset Extract File after the data is restored.

Substring Function
The Substring Function specifies a column value in a Column Map or Relationship by returning a substring of the named column. This function is formatted as: SUBSTR(*column-name,start,length*)

column-name
Name of character or binary column.

start Position of the first character in the string.

length Number of characters to use.

TableList
List of table names in an Access Definition to specify the data to extract or archive. To create the table list, specify a Start Table and add other tables.

Table Map
A Table Map defines specifications for correlating source and destination tables of compatible data. You can map tables that have different names, modify table names, exclude tables from a process, or include Column Maps for greater control over the data.

You can use Table Maps in a Convert, Insert, or Load request, or with the Create Utility. You can save and reuse Table Maps stored in the Optim Directory.

Target Aging
Target Aging is a feature that allows you to perform date aging based on a specific date. For example, you can specify the base date of the production run and

specify a target date for a particular test. Optim applies business rules and automatically ages dates to maintain the relationship between the base and target.

Traversal Cycle
A traversal cycle occurs when an Extract or Archive Process revisits tables specified in an Access Definition. Traversal cycles depend on the specifications for processing each relationship.

Traversal Path
A traversal path directs an Extract or Archive Process through related tables specified in an Access Definition. The traversal path is defined by relationships selected between tables, as well as the responses to Options 1 and 2 on the **Relationships** tab of the Access Definition Editor.

The typical traversal path begins at the Start Table and proceeds through the data model traversing all relationships from parent to child.

Unicode
The universal character encoding, maintained by the Unicode Consortium. This encoding standard provides the basis for processing, storage and interchange of text data in any language in all modern software and information technology protocols.

The Unicode Standard provides a uniform architecture and encoding for all languages of the world, with over 95,000 characters currently encoded. Unicode is a fundamental component for providing seamless data interchange around the world, and has been adopted by such industry leaders as Apple, HP, IBM, JustSystem, Microsoft, Oracle, SAP, Sun, Sybase ASE, Unisys and many others.

For additional information on Unicode or the Unicode Consortium, please visit http://www.unicode.org.

User ID
A User ID specifies the authorization control that permits individuals to use specific Optim options and processes. The database administrator usually assigns User IDs when setting authorizations and privileges required for accessing a particular database.

Variables
Variables are user-defined default values specified in an Access Definition. You can use these substitution variables to specify column selection criteria or to create an SQL WHERE Clause.

Index

A

Access Control Domain
  definition 491
Access Control List
  definition 491
Access Definition Editor
  criteria indicators 51
  default qualifier 50
  extract parameters 53
  Group Selection 74
  Indented Table Display 104
  menu commands 49
  Point and Shoot options 73
  reference tables 52
  Relationship Index Analysis 106
  Select Tables dialog 55
  show steps 108
  specify relationships 57
  Start Table 50
  substitution variables 71
  table specifications 52, 76
  table/view list 51
  Tables tab 50
  uncommitted read 53
Access Definitions 45, 119
  components of 45
  create 46
  edit 48
  effects of database changes 118
  export/import file format 341, 349
  select to edit 47
Action Request Editor, email
  notification 32
Actions Menu
  browse files 300
  schedule process request 283
  selecting commands 7
Actions Options 461, 463
  Column Map Procedure 461
  format numeric values 463
  maximum database connections 466
  monitor update frequency 462
  report retention 463
  show aging tabs 462
  show currency tab 462
Add Database Indexes 100
Add Relationship Indexes 100
Advanced Options, Access Definition 49
  Key Lookup Limit 60
  Table Access 60
Age Function 163, 165
  description of 163
  destination format exit 473
  examples 165
  incremental aging 163
  restart/retry a process 257
  semantic aging 163
  source format exit 473
  using calendars 259
Allocation Percent
  DB Alias indexes 199
Allocation Percent *(continued)*
  index defaults 444
  table defaults 443
Archive Actions 85, 99
  Action phase 85
  Built-ins 88
  process status 99
  row status 99
Archive File Collections
  export/import file format 349, 350
Archive Files
  browsing 299
Archive Index 100
Archive Options 458, 459
  Archive browse index directory 458
  Archive directory 458
  Archive index directory 458
  directory for Subset Extract Files 459
  file backup device 459
  trim search list 459
  use index to search 459
Archive Requests
  export/import file format 369, 374
Auditing
  enable/disable for tables 455
  Personal Options tables list 454, 455
  PSTAUDIT 455
Auto Switch
  stacked tables 317

B

Backup devices 3
Browse file 299, 318
  join button 305, 315
  join tables 314
  stacked tables 317
  unjoin button 305
  unjoin tables 317
Browse Options 456, 457
  auto switch 457
  display changed data 457
  display column attributes 456
  display row count 457
  maximum exclude rows 457
  side label display 456
Browse Utility
  Optim Files 299
Browse Window
  columnar display 307
  filtering on/off 113
  join button 113
  refetch button 113
  side label display 308
  toolbar buttons 304
  unjoin button 113
Buffer Pool
  Create Utility indexes 421
  DB Alias indexes 199
  index defaults 416, 444

C

Calendar
  export/import file format 350, 353
Calendar Editor
  calendar dates 263
  calendar months 262
  calendar rules 267
Calendar Utility 259, 273
  create a calendar 260
  description of 3, 259
  edit a calendar 261
  sample business days 269
  sample dates 268
  sample rules 270, 273
  select a calendar to edit 260
Cascading Delete/Update
  Confirmation dialog 35
Code Page
  selecting 40, 193
Column Map Editor
  edit source columns 132
  menu commands 129
  select source columns 131
  show column status 130
  specify column values 130
Column Map Procedure Editor 176, 186
Column Map Procedures 175, 187
  create new 176, 177
  delete 187
  export 186
  export/import file format 354, 355
  import 186
  sample 177, 178
  save 187
  select to edit 177
Column Maps 121, 173
  Age Function 163, 165
  Compatibility Rules 169
  create 123
  Currency Function 165, 168
  data compatibility rules 168
  Data Privacy Transformation Library
    Functions 151
  description of 121
  edit 127
  export/import file format 353, 354
  Hash Lookup Function 143
  Identity Function 136
  in Table Maps 247, 248
  Lookup Function 141
  numeric expressions 139
  Oracle Sequence Function 137
  Propagate Function 138
  Random Function 135
  Random Lookup Function 147
  select to edit 125
  Sequential Function 136
  Serial Function 136
  Source Column dialog 134
  source/destination columns 124
  Substring Function 135

Column Maps *(continued)*
syntax guidelines 133
using exit routines 175, 473
Columnar Display
description of 307
switch to side label 307
Command Line Interface 325, 328, 337, 339
import 325, 337
syntax guidelines 325, 337
Commit Frequency
restart/retry a process 257
Compare Files
browsing 299
Compare Requests
export/import file format 374, 375
Comparison Compatibility Rules 169
Compatibility Rules
in Column Maps 168, 169
Confirm Options 431
before deleting 431
before losing DDLs 431
before overwriting files 431
Control Files
browsing 299
example 318
Convert Requests
export/import file format 376, 380
Create tab, Personal Options 440, 446
compile error drop 441
DB Alias 441
DB2 OS/390 current rules 442
Limits 442
Object Name Highlighting 442
replace UDTs 442
set alias defaults 445
set index defaults 443
set synonym defaults 446
set table defaults 442
set trigger defaults 447
Create Utility 411, 426
browse SQL 423, 425
create a test database 411
create objects 417
create/drop database keys 206
default tables/indexes 418
drop objects 418
object status 414
object types 415
resolve conflicts 414
respecify options 414
review SQL 423
select objects 414
show all objects 416
switch DBMS and Optim keys 417
Table Map options 413
Create Utility indexes 421
Cross-Hatching 22
Currency Editor 276, 281
conversion rates 278
create a currency table 276
currency types 281
date ranges for rates 278
decimal places for conversion 277
show currency codes 277
Currency Function 165, 168
description of 165
Currency Function *(continued)*
direct conversion 165
examples 167
triangulation 166
valid parameters 166
Currency Tables
export/import file format 355, 356
Currency Utility
currency tables 275
description of 275
sample currency tables 282

D

Data Privacy Transformation Library Functions 151
CCN function 154
EML function 157
SSN function 151
Data Types
in Column Maps 168
in relationships 226
DB Alias 189, 203
default for Import 332
description of 189
export/import file format 356, 359
select to browse 189
storing Optim Directory 194
DB Alias Editor
browse DB Alias details 190
set alias defaults 201
set index defaults 198
set synonym defaults 200
set table defaults 197
set trigger defaults 202
view DBMS details 192
view server details 196
view/change connection string 195
DB2
MVS Buffer Pool 199, 416, 421, 444
specify default database 197
DBMS
network server name 196
package/procedure qualifier 196
server signature 197
type and version 192
Decimal Separator
for numeric strings 194
Default Database
DB Alias tables 197
Default Directories
Archive 458, 459
Archive Browse Index Directory 459
Archive Index Directory 458
Data Directory 430
for DBMS loaders 438
Product Configuration File 430
scheduling 437
Temporary Work 430
Trace Files 430
Default Identifier
aliases 201
indexes 198
synonyms 200
triggers 202
Default Tablespace
Create Utility indexes 421
Default Tablespace *(continued)*
Create Utility tables 420
DB Alias indexes 199
DB Alias tables 197
Definitions Menu
selecting commands 8
Delete Requests
export/import file format 380
Delimiters
string delimiter for DB Alias 194
Dialogs and Editors
adjust a grid manually 19
grid column shortcuts 23
grid heading shortcuts 21
parts of a dialog 15
using Find/Replace 24
using grids 18
using key combinations 20
using the Open dialog 25, 26
Discard Row Limit
restart/retry a process 257
Display
column attributes 111, 304
Display tab, Personal Options 432, 434
column delimiters 432
Large Objects 433
main window 433
maximum fetch rows 432
maximum File menu entries 433
maximum history entries 433
menu behavior 433
null value indicator 432
system messages 434
tooltips and toolbars 433

E

Edit Definitions
export/import file format 359
Edit Menu
selecting commands 17
Edit tab, Personal Options 452, 454
audit tables 454
auditing active 453
auto switch 452
default mode 454
display column attributes 452
display deleted rows 452
display row count 454
null as default 453
prompt for variables 453
retain selection criteria 453
side label display 453
single view 453
undo levels 454
user supplies defaults 453
warn on cascade 453
Edit Window
columnar display 307
Email Notification 32, 468
Enhanced file names, using macros 471
Error Messages
Export processing 323
Import processing 336
Errors tab, Personal Options 435, 436
display lines 436
error messages font 436

Errors tab, Personal Options *(continued)*
hide empty message bar 436
informational messages font 435
warning messages font 436
Exit Routines
destination format exit
abort modes 480
call-back function 479
formats 479
function 479
input to Age Function 473
parameters 479
processing 480
return codes 480
guidelines for writing 474
in Column Maps 473
requirements 474
sample header files 474
source format exit
abort modes 477
call-back function 477
function 476
input to Age Function 473
parameters 477
processing 477
return codes 478
standard exit
call-back functions 475
parameters 475
processing 476
return codes 476
types of 473
using DLLs 474
Export Utility 320, 325
append definitions 321
building the output file 321
exporting objects 320
menu commands 323
MVS definitions 339
output file 321
processing errors 323
show/print process log 325
subordinate definitions 322
Export/Import file formats 339, 410
Extract Files
browsing 299
Extract Requests
export/import file format 381, 385
use a Row List File 485

F

Fetch Set
refetch rows 113
File Menu
selecting commands 6
Find/Replace
column criteria 25
search criteria 24
using to search/replace 23
Functions and statements
listed by type 180
quick reference 180

G

General tab, Personal Options 429, 430
Caps mode 430
data directory 430
days to keep trace files 430
Local Optim Server (ODBC) 430
Product Configuration File 430
SQL LIKE character 429
temporary work directory 430
unchained mode 467
warn on cascade 467
z/OS code page 430
Global functions 183
Grid Patterns 22
Group Selection
indicator 51
set of unique rows 74

H

Hash Lookup Function 143
Help Menu
selecting commands 12

I

Identifer Case
for database objects 192
Identity Function 136
Import Utility 328, 337
Access Definition names 333
continue if errors 331
default DB Alias 332
description of 319
error messages 336
generate constraint names 332
importing objects 328
menu commands 329
monitor progress 331
MVS and OS2 definitions 339
Optim objects 335
overwrite existing definitions 331
select objects to import 330
show/print process log 337
Increase Fetch Limit 112
Indented Table Display
default qualifier 104
show relationships 104
Index Analysis 106
Index Buffer Pools 199, 416, 421, 444
Index Defaults 416, 444
Insert Requests
export/import file format 385
Invert Row Selection 112

J

Job Details
adjust dates to calendar 286
edit job start times 284
edit job steps 287
edit repeat cycles 285
stop job on error 288, 289
Join
related tables 113, 305
Join *(continued)*
specify a relationship 114, 316
stacked tables 114, 317
unjoin tables 113, 305
Join Button
in browse window 315

K

Key Lookup Limit
Advanced Option 60

L

List Constraints 112
Load Requests
export/import file format 391, 403
Load tab, Personal Options 438, 439
DB Alias 438
exception tables 438
path to DBMS Loaders 439
LOB columns
Display options 433
Logon tab, Personal Options 448, 449
always ask for password 449
always fail connection 449
connection string 449
for Optim Directory 448
set passwords 449
specify User ID 449
test database connection 449
Long Object Names 299
Lookup Function 141

M

Macros, generating unique file names 471
Main Window
auto size 433
maximum components 433
Maximum
fetch rows 432
history lists 433
menu entries 433
visible components 433
Messages
errors 436
informational 435
limit lines to display 436
resetting 434
set font style 435
show or hide 436
warnings 436

N

Notify
Action Request 33
Notify tab, Personal Options
Send Test eMail 470

O

Object Name Highlighting 442

Objects
list of properties 182
Open
delete an object 30
select/open an object 29
specify a Pattern 27
using the Open dialog 25
Operators 184
Optim Directory 37, 43
connect or disconnect 38
specify DB Alias 194
storing objects 2, 37
Optim Directory Editor
select a code page 40
view DBMS details 40
view server details 43
view/change connection string 42
Optim Security
Access Control Domain 491
Access Control List 491
File Access Definition 497
Optim Server 1, 429
Options Menu
Point and Shoot 111
selecting commands 11, 17
Oracle
allocation percent 198, 199
default tablespace 199
Oracle Sequence Function 137
examples 137

P

Personal Options 427, 470
actions 461
archive 458
browsing files 456
configuring 427
confirm 431
creating objects 440
database 466
DBMS loaders 438
display 432
editing data 452
email notification and configuration 468
error defaults 435
MBCS Roundtrip Processing 466
printers, language options 464
removable media 460
scheduling monitor 437
servers 450
set general defaults 429
to logon 448
Point and Shoot
browse window 111
convert to a file 117
create a Row List File 483
Editor browse window 110
file options 74
find specific rows 115
format 111
indicator 51
Options menu 111
primary keys 109
processing rules 110
toolbar buttons 111
Point and Shoot *(continued)*
using the editor 109
Point and Shoot Editor
join multiple tables 113
stacked tables 114
Populate Column Maps
matching 249
merge type 249
processing sequence 250
select one 251
Prefix/Suffix dialog 22
Primary Key Editor
menu commands 208
populate the editor 205
respecify base creator ID 210
unique indexes 209
Primary Keys 203, 213
convert to generic 210
create database keys 206
creating 204
defining relationships 213
editing 207
explicit and generic 203
export/import file format 361
propagate value 138
required for database tables 203
select a table 205
select to edit 206
switch to DBMS keys 417
Printer Options 464, 465
Product Configuration File
set default directory 430
Propagate Function 138
Propagate, primary key value 138

R

Random Function 135
Random Lookup Function 147
Redisplay Results, File menu command 17, 35
Refetch rows 113
Relationship Editor
child expression 218
convert to generic 224
edit parent/child columns 221
menu commands 219
parent expression 218
populate the editor 216
respecify base creator ID 225
select columns 220
specify column values 219, 221
Relationship Index Analysis 106
Relationships 213, 227
concatenated expressions 139, 222
creating generic 224
data compatibility 226
data driven 223
editing 217
explicit and generic 213
export/import file format 362
extended 213, 417
find related to a table 29
for browsing file 316
for joining tables 114
restrictions 219
Relationships *(continued)*
select to browse database relationship 216
select to edit Optim relationship 216
status of 58
use new relationships 59, 63
Removable Media, Personal Options 460
default segment sizes 460
Replace UDTs, Personal Option 442
Replace with Database Indexes 100
Report Requests
export/import file format 403, 407
Report Retention, Personal Options 463
Reset Messages 434
Restart/Retry 253, 259
information in Control File 253
pending process attributes 255
specify process parameters 256
specify SQL ID 254
Restore Requests
export/import file format 407, 410
Retained Process Reports dialog 35, 36
Row List Files
data formats 483
external row lists 483
in an Extract Request 485
Rows meeting Criteria 112

S

Schedule Utility
edit a scheduled request 283
Scheduling Editor 288, 291
hold a job 285
review scheduled jobs 288
Scheduling Monitor 291, 299
active jobs 293
completed jobs 294
directory path for 437
job step results 295
monitor job processing 283
process reporting 297
show/hide monitor 294
start immediately 437
startup options 437
view scheduled jobs 291
Scheduling Options 437
Segment
naming 460
size 460
Select All Rows 112
Select Tables
find related tables 56
levels to search 56
show only 55
Selection Criteria
guidelines for using 80, 109
logical operators 83
relational operators 83
SQL WHERE Clause 82
Semantic aging, definition 259
Sequential Function 136
examples 136
Serial Function 136
Server Options 450, 451
always ask for password 451
check logon 451

Server Options *(continued)*
Optim Server 450
User ID, Password, Domain 450
Show
excluded rows 112, 304
SQL 112
unmatched columns 304
Show Steps analysis 108
Shuffle Function 149, 151
Side Label Display
description of 308
switch to columnar 307
Sort 111
Special Characters
Caps mode 430
underscore as SQL LIKE 429
Special register 129, 132, 173, 503
SQL ID
browse a File 302
restart/retry a process 254
Stacked Tables
browsing file 317
in the Table Editor 114
Status
of Column Maps 236, 248
of object definitions 414
of relationships 58
restart/retry pending process 254
rows in a Control File 305
Storage Profiles
export/import file format 362, 367
Substitution variables
Access Definition Editor 71, 73
Substring Function 135
examples 135, 222
Suffix/Prefix dialog 22
SybTimeOverflow 173

T

Table Access, Advanced Option 60
Table Map Editor
Column Map ID 127, 235
Column Map status 236
create a new Column Map 248
defaults for unused objects 235
edit a Column Map 247
list Column Maps 248
menu commands 235
populate with Column Maps 249
select Column Maps 248
select objects 244
Table Maps 229, 252
always view 412
Column Maps 229, 247
create objects 413
creating 230
editing 234
export/import file format 367, 369
merging 245, 246
select to edit 232
specify source objects 232
Table Specifications
Archive Actions 85
Archive Index 100
columns 77
File Attachments 102
Table Specifications *(continued)*
filtering on/off 113
selection criteria 79
sort columns 84
SQL 82
Tools menu 100
Toolbar
add buttons 14
customizing 13
moving a button 14
remove or move buttons 14
Tools Menu
selecting commands 17
Trace File 430
Traversal Path
child as Start Table 64
for extracting data 57
multiple children 66
multiple cycles 67
multiple parents 64
multiple relationships 65
option 1 59
option 2 59
parent to child 63
referential cycles 68

U

UDT 240
see User Defined Types 442
Uncommitted Data, Extract 53
Unjoin Tables 317
User Defined Types
Replace UDTs, Personal Option 442
tab, Table Map 240
Utilities Menu
selecting commands 10

V

Variable data types 184
Variables
AA delimiter 86
AD delimiter 86
default values 72
prompt string 73
SQL syntax 72
variable delimiter 79, 82

W

Warn on Cascade 467

Printed in USA