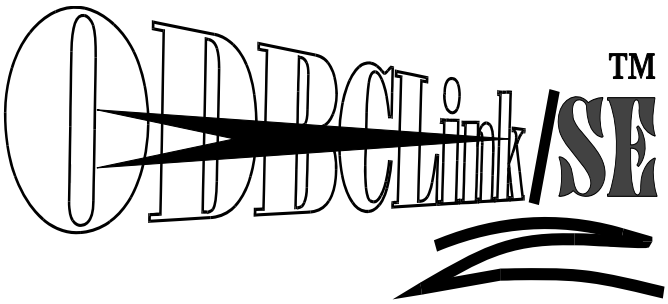

Reference Manual

Version E.56

M.B. Foster Software Labs Inc.

Head Office: 82 Main Street South
Chesterville, Ontario
K0C 1H0 Canada
Telephone: (613) 448-2333
Fax: (613) 448-2588

NOTICE

The information in this manual is subject to change without notice.

LIMITATIONS ON WARRANTIES AND LIABILITY

M.B. Foster Software Labs makes no warranties, either express or implied, regarding this manual or the computer software package described in this manual, its merchantability or its fitness for any particular purpose. The exclusion of implied warranties is not permitted by some states.

COPYRIGHT

ODBCLink/SE offers the following features:

- ODBC Level 2 compliance (except SQLBrowseConnect, SQLExtendedFetch and SQLSetPos)
- Access to an IMAGE/SQL database environment (DBE)
- Access to an ALLBASE/SQL DBE
- Access to a single DBE
- Support for 16 and 32 bit desktop clients

The additional features in the DataExpress/ODBClink product are:

- Native read/write access to TurboIMAGE databases - Why add the overhead of IMAGE/SQL when you can go direct
- Support for Serial and/or Modem connections - for traveling or home office employees
- Read/Write access to KSAM files - read and write KSAM files from your desktop application.
- Read/Write access to MPE files - read and write MPE files from your desktop application.
- Support for PowerHouse PDL dictionaries and subfiles - helping you to protect your investment in Cognos products.
- Support for TPI keyword indexes used by Omnidex and Superdex - helping you to protect your investment in these products.
- Read/Write access to Suprtool files
- Access to Oracle databases on HP3000 and HP9000 platforms
- Access to Multiple DBE's - multiple listeners not required, but they are supported.
- Access to HP intrinsics and procedures via a remote procedure call mechanism
- Enhanced security via field level access controls - making true the statement "open but secure".

Require access the legacy data structures? Upgrade to DataExpress/ODBCLink before December 31, 1998, and receive 35% off the full retail price!

Please contact M. B. Foster Associates at 613-448-2333 or 1-800-ANSWERS (800-267-9377) or use the enclosed faxback sheet to order a fully functional evaluation copy of ODBCLink.

ODBCLink/SE FAX BACK
to
M. B. FOSTER ASSOCIATES LIMITED
613-448-2588

NAME_____

TITLE_____

COMPANY_____

ADDRESS_____

CITY/STATE/ZIP_____

COUNTRY_____ E-MAIL_____

PHONE_____ FAX_____

Please check the features that you are most interested in:

- ❑ Serial/Modem connection
- ❑ MPE file access
- ❑ PowerHouse PDL support
- ❑ Oracle access
- ❑ Multiple DBE access
- ❑ Remote Procedure Call Support
- ❑ KSAM file access
- ❑ TurboIMAGE access
- ❑ Suprtool file access
- ❑ TPI interface support
- ❑ Enhanced security (field level)

I am using the following desktop product(s):

- ❑ MS Access
- ❑ MS Query
- ❑ Impromptu
- ❑ Lotus
- ❑ Visual Basic
- ❑ Paradox
- ❑ Axiant
- ❑ Excel
- ❑ Fox Pro
- ❑ Jetform
- ❑ PowerBuilder
- ❑ Crystal Reports
- ❑ Other_____

❑ 16-bit ❑ 32-bit

❑ **I am interested in an upgrade, please send a demo immediately!**

HPCPUNAME: _____
HPSUSAN NUMBER: _____
Media Required: ❑ DAT ❑ Tape

IN A HURRY? GET YOUR QUESTIONS READY AND CALL 1-800-ANSWERS (267-9377) NOW!

We can also be reached at:

PHONE: 613-448-2333
FAX: 613-448-2588
sales@mbfoster.com

Table of Contents

THIS PAGE INTENTIONALLY LEFT BLANK

Section 1
Overview and Specifications

ODBCLink/SE Overview

ODBCLink/SE is an implementation of Microsoft's Open Database Connectivity (ODBC) interface that enables Microsoft Windows based applications and tools to access IMAGE/SQL on the HP3000 and ALLBASE/SQL on the HP3000 and HP9000, in a client/server environment.

In this environment, application developers and end-users can take advantage of the PC's graphical user interface (GUI) and processing power, while relying on the security, integrity, and database management capabilities of ALLBASE/SQL and IMAGE/SQL.

ODBCLink/SE runs under MPE/iX 5.0 or higher on the HP3000 server, or under HP-UX Versions 9.x and 10.x on the HP9000 Series 7xx and 8xx servers. ODBCLink/SE runs under Microsoft Windows 3.1 and 3.11, Windows95, or WindowsNT, on the client. Connection via Winsock is available in 16 and 32-bit versions, regardless of the host environment.

ODBCLink/SE can be used in two ways: either by direct calls to the Windows DLL from a Windows program (Foxbase, etc.) or else through an ODBC-compliant application such as Microsoft Access, Visual Basic, etc.

ODBCLink/SE supports the ODBC LongVarBinary data type for binary large objects (BLOBs), such as compressed photographs or document images. These are stored on the host as ALLBASE/SQL LongVarBinary items.

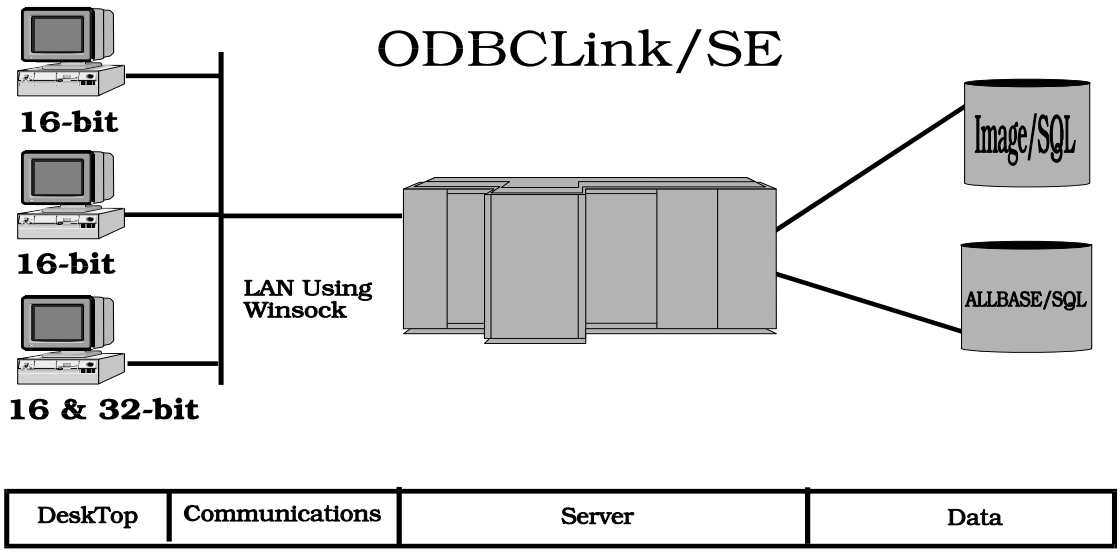

Figure 1

ODBC Compliancy Level

ODBCLink/SE is ODBC Level 2 with the following exceptions:

SQLBrowseConnect, SQLExtendedFetch, SQLSetPos are not supported. Scalar Functions are not support. *For further details on Supported Functions see Page 33*

SQL_BIT, SQL_TINYINT and SQL_BIGINT data types are not supported.*For further details on Supported Data Types see Page 36*

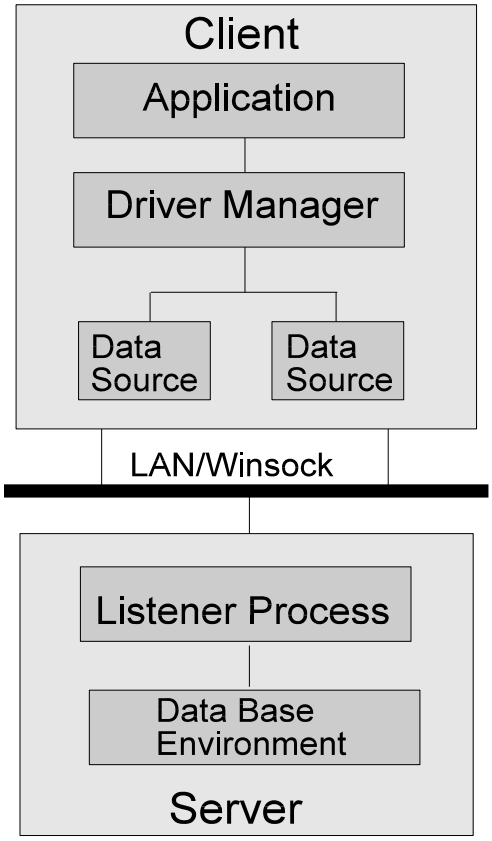

Figure 2

Client Environment

ODBCLink/SE and application software reside on the PC client. With ODBCLink/SE and applications software you can develop SQL applications, generate reports, and query ALLBASE/SQL or IMAGE/SQL on the database server. The tested client applications include:

Cognos Axiant	Cognos Impromptu
Jetform	Lotus 123
MS-Access	MS-Query
Paradox	PowerBuilder
Crystal Reports	Visual Basic
Visual C++	Visual FoxPro
and many more	

Server Environment

The networked HP 3000 Series 900 with ALLBASE/SQL and/or IMAGE/SQL or the HP9000 Series 7xx or 8xx with ALLBASE/SQL provide the relational database environment on the server. Security is provided by the MPE/iX or HP-UX logon system and ALLBASE/SQL or IMAGE/SQL.

The Connecting Link

ODBCLink/SE links the client to the server with a collection of dynamic link libraries (DLLs) and other files that reside on the client and run under Microsoft Windows 3.11, Microsoft Windows95 or Microsoft WindowsNT. ODBCLink/SE routes requests, made from the PC client application, over the network to an ALLBASE/SQL or IMAGE/SQL database on the database server, and returns replies to the client application.

A listener process establishes the connection between the PC client application and the target database on the server. The listener works with HP ThinLAN 3000/iX on the HP3000 or ARPA Services on the HP9000.

Microsoft ODBC Overview

ODBC applications make functions calls into ODBC.DLL (the Microsoft 16-bit ODBC driver manager), or ODBC32.DLL (the 32-bit driver manager). Since a single application can make calls to more than one ODBC database (e.g. Sybase, Microsoft SQL server) the driver manager is in charge of routing ODBC calls to the appropriate driver. The ODBC.INI file contains a list of drivers and associated data sources (one driver may have several data-sources). In Windows 95 or NT, some of this information is kept in the system registry.

To connect to an ODBC database, you call SQLDriverConnect (or you let your ODBC-compliant application call it for you) and specify your data-source name in the connection string variable, which is in the form "DSN=data_source". The driver manager reads ODBC.INI or the registry, to find out which driver handles that data source and passes the call on to the driver. The driver opens the communication channel and does all necessary initialization.

The ODBC.INI file (or the ODBC System Registry) is maintained through the ODBC Control Panel application ODBCADM.EXE , or the 32-bit ODBC Control Panel application ODBCAD32.EXE. The control panel application will call the setup DLL for the driver it is accessing (ODBCLS16.DLL or ODBCLS32.DLL in the case of ODBCLink/SE) which will update ODBC.INI or the system registry.

ODBCLink/SE System Requirements

HP3000 Server Requirements:

Operating System Requirements:
MPE/iX Version 5.0 or Version 5.5

Database Requirements :
IMAGE/SQL Version B.G2.03 or later
ALLBASE/SQL Version A.G2.00 or later

Network Requirements:
ThinLANLink/XL must be configured and turned on.

HP9000 Server Requirements:

Operating Systems Requirements
HP-UX Version 9.0 and Version 10.0

Database Requirements:
ALLBASE/SQL Version A.G2.01 or later

Network Requirements:
ARPA Services must be configured and turned on.

Client Requirements:

Platform:
486 processor at a minimum, preferably a Pentium
8 mg memory at a minimum, preferably 16 mg
5 mgs of free disc space

Operating Systems:
Windows 3.1
Windows 3.11
Windows 95
Windows NT Version 3.51
Windows NT Version 4.0

Network Requirements
Winsock or other TCP/IP software

Software provided:
Server program for the HP3000 or HP9000 as appropriate
Client programs both 16 & 32-bit platforms.

Field types supported:

All data types supported by Allbase and Image/SQL. *For details see Supported Data Types - Page 36.*

SQL Commands supported:

ODBCLink/SE supports the complete ALLBASE/SQL and IMAGE/SQL syntax dynamic SQL. *For details see Supported Functions - Page 33*

THIS PAGE INTENTIONALLY LEFT BLANK

Section 2
Preparing the Database Server

Before you install ODBCLink/SE on the PC client, perform the following steps to prepare the database server:

1. Verify the software version numbers.
2. Verify the Connection Assurance Parameters.
3. Starting and Stopping the listener.
4. Gather information.
5. Create the DBEnvironment

1. Verify Software Version Numbers

* If you are using ALLBASE/SQL, version A.G2.00 or higher is required.

  On the HP3000 enter:

  ```
  : SQLVER.PUB.SYS
  ```

  On the HP9000 enter:

  ```
  $sqlver
  ```

* If you are using IMAGE/SQL, the version number should be B.G2.03 or higher. Type:

  ```
  :IMAGESQL.PUB.SYS
  >>exit
  ```

NOTE: If both ALLBASE/SQL and IMAGE/SQL are installed, both must be either the respective version state above or newer in order to use ODBCLink/SE.

2. Verify the Connection Assurance Parameters

When the connected PC client application terminates abnormally or aborts, connection assurance parameters, set under MPE/iX, determine the length of time that the server takes to time out the connection. Because the server process may be holding locks when the PC aborts, it is important that these parameters are set properly.

Your network administrator should verify that the settings for Connection Assurance Interval and Maximum Connection Assurance Retransmissions parameters are set appropriately.

The Connection Assurance Interval specifies the amount of time between each polling event. The default value is 600 seconds.

The Maximum Connection Assurance Retransmissions specifies the number of retries the server is polled after a PC client abort has been detected by the polling event. The default value is four times.

After the connection abort, the server polls the connection five times (the polling event plus four retries). The connection times out after 4 ten-minute intervals plus the amount of time between the polling event and the first retry after the connection aborted.

To change the network assurance parameters, you must bring the network down. The sequence of commands used to set network assurance parameters are as follows:

```
:HELLO MANAGER.SYS
:NMMGR
```

Next, press the function keys in the sequence shown in the following example (these function key labels are for MPE/iX 5.0):

```
F1 Open Config
F2 NS
F2 Unguided Config
F1 Go To Netxport
F2 Go To GPROT
F2 Go To TCP
:
```

Modify values for Connection Assurance Interval (e.g. 60) and Maximum Connection Assurance Retransmissions (e.g. 2).

```
:
F6 Save Data
```

NOTE: Connection assurance parameter values control polling times on all active server LAN connections (not just ODBCLink/SE connections). More frequent polling uses more server CPU time and increases LAN traffic.

3. Starting and Stopping the Listener

To start the listener on the HP3000 data base server, enter:

```
STREAM ODBCJOB.ODBCSE.SYS
```

NOTE: PM and NM capability is required to start the listener job, therefore it is normal for the job to be started by MANAGER.SYS.

To start the listener on the HP9000 database server, enter:

/usr/bin/odbcse/odbclnse server On a 9.x server
/opt/allbase/bin/odbcse/odbclnse server On a 10.x server

NOTE: You should start the listener as root user-id preferably in your system startup file.

NOTE: When the server is started it makes a call to sqlver and puts the output into a file called odbcver for HP-UX and verstr for MPE in the same directory that the server is running from. For this reason, upon startup, it may take a few seconds for the program to return. The file is used to determine the DBMS version for a SQLGetInfo call. If the file is deleted the SQLGetInfo call for DBMS version will return “UNKNOWN”

To determine whether or not the listener job is running:

On the HP3000 database server, enter:

SHOWJOB JOB=@J

The resulting display will be something like this:

```
JOBNUM  STATE   IPRI  JIN  JLIST  INTRODUCED   JOB NAME
#Jnnn   EXEC          10S  LP     THU 12:01P   ODBCLNSE,MANAGER.SYS
```

On the HP9000 database server, enter:

ps -ef | grep odbclnse

The resulting display will be something like this:

```
root   7223     1  0   15:37:15  ?       0:00  /usr/bin/odbcse server
paw    7226  7223  0   15:40:56  ?       0:00  /usr/bin/odbcse server
root   7238  7230  1   15:41:29  ttys2   0:00  grep odbcse
```

The first line shows the server running. The second line shows a user with a login of “paw” is using the server and the last line shows the command you just entered.

To stop the listener:

On the HP3000 database server, enter:

ODBCLNSE.ODBCSE.SYS STOP

NOTE: If there is a connection open when the stop command is issued the process will not be stopped. If you are not able to stop the listener in the normal way, or if you want to terminate active ODBC client sessions, you may abort the listener Job by using “abort” parameters in place of “top” or alternatively you can use the :ABORTJOB command

On the HP9000 database server, enter:

/usr/bin/odbcse/odbclnse stop (or abort)	On HP-UX 9.0
/opt/allbase/bin/odbcse/odbclnse stop (or abort)	On HP-UX 10.0

Note that in Unix, the STOP command will prevent any new connections but will not affect connections that were active at the time the command was issued.

You may also use the following command to stop the listener. It is, however, strongly suggested that you use the “odbcse stop” command to stop the listener.

kill -16 “pid” where “-16" is the “siguser1" and “pid” is the process ID number

In the example above, if the “pid” was 7223" then “kill -16 7223" would stop the server. Using a “kill -16" is less dangerous than using “kill (pid)”. If you issue a kill command without the -16 and enter the wrong “pid” number then you may stop a server process that does not belong to you. Using “kill -16" will prevent this from happening.

4. Gather Information

When you install ODBCLink/SE on the PC client, you will need database server information. Now is a good time to note it. The database administrator will have information about the database.

- Host or node name of the HP3000 or HP9000 database server, or its IP address (For example: HP3000 or HP9000 or 192.9.2.10)

  To obtain the host name used by the ARPA interface:

  - On the HP3000 check the HOSTS.NET.SYS file. If the hosts file does not exist, check the domain name service file. This file contains the names of the systems that have hosts files.
  - On the HP9000 check the /etc/hosts file. If the hosts file does not exist, check the

domain name service file, /etc/resolv.conf. This file contains the names of the systems that have hosts files.

- Fully qualified name or the absolute pathname of the DBEnvironment on the server:
  (For example: PartsDBE.SomeGrp.SomeAcct on the HP3000 or
  /usr/hpsql/sampledb/PartsDBE on the HP9000)

- Logon string and password used to connect to the DBEvironment:
  (For example: SomeUser/passwd.SomeAcct/AcctPass,SomeGrp/GrpPass on the HP3000 or
  User1,passwd on the HP9000)

5. Create the DBEnvironment

For details see - Appendix B - Creating a DBEnvironment - Page 79

THIS PAGE INTENTIONALLY LEFT BLANK

Section 3
Installing ODBCLink/SE on the PC

ODBCLink/SE is contained in self-extracting archive files:

ODBCCL16 for the 16 bit driver and
ODBCCL32 for the 32 bit driver.

Setting up ODBCLink/SE involves:

- Downloading the appropriate ODBCLink/SE file to a temporary sub-directory.
- Extracting the files on your PC client.
- Running the SETUP program to install the driver.

Once the driver has been successfully installed, the sub-directories created as part of the install may be deleted.

Downloading ODBCLink/SE to the PC

In the following section, replace *Server* with the name of your HP3000 or HP9000 server and *nn* with 16 or 32 depending on which driver is being installed. *For details see Notes on 16 or 32-bit Access - Page 19.*

Using Reflection® software:

From your PC:

1. Create a directory:

   MD ODBCSEnn

2. Move to the new directory:

   CD ODBCSEnn

3. Start Reflection and copy the appropriate ODBCLink/SE self-extracting file to your PC.

For the HP3000 server, press Alt/Y to open the Command Window and enter

receive \odbcse*nn*\odbccl*nn*.exe from odbccl*nn*.odbcse.sys binary

For the HP9000 server use any login as the Username:

For HP-UX Version 9.x

receive \odbcse*nn*\odbccl*nn*.exe from \usr\bin\odbcse\odbccl*nn* binary

For HP-UX Version 10.x:

receive\odbcsenn\odbcclnn.exe from \opt\allbase\bin\odbcse\odbcclnn binary

Using ARPA Services (FTP):

From your PC:

1. Create a directory:

   MD ODBCSEnn

2. Move to the new directory:

   CD ODBCSEnn

3. Copy the appropriate ODBCLink/SE self-extracting file to your PC

   For the HP3000 server, use MANAGER.SYS,ODBCSE as the Username

   C:\ODBCSE*nn*>
   ftp *Server*
   binary
   get odbccl*nn* odbccl*nn*.exe
   quit

   For the HP9000 server use any login as the Username:

   C:\ODBCSE*nn*>
   ftp Server
   binary
   get /usr/bin/odbcse/odbccl*nn* odbccl*nn*.exe (for HP-UX V9.x)

   or

   get /opt/allbase/bin/odbcse/odbcclnn odbcclnn.exe (for HP-UX V10.x)

Using NS Services (DSCOPY) on the HP3000:

From your PC:

1. Create a directory:

   MD ODBCSEnn

2. Move to the new directory:

   CD ODBCSEnn

3. Copy the appropriate ODBCLink/SE self-extracting file to your PC

   dscopy -r -F -B -L256 *Server*#manager.sys,odbcse# odbccl*nn* odbccl*nn*.exe

Extracting the ODBCLink/SE file

From your PC, type the following to extract the ODBCLink/SE files from the self-extracting file (replace *nn* with 16 or 32 depending on which driver is being installed):

C:\CD \ODBCSEnn
C:\ODBCSEnn> ODBCCL*nn*.EXE

NOTE: If you are installing ODBCLink/SE on a number of PC's, after extracting the files you might wish to create installation disks by copying the extracted files onto a diskette.

Running the SETUP program

Where *nn* is 16 for the 16 bit driver and 32 for the 32 bit driver.

A problem associated with Microsoft ODBC 2.10 setup causes the setup on a WindowsNT 4.0 system to install as if it was a Windows95 system. We have therefore created two information files to be used with the 32-bit setup program. A batch file copies the appropriate file to odbc.inf **depending on the parameters given.**

To allow the appropriate file to be installed do the following from the setup directory (eg \ODBCSEnn). It is preferable that this procedure be performed at the DOS prompt so that the window will not disappear before you have a chance to read it. The process can however be performed from the Program Manager.

On a WindowsNT system run "infsetup wnt"
On a Windows95 systems run "infsetup w95"

The batch file will read the parameter entered and copy the appropriate file to odbc.inf. Entering infsetup with no parameter specified will simply describe the usage but **will not copy the file.**

NOTE: We have left the original odbc.inf file on the disk so if infsetup is not run, the install will still work in most cases. If the wrong options is entered or your are installing on both a WindowsNT and Windows95 platform, simply re-run the infsetup.bat file with the proper option.

Once the above has been accomplished proceed with the setup program as follows:

From the Program Manager:

Click on File then Run

enter \ODBCSE*nn*\SETUP.EXE

From DOS:

enter \ODBCSE*nn*\SETUP.EXE

The Driver Setup Program will display this window.
When you click on Continue, the Install Drivers window appears.

Figure 2

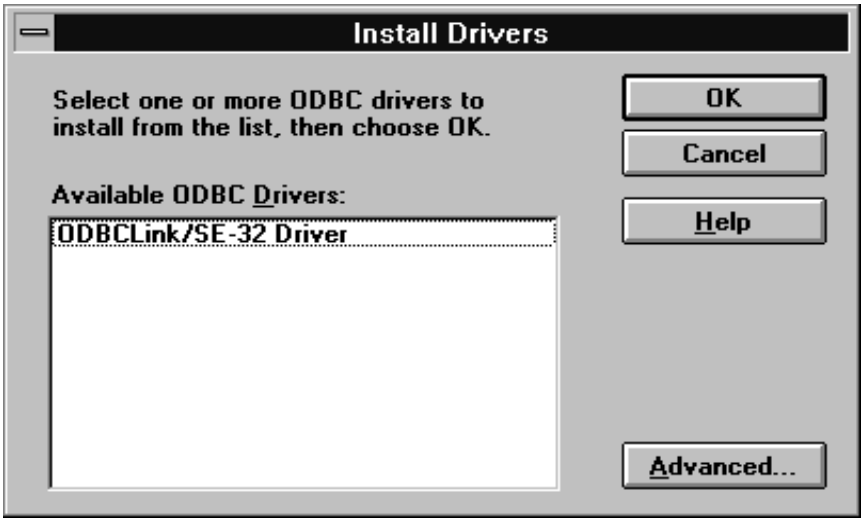

Figure 3

NOTE: You MUST click on a driver name, even if only one driver name appears in the list, then click OK. The driver names for ODBCLink/SE are:

"ODBCLink/SE-16 Driver" for 16 bit applications
"ODBCLink/SE-32 Driver" for 32 bit applications.

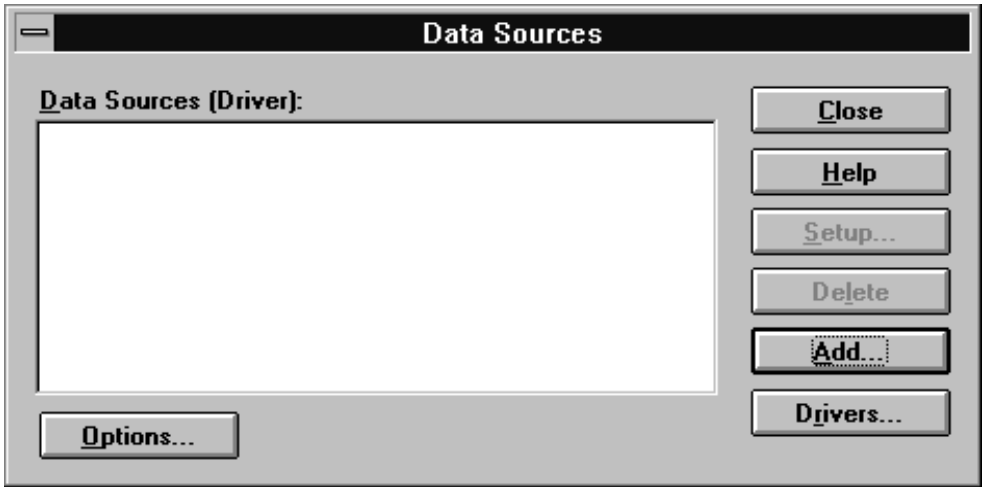

Figure 4

The driver is now installed. If this is the first driver you have installed, the Data Source name list will be empty.

You may now click on:

Close to end the installation.

Add to add a new Data Source *See Section 4 - Configuring Data Sources.*.

Setup to modify existing Data Sources S*ee Section 4 - Configuring Data Sources.*
Delete to delete the highlighted data source (16 bit only)
Drivers to do driver maintenance (16 bit only)
Options to establish ODBC tracing rules; *See Section 7 - Troubleshooting*.

Figure 5

The driver has been successfully installed. You may now create data sources for use in applications that use this driver. *See Section 4* for a complete description of creating data sources.

The sub-directories ODBCSE16 and ODBCSE32 are no longer required and may be deleted, if you wish. If you need to re-install a driver you can always download the files from the host.

Notes on 16 or 32-bit Access

On a 32-bit operating system (Win95 or WinNT) you can run both 16 and 32-bit versions of the ODBCLink/SE driver. The server software runs the same for both.

You should always use the 32-bit driver, if you are using a 32-bit operating system. *For details see Performance Considerations - Page 31.* In some cases however you may not have a choice, since a 16-bit application in Win95 or WinNT **cannot use a 32-bit driver.** If you want ODBC access from a 16-bit application on Win95 or WinNT, you must install the 16-bit version of ODBCLink/SE.

16 and 32-bit data-sources are maintained separately and can coexist. 32-bit data-source information is kept in the registry, and maintained by the 32-bit ODBC control panel application ODBCAD32.EXE. 16-bit data-source information is kept in the ODBC.INI and ODBCINST.INI files in your windows root directory (not the windows\system directory) and is maintained by the 16-bit control panel application ODBCADM.EXE. The 16-bit ODBC control panel application will show 32-bit data-sources, but the 32-bit ODBC control panel will **not** show 16-bit data-sources.

To distinguish between 16 and 32-bit data-sources, look at the driver name in the data-source window in the ODBC control panel. You will either see ODBCLink/SE-16 or ODBCLink/SE-32

You do not normally make entries manually into either the registry or the ODBC.INI file. However information has been provide in the manual to help you do this. *For details see Modifying the System Registry and ODBC.INI Files - Page 70.*

THIS PAGE INTENTIONALLY LEFT BLANK

Section 4
Configuring Data Sources

Before you access data defined to ALLBASE/SQL or IMAGE/SQL you must configure one or more *data sources* for each PC you wish to access your server.

A data source includes the name you wish to give for a database environment and the information necessary to find it.

You configure data sources with the "ODBC" application in the Windows Control Panel.

In Windows 3.11, double click on Main, Control Panel and ODBC

In Windows95 and WindowsNT 4.0 , click on Start, Settings, Control Panel and double click on ODBC.

In Windows NT V3.51, double click on Main, Control Panel and ODBC for 32-bit data sources.

For 16-bit data sources use the ODBC Administrator in the ODBC Program Group.

Data sources created in HP ALLBASE/SQL PCAPI can be migrated to ODBCLink/SE using the Data Set Migration Tool described later in this section.

Data Source Configuration

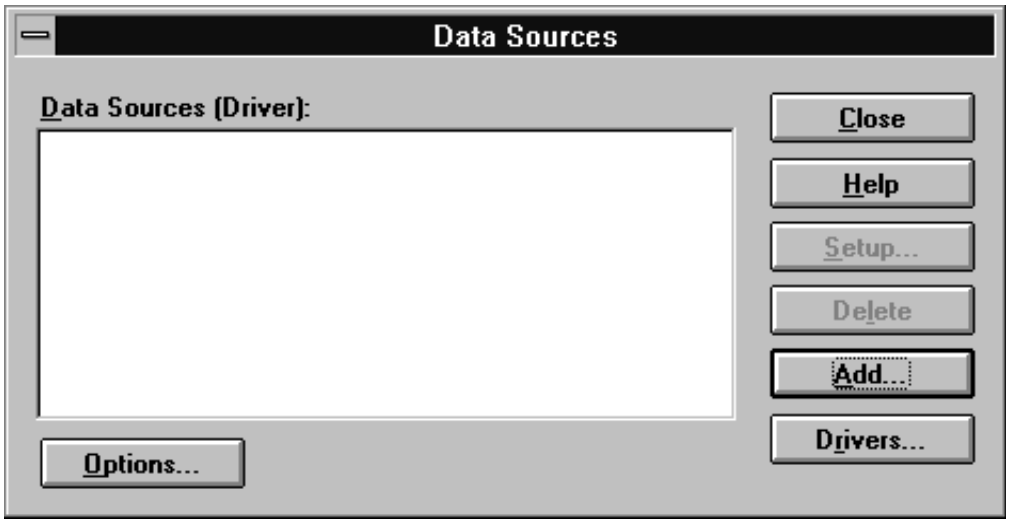

Figure 6

You may now click on:

Close to return to the Control Panel
Add to add a new Data Source
Setup to modify existing Data Sources
Delete to delete the highlighted data source (16 bit only)
Drivers to do driver maintenance (16 bit only)
Options to establish ODBC tracing rules; *See Section 7 - Troubleshooting*.

Choosing an ODBC Driver

You MUST click on a driver name even if only one name appears in the list. Click **Close**.

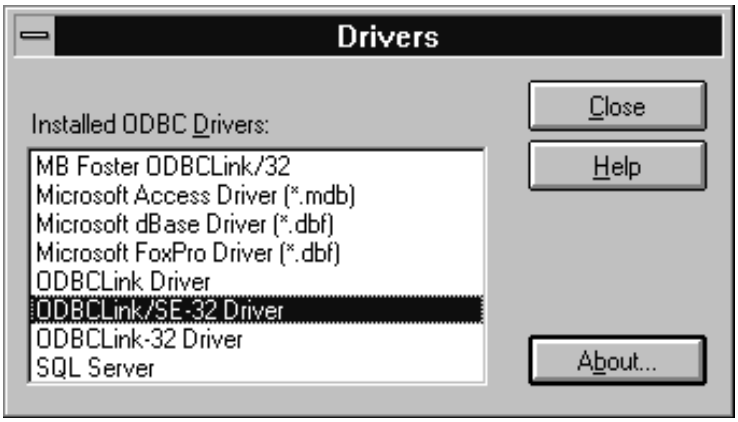

Figure 7

ODBCLink/SE Setup

On this screen you give the database environment a name and indicate where it is located. The Data Source Name may be any combination of alphanumeric characters including blanks.

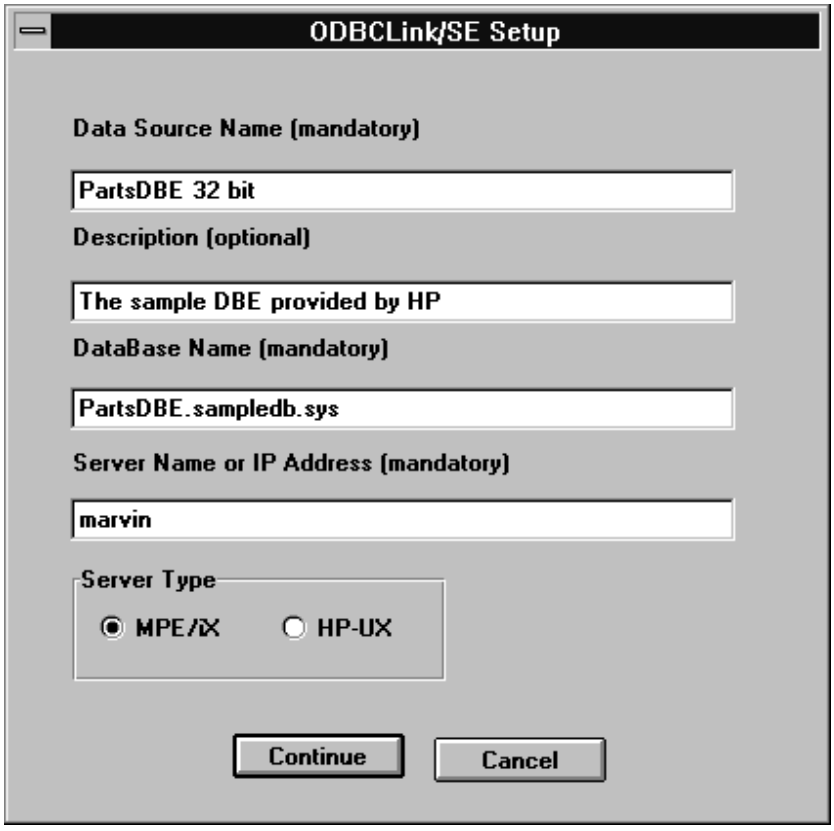

Figure 8

The maximum length is 30 characters. If 16 and 32 bit applications are being used on the same PC, it is useful to know which data source is for which driver; it may help to include '16' or '32' in the name. You may choose your own naming convention.

The Description is optional and may contain any string of characters you wish. It may be longer than the display window.

The Database Name is the name of the ALLBASE/SQL Database Environment you want to access with this data source. The name will usually include the location of the DBE as shown in the sample screen. If the logon provided in the next screen does not include the same logon group, the location MUST be provided. A sample HP/UX name might be:

/usr/users/data/PartsDBE

The Server Name or IP Address is the name or address of the computer where your database environment resides. If you don't know this name, contact your network administrator.

The Server Type radio button indicates the type of computer on which the DBE is located. Click on:

MPE/iX if it is an HP3000
HP-UX if it is an HP9000

Setup for MPE/iX

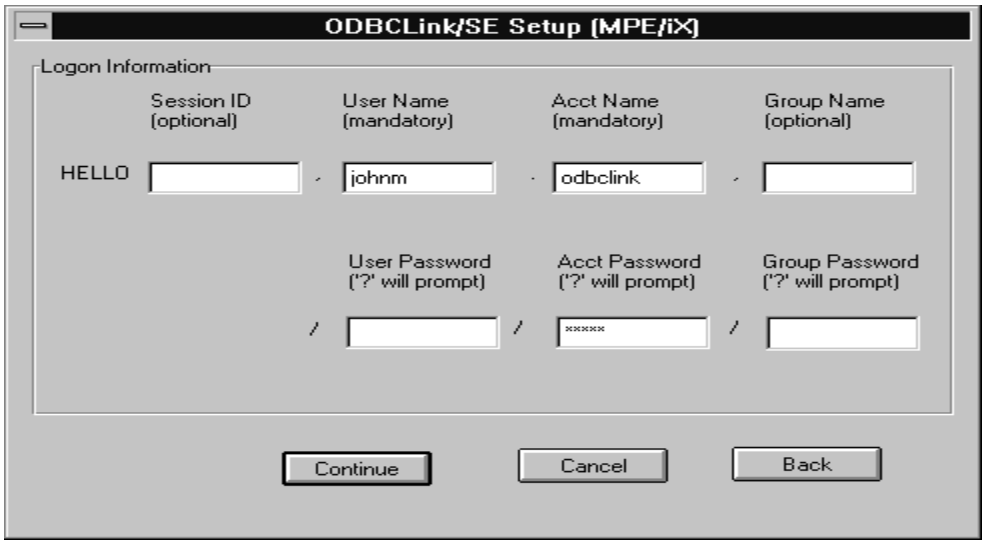

Figure 9

The SessionID, User Name, Acct Name, Group Name, User Password, Acct Password and Group Password entries correspond to those used if you were manually connecting to the HP3000 or HP9000. If you don't know this information, contact your System Manager.

If you want the client application to prompt for the passwords at run-time, enter a “?” in the password field. If you put a question mark in any password field, you will get a dialogue box and be prompted to enter the password **every time you run your application and request a connection.** For an attached table in MS-Access, ODBCLink/SE will **not** store the password anywhere within the application, so you will have to enter the password the first time you open the table. However, many applications, including Access, connect more than once to the same data-source, and if you have already entered the passwords and created a new connection, you will not normally have to enter it again for additional connections to the same data-source.

As you might expect, "mandatory" means you must enter a value. However, "optional" doesn't necessarily mean you don't have to enter a value; it means there may not be a value required.

For example, on the MPE/iX logon, "Session ID" is any 8 characters you choose, starting with an alphabetic character with no embedded blanks. You may choose not to enter a "Session ID". On the other hand, "Group Name" will default you to your "home group" so may be left blank; if your "User Name" does not have a "home group" this field is required or the connection will fail.

Setup for HP-UX

Figure 10

As you might expect, "mandatory" means you must enter a value. However, "optional" doesn't necessarily mean you don't have to enter a value; it means there may not be a value required.

For example, on the MPE/iX logon, "Session ID" is any 8 characters you choose, starting with an alphabetic character with no embedded blanks. You may choose not to enter a "Session ID". On the other hand, "Group Name" will default you to your "home group" so may be left blank; if your "User Name" does not have a "home group" this field is required or the connection will fail.

On HP-UX, the Login name and password are case sensitive.

If you want the client application to prompt for the passwords at run-time, enter a “?” in the password field. If you put a question mark in any password field, you will get a dialogue box and be prompted to enter the password **every time you run your application and request a connection.** For an attached table in MS-Access, ODBCLink/SE will **not** store the password anywhere within the application, so you will have to enter the password the first time you open the table. However, many applications, including Access, connect more than once to the same data-source, and if you have already entered the passwords and created a new connection, you will not normally have to enter it again for additional connections to the same data-source.

Setup Options

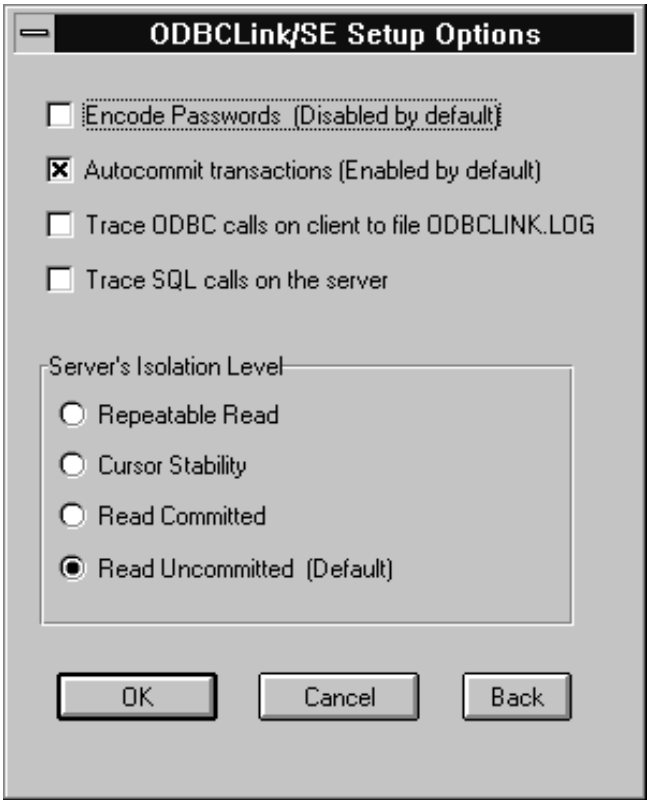

Figure 11

Encoded passwords: Check this box to encrypt the connection passwords in the ODBC.INI file or system registry. For security reasons once the passwords are encoded if you decide to un-click the box you must DELETE the current passwords first.

AutoCommit transactions: Check this box to have UPDATE and INSERT transactions committed as soon as they are executed. If this box is not checked, transactions must be explicitly committed by the client application.

It is recommended that AutoCommit be left enabled

Trace ODBC calls on client: Check this box to have client calls to ODBC logged in the file ODBCLINK.LOG on the PC. *See Section 7 - Trouble Shooting*.

Trace ODBC calls on server: check this box to have server calls to ODBC logged in the file ODBCLOG.ODBCSE.SYS on the HP3000 or /tmp/odbclog on the HP-UX. *See Section 7 - Trouble Shooting.*

Isolation levels allow you to control the degree of concurrency by regulating the extent to which operations performed by one user in a multi-user environment can be affected by operations performed by other users. *See the ALLBASE/SQL Reference Manual for details.*

The recommended isolation level is “RU” for Read Uncommitted. This minimizes the number of locks that are held on your database.

Select Translator

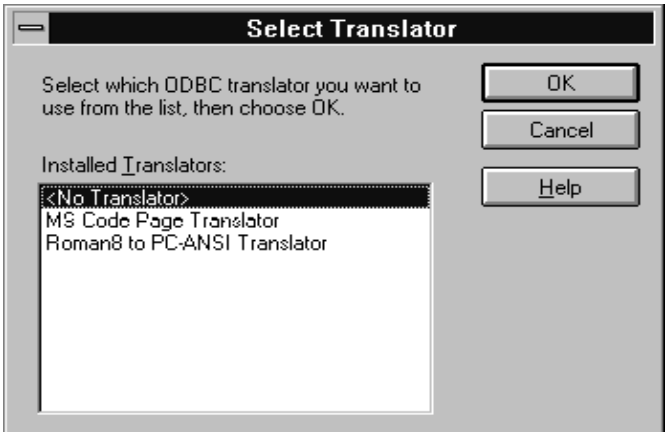

Figure 12

ODBC Translator

Select “Roman8 to PC-ANSI Translator” if you want extended characters on your HP3000 (such as foreign accented characters) to appear correctly on your PC. The translation goes both ways. For example; extended characters are entered on the PC and sent to the host in an SQL update or Insert operation will be translated.

Successful Completion

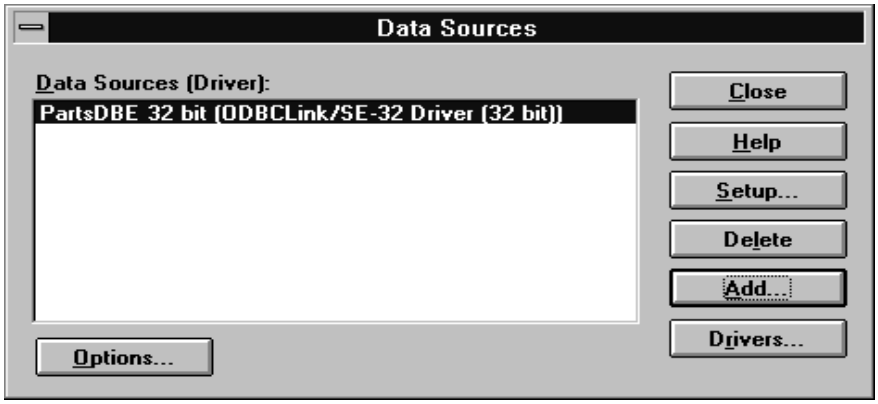

Figure 13

The new data source is now ready for use in your client application.

It may appear that the "32 bit" included in the data source description is redundant, as it is also included in the driver name. However in a mixed environment; some clients show only the data source name. Adding “32bit” make it easier to distinguish between 32-bit and 16-bit drivers.

Some applications may allow you to go directly to the "Data Sources" menu without going to Control Panel to click on the ODBC Administrator. Check you client application documentation for details.

Migrating HP PCAPI Data Sources

Data sources created in the HP ALLBASE/SQL PCAPI can be migrated to ODBCLink/SE by running the data set migration tool DXNMIG16.EXE either from the RUN window or in an MS-DOS prompt window.

Figure 14

Click on all the data source names you wish to migrate and then click OK. Data sources will be migrated to ODBCLink/SE data sources with the same name. Copies of the PCAPI data sources will be made with the same name followed by "-orig".

When the copies of the originals are no longer required, they may be deleted with the 16-bit ODBC Administrator program.

Section 5
Application Development

If you are developing applications that use the ODBC interface, you may want to order the Microsoft Developer Network (MSDN) Professional (Level 2) from Microsoft. The MSDN contains the ODBC Software Developers Kit (SDK), which includes the ODBC API Reference, sample ODBC applications, and other technical information to assist you in developing your applications. There are also many good books written on the use of ODBC.

Using Stored Procedures

ODBCLink/SE implements all types of ALLBASE/SQL stored procedures, including those which return multiple-format results sets. You can list the stored procedures available to your ODBC session by calling SQLProcedures and you can get parameter information for stored procedures by calling SQLProcedureColumns.

Stored procedures return data either through bound parameters or through one or more results sets. To get a return value that is a procedure parameter, prepare an SQL statement (with SQLPrepare) of the form:

{?=call owner.procedue (?,?..)}

Then bind the parameters with SQLBindParameter, specifying an output parameter where appropriate (SQLProcedureColumn will tell you which are input, which are output, and which are input/output parameters). Then call SQLExecute. The return value(s) will be copied to the storage locations that you specify. Note that all stored procedures return an integer return status called RETURN_STATUS.

If your stored procedure also returns one or more result sets, you call SQLNumResultCols, SQLDescribeCol, SQLFetch, SQLGetData in the normal way you would for any Select statement. If the procedure returns more than 1 result set, you call SQLMoreResults when you get to the end of the first result set, and, if successful, you can then retrieve another result set. It is permissible to call SQLMoreResults() at any time to switch to a different result set.

Performance Considerations

Most people find ODBC access to their server very fast. You should however be aware of the following:

- If you are reading and sorting a large table, the host may not return control to the client until the entire data-base has been read and sorted. If this occurs, even restarting your PC will have no effect and you will have to manually abort the server, or terminate the connection with the ISQL “Terminate User” command.

- When you are opening a large table in MS-Access, the program typically displays a screenful of data and then seems to wait for user input. In many cases, however, it will actually be downloading data to the client in the background. You should think about this when giving your users ODBC access to large tables on the server. You may give or withhold access to certain tables with the ALLBASE/SQL GRANT and REVOKE commands. There is no way currently of limiting the number of rows or the time the server is allowed to execute a request.

- As a rule of thumb, ODBCLink/SE will not complete any request faster than ISQL will, and could be considerably slower due to network overhead. If you are writing your own SQL, you may want to verify in ISQL how ALLBASE/SQL optimizes your queries by reading the SYSTEM.PLAN pseudo-table. *Refer to the ALLBASE/SQL manual for details.*

- If you are writing your own SQL to do multi-row inserts or updates, you can speed up your application by using dynamic substitution parameters. See the ODBC SDK manual (from Microsoft Press) for details.

Turning AutoCommit On/Off

AutoCommit “on” is the default value for all ODBC drivers. Turning AutoCommit on/off is normally accomplished by the application calling SQLSetConnectOption. The AutoCommit on/off setting in ODBC Administrator is there for historical reasons only and should not be used. Setting AutoCommit “off:” in the ODBC Administrator is not a good idea as the application can get confused. For instance, it can incorrectly assume that AutoCommit is on, and then emit SQL that never gets committed.

Supported Functions

The ODBC functions are listed here for application software developers. These functions are supported by the ODBCLink/SE driver at the time of publication of this manual. Refer to the README.HLP file included with ODBCLink/SE for further developments.

You can use the ODBC SQLGetFunctions call for a list of the supported ODBCLink/SE driver functions. Descriptions and conformance designations are listed in ODBC API Reference included in the Microsoft Software Development Kit (SDK).

Functions that are not completely implemented are noted by an R in parenthesis and are discussed below this list.

SQLAllocConnect
SQLAllocEnv
SQLAllocStmt
SQLBindCol
SQLFindParam
SQLCancel
SQLColAttributes (R1)
SQLColumns
SQLColumnPrivileges
SQLConnect
SQLDataSources
SQLDescribeCol
SQLDisconnect
SQLDriverConnect
SQLError
SQLExecute
SQLExecDirect
SQLFetch
SQLForeignKeys
SQLFreeConnect
SQLFreeEnv
SQLFreeStmt
SQLGetConnectOption
SQLGetCursorName
SQLGetFunctions
SQLGetInfo (R3)
SQLGetTypeInfo
SQLNumResultCols
SQLPrepare
SQLPrimaryKeys
SQLProcedureColumns
SQLProcedures
SQLRowCount

SQLSetConnectOption (R2)
SQLSetCursorName
SQLSetParam
SQLSetStmtOption (R4)
SQLSpecialColumns
SQLStatistics
SQLTablePrivileges
SQLTables
SQLTransact

R1: The following SQLColAttributes options are implemented; the others will receive a return value of 'Not Implemented'.

SQL_COLUMN_COUNT
SQL_COLUMN_NAME
SQL_COLUMN_TYPE
SQL_COLUMN_LENGTH
SQL_COLUMN_PRECISION
SQL_COLUMN_SCALE
SQL_COLUMN_DISPLAY_SIZE
SQL_COLUMN_NULLABLE

R2: The following SQLSetConnectOptions are supported. All other will return a “Driver not Capable (SQLState S1C00)”.

SQL_AUTOCOMMIT	1=ON (default) 0=OFF
SQL_OPT_TRACE	supported by Driver Manager
SQL_OPT_TRACEFILE	supported by Driver Manager
SQL_TRANSLATE_DLL	supported
SQL_TRANSLATE_OPTION	supported
SQL_TXN_ISOLATION	supported: Cursor Stability isolation level (CS) is not supported by SQLSetConnectOption but you may set it from the ODBC Administrator
SQL_TXN_READ_UNCOMMITTED	(Read Uncommitted (RU))
SQL_TXN_READ_COMMITTED	(Read Committed (RC))
SQL_TXN_REPEATABLE_READ	(Repeatable Read (RR))
SQL_TXN_SERIALIZABLE	(Note: Sets to Repeatable Read (RR))
SQL_TXN_VERSIONING	Not Implemented - DO NOT USE

For information on isolation levels, refer to the "Isolation Levels" section in this chapter.

R3: All SQLGetInfo Options are supported.

R4: SQLSetStmtOptions are not supported. The driver will return either a “Driver not Capable (SQLState S1C00)” or “Option value changed (SQLState 01S02)”.

Isolation Levels

Isolation levels allow you to control the degree of concurrency by regulating the extent to which operations performed by one user in a multi-user environment can be affected by operations performed by another user.

ALLBASE/SQL allows four different isolation levels:

- Cursor Stability (CS)
- Repeatable Read (RR)
- Read Committed (RC)
- Read Uncommitted (RU)

ODBCLink/SE uses the isolation level specified in the ODBC setup screen. The application can later change this by calling SQLSetConnectOptions with SQL_TXN_ISOLATION option.

The recommended isolation level is “RU” for Read Uncommitted. This minimizes the number of locks that are held on your database.

Refer to the ALLBASE/SQL Reference Manual for further information.

Supported Data Types

The following is a table of correspondence between HP data types and ODBC data types.

ALLBASE/SQL or Image or IMAGE/SQL Data Type	ODBC Data Type	Comments
CHAR, max length <255	SQL_CHAR	
CHAR, max length >=255	SQL_LONGVARCHAR	
VARCHAR, max length <255	SQL_VARCHAR	
VARCHAR, max length >=255	SQL_LONGVARCHAR	
BINARY, length <256	SQL_BINARY	
BINARY, length >=256	SQL_LONGVARBINARY	
VARBINARY <256	SQL_VARBINARY	
VARBINARY >=256	SQL-LONGVARBINARY	
LONG BINARY	SQL_LONGVARBINARY	
LONG VARBINARY	SQL_LONGVARBINARY	
INTEGER (32-bit)	SQL_INTEGER	
SMALLINT (16-bit)	SQL_SMALLINT	
DECIMAL (Internal representation is packed decimal)	SQL_DECIMAL	Character representation with leading sign and decimal point
Image Zoned (Z)	SQL_DECIMAL	Converted by Image/SQL
Image Packed (P)	SQL_DECIMAL	Converted by Image/SQL
Image I3 (48-bit integer)	SQL_DECIMAL	Converted by Image/SQL
Image I4 (64-bit integer)	SQL_DECIMAL	Converted by Image/SQL
FLOAT(24) or REAL or Image E2	SQL_REAL	
FLOAT(53) or DOUBLE PRECISION or Image E4	SQL_DOUBLE	
Image R2 (non-IEEE float)	SQL_REAL	Converted by Image/SQL
Image R4 (non-IEEE float)	SQL_DOUBLE	Converted by Image/SQL
DATE	SQL_DATE	6-byte with year, month, day in 2-byte binary fields
TIME	SQL_TIME	6-byte with hour, minute, second in 2-byte binary fields
DATETIME	SQL_TIMESTAMP	16-byte made up of year(2), month(2), day (2), hour (2), minute (2), second (2), fraction (4)
INTERVAL	SQL_CHAR	Format is “ddddddd hh:mm:ss.fff”

NOTE: When creating a table using CHAR and BINARY data types that are greater than 255 characters the resulting data types used will be LONGVARCHAR and LONGVARBINARY, which may not be the data type expected by the the application.

Note on Using BLOBs

It is strongly recommended that when creating an ALLBASE table for storage of BLOBs that you use a LONG VARBINARY column. ALLBASE will allocate storage space according to the actual size of the BLOB. Eg. Creating a table with a column defined as LONG VARBINARY (200000000) and writing a 10K BLOB to it will result in only 10K of space being used (not the maximum of 2 GB as specified when the column was created).

Although you can store a BLOB using a LONG BINARY column, this is not recommended, because ALLBASE will allocate storage space according to the specified column size for each BLOB regardless of the size of the actual data. Eg. Creating a table with a column defined as LONG BINARY(1000000) and writing a 10K BLOB will result in 1000000 bytes of space being used for each BLOB. You would quickly run out of space in your database.

THIS PAGE INTENTIONALLY LEFT BLANK

Section 6
Using ODBCLink/SE from ODBC Applications

After you have installed the drivers and created the data sources you are ready to use ODBCLink/SE in your applications. Each client application will have a slightly different way of connecting to the data sources you have set up.

In this section you will find examples of how to access data sources for many of the popular client tools used today.

Using Axiant

Importing Database Structure Information into Axiant via ODBCLink/SE

1. Start Axiant and create a new task

   (I) Start Axiant
   (ii) Select File, New
   (iii) You will be prompted "Connect new task to a repository?" Click YES and then in Connect repository click NEW
   (iv) Name the new repository
   (v) Click on prepopulate-NO then click OK.

2. Now you should have an empty task....to import the data from a database.

   (I) Select Tools - Import Data Definition
   (ii) Now Select the import method (probably ODBC) or ODBC with Qualifier.
   (iii) You will see a list of available Drivers (blanks indicate that Axiant does not see any drivers). Note : Axiant 1.2c and earlier is 16-bit and therefore requires a 16-bit driver.
   (iv) When you select your driver you will get a list of data sources.
   (v) Select the data source(s) that you want. Select “don’t prompt” so the applications won’t prompt you again for all of the data source information.
   (vi) Click on the Import key.

All the structure information from the database(s) belonging to the data source you gave should be imported. If your data source is set up to access more than one database then you will get a database domain for each database you have access to.

NOTE: You will need the full version of M.B. Foster’s ODBCLink to connect to multiple DBEs.

Creating the screen application

(I) Select View - List
(ii) Highlight the application icon and then select Objects, Create, Screen Applications. You should be in the "New Screen Creation" form.
(iii) Select the table that you want to access and then click on Next. If you wish to join two tables, select the other table and define the relationship; e.g.: 1:1 or 1:10 etc.
(iv) Add the screen items that you want from your table(s) and hit Next.
(v) Build the Application (click on Build). Two objects should be created, a form and the code for the form.
(vi) After the application is built you need to "Execute it". This is done by highlighting the form and selecting Execute.

Using Cognos Impromptu

Use the following procedure:

1. Click on Catalog, Databases, Add. Enter the name you wish to call your database. Select ODBC Gateway from the Database type pull down list and the name of your data source from the ODBC Data source pull-down list. If the data source is not there, click on Admin and add it.

   Click on the User ID check box to disable the prompt for user logon information (it is in your data source). Click OK.

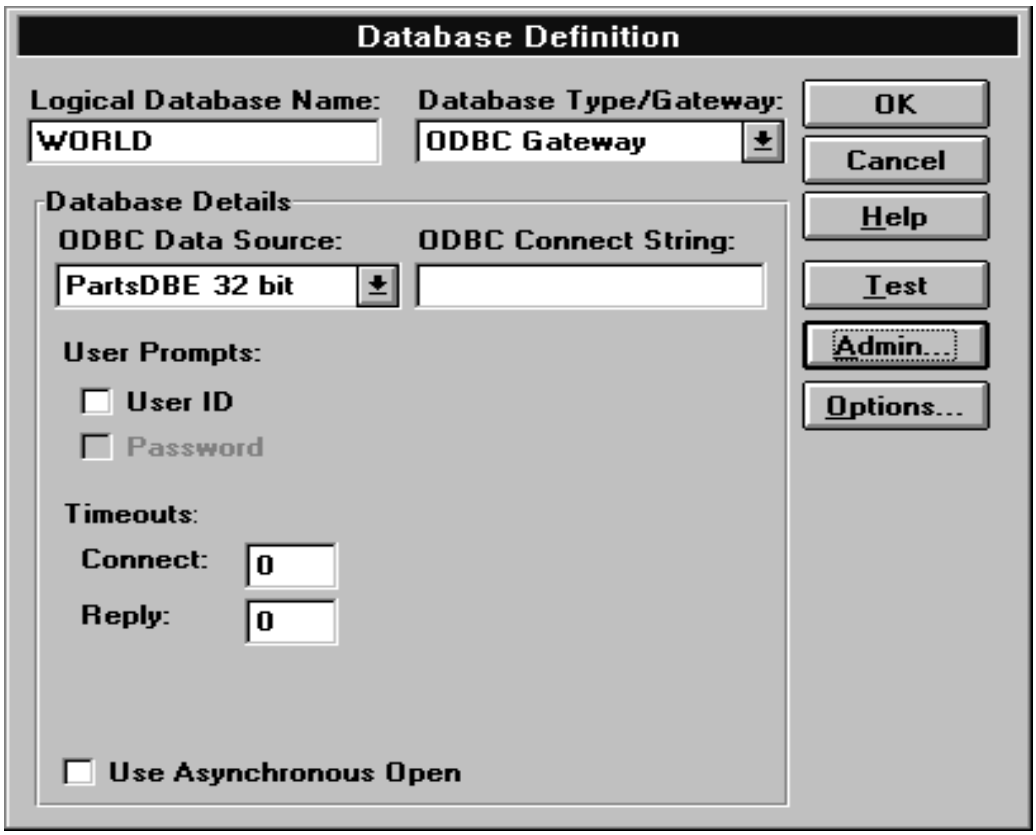

Figure 15

2. Click on Catalog, New

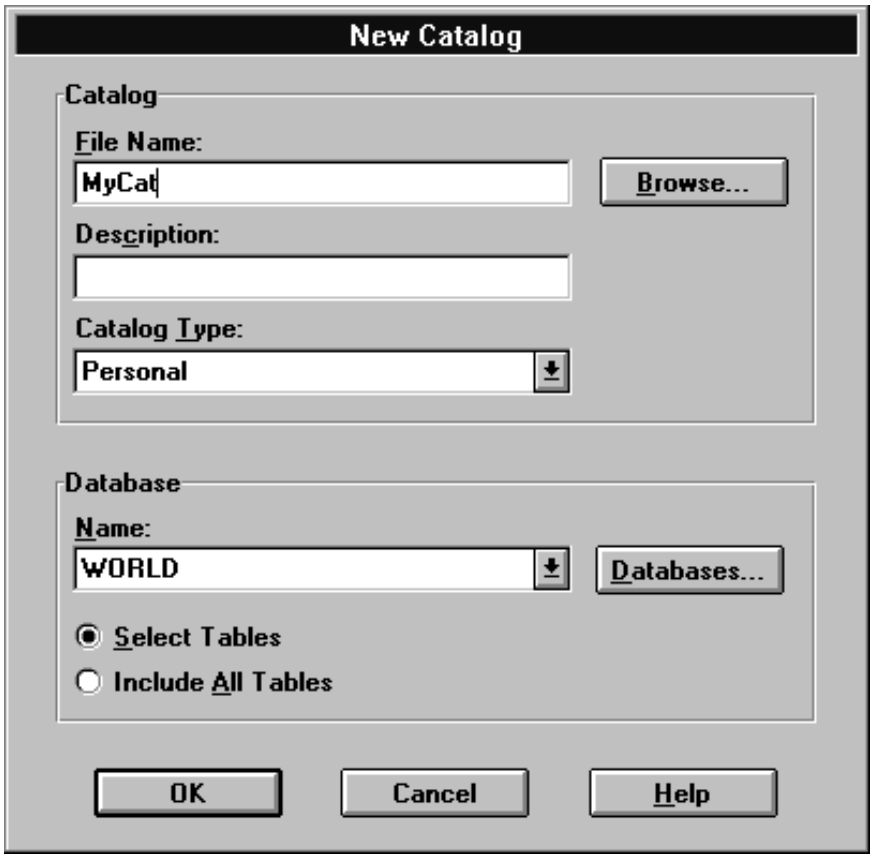

Figure 16

3. Enter a catalog name and then select the name of your database from the Database Name pull down list. Click OK. Impromptu will make the connection and display the Edit Tables Screen.

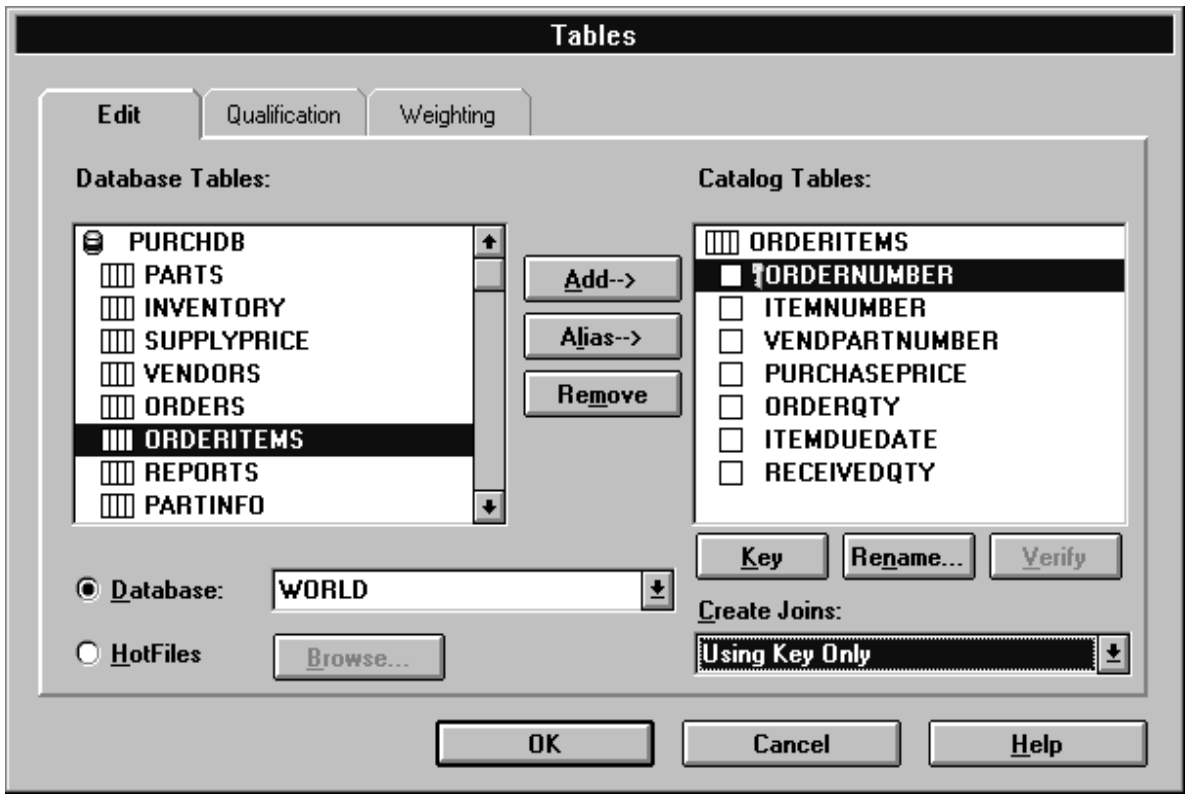

Figure 17

4. The left side of the Edit Tables screen is a list of table owners. In ALLBASE/SQL, a table owner is the owner name assigned by the database administrator (default is *User@Account*). To get a list of all the tables for a specific owner name, double click on it. To return to the list of owners, double click on the owner name shown.

5. You may now select the tables you want included in your catalog. Click OK, to load the table definition into the catalog.

6. To add tables from another database to your catalog, select Catalog, Tables, and repeat step 4. Double click on Database Name to see all entries.

7. You may now create a report through the File, New command. You will be prompted for the fields to include in your report, and Impromptu will generate the report on your screen.

Special Notes for Users of Impromptu

Viewing the SQL generated by Impromptu 3.0

Enter a new report and select Report, Query, Profile, View SQL. You can also write your own SQL and override the one generated by Impromptu. Search for help on topic “Writing your own SQL”.

Joins

ODBCLink/SE supports two kinds of joins: inner joins, sometimes called natural joins, and left outer joins. An inner join will return no records if any of the secondary records are not found; a left outer joins will still return the primary record and give nulls (or zeros) on the secondary table. To add or modify a join definition, select Catalog, Joins.

Note: To do left outer joins on the host (as opposed to Impromptu downloading both tables and doing the joins locally) you must modify file COGDMOD.INI in directory \COGNOS\COGAPPS as follows:

```
[Exception tables]
Joined=T
```

Changing the display format

Impromptu has a limited number of display formats available. To change the display format, highlight the column and select Format Data. You cannot specify edit masks, however you have a choice of numeric display formats. To add dashes or other constants to your output, you must build an expression with SUBSTRING functions. To add a decimal point to a number that is not defined as a decimal, you must create a calculated field by dividing the number by 100.

Creating Joins Using Calculated Columns

It is possible in Impromptu to create a join using a column that is calculated from two or more fields in one or more tables. To do this, select Catalog, Edit joins; select your primary table and the table you want to join in the upper left part of the screen; click “View as Expression”; and then enter an equation such as:

Table.Column3 = Table1.column1 + Table2.Column2 ....

The left hand side of this expression is the table you want to join to, and the right hand side is the expression to generate the key. An expression can consist of concatenated fields, SUBSTRING statements, constants, or all of these.

Using Jetform

In Jetform Design select Database, Select. You will be in the "Database Select Rules" screen.

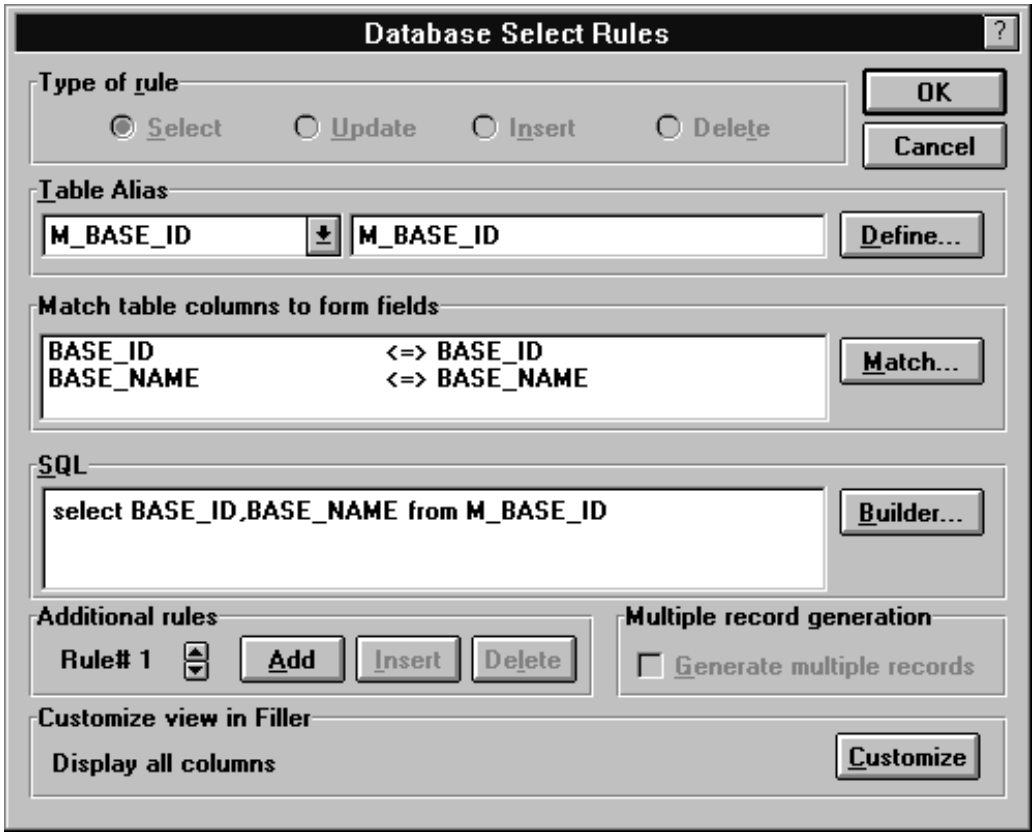

Figure 18

Click on "Define". This will put you in the "Define table alias screen".

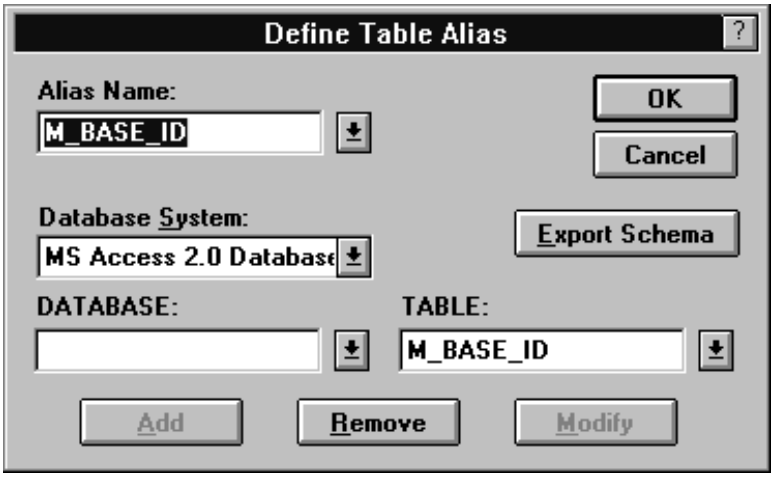

Figure 19

Under Alias Name, enter a name for your table or use the "Table Name" pull-down menu to select your table name. Then click "Add" and "Ok".

You are now back in the "Database Select Rules" screen. Click on "Match" and then match all the column names in the "Table Column" column with those in the "Form Field" column.

You should then enter Select, Update and Insert rules. Please refer to the Jetform documentation for addition information on this function.
When you are done, click on "OK". Then compile your form.

Exit Jetform design and run Jetform filler. Open your form and initiate a query by selecting File, Database, Select Transactions. This should fill-in your form. If you have entered an Update or Insert rule, you should also be able to change the data and then select File, Database, Update.

Using Lotus 123 Release 5.0

To enable Lotus 123 to use ODBC, you will have to add the following line to the file \LOTUSAPP\DATALENS\LOTUS.BCF. (Note: The following information is summarized from the file \123R5W\readme.txt):

DN="ODBC" DL="DLODBC" DD="All ODBC Sources";

Once you have done this, you may access ODBC data sources.

1. Select Tools, Database, New Query, External, ODBC.

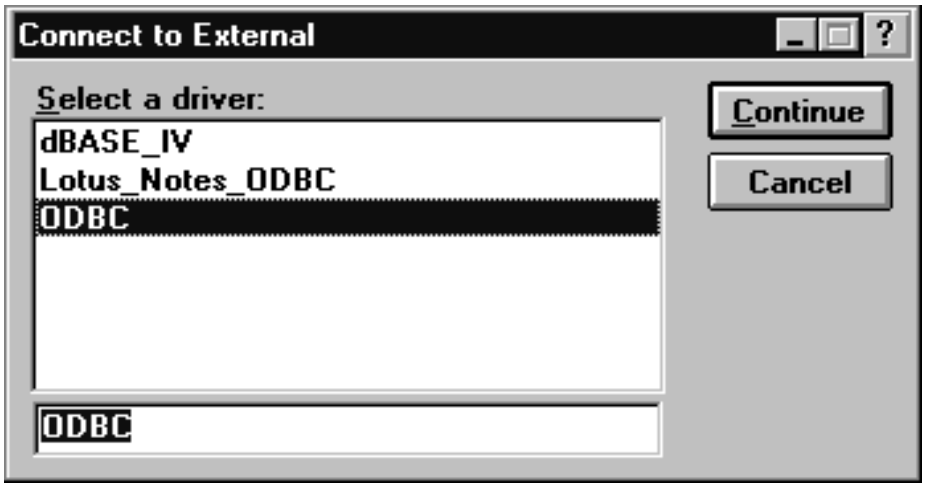

Figure 20

2. Select a data-source from the list box.

3. Select a table to query.

4. You can then choose fields to include in the query, set criteria, enter joins, etc.

Using Microsoft Access 2.0

To access data through Microsoft Access 2.0 open a new data base and select File, Attach, External SQL database.

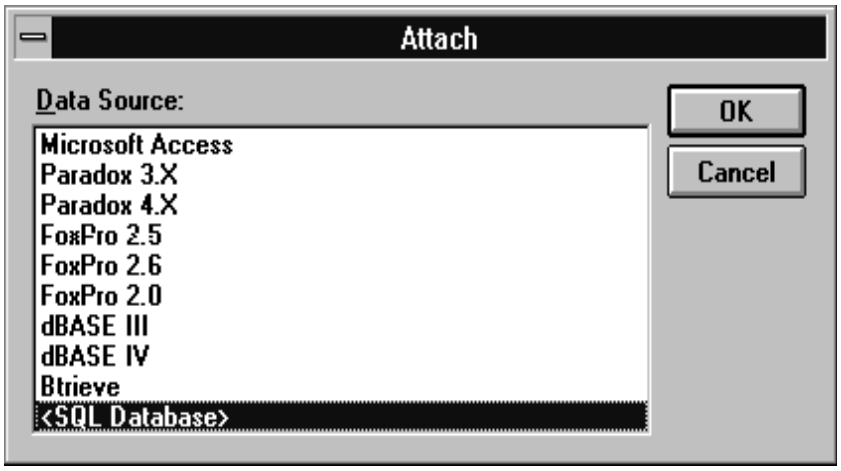

Figure 22

Then select the desired data source from the data-sources dialog box.

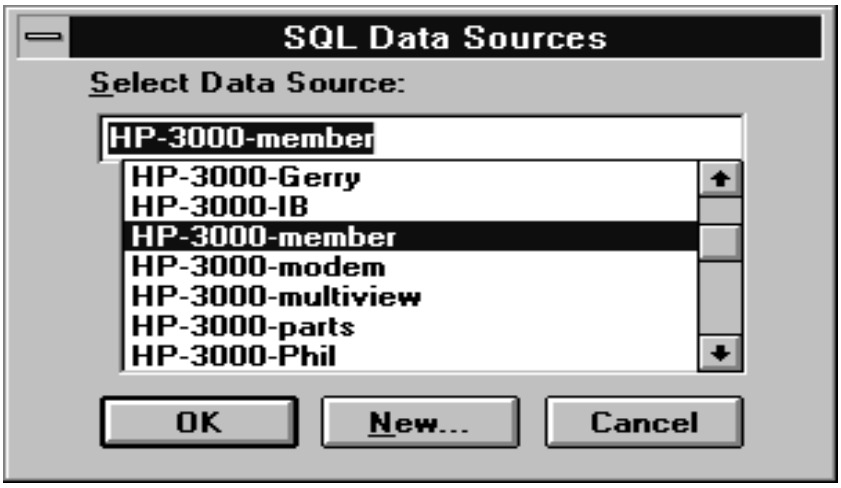

Figure 23

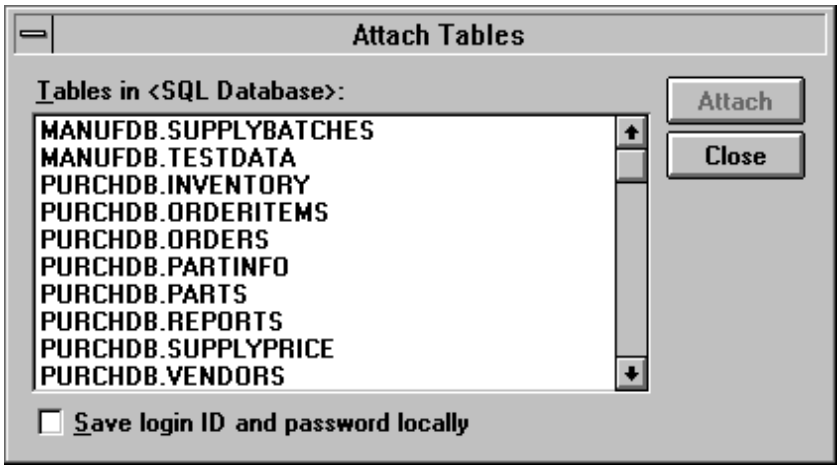

Figure 23

You will get a list of tables on the host.

The “Save login-ID and password locally” check box is not used by ODBCLink/SE.

Once the table is attached, it is treated like a local table in every way (except that you may not modify the table structure). You may open the table in table view, run reports or queries against it, or link it to other local or remote tables.

Using Microsoft Access 7.0

To access data through MS-Access 7.0, open a database and click on File, Get External Data, Link Files and select ODBC Databases from the "file of Type" pull down list.

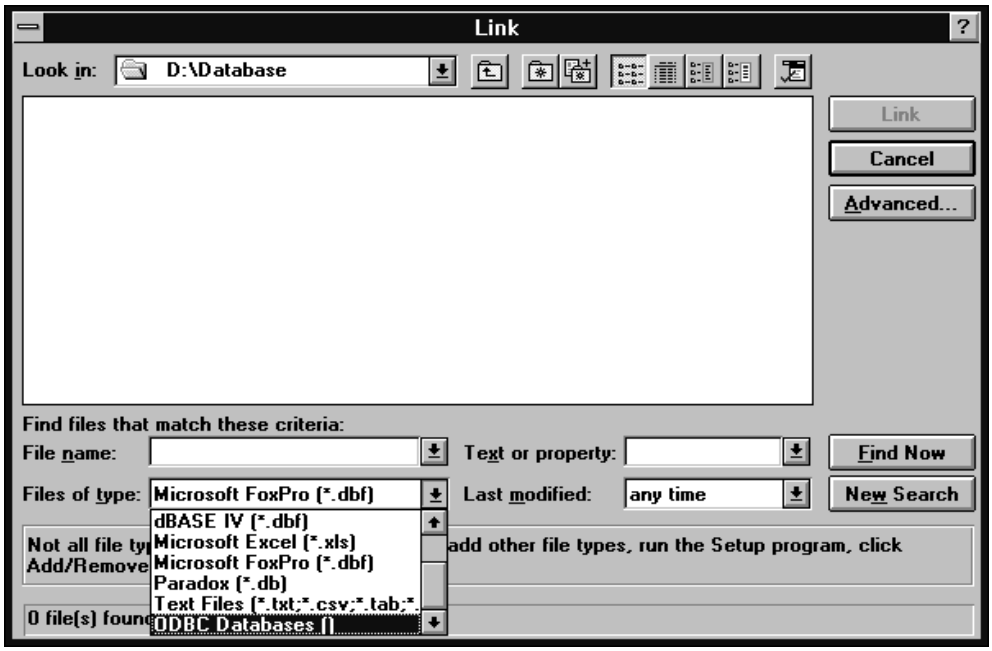

Figure 25

You now see the SQL Data Sources window.

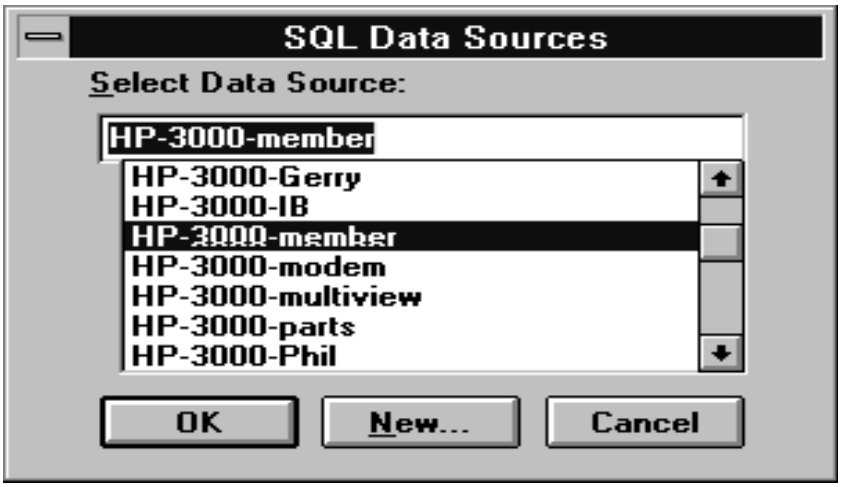

Figure 25

If the data source you require is not in the list, click on New and create a new data source. *For details see Section 4 - Configuring Data Sources.*

You must click on the name of the data source you wish to use even if there is only one.

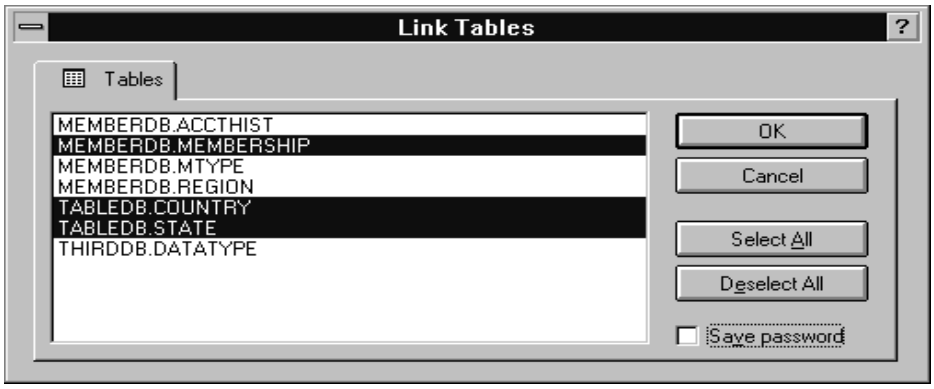

Figure 26

You may highlight one or more tables and then click “OK”. This will create an “attached table” which you may later open and modify. An attached table is treated like a local table in every way (you may read it, update it, delete it, import it, or join it to other local or remote tables.)

The “Save password” check box is not used by ODBCLink/SE.

Notes for Users of MS-Access

- For a table to be updatable in Access, it must have a unique key and Access must know about it. If the table is defined in ALLBASE/SQL with a unique key, this will be reported by ODBCLink/SE (in the SQLStatistics call that Access makes) and the table will be updatable. If Access cannot locate a primary key, it will prompt for the column or combination of columns that make up a unique key. You can ignore this prompt, and the table will not be updatable from Access; or you can specify one or more columns that make a unique key. If this combination of columns is not unique (i.e. if there are duplicate records with the same key combination) **Access will not work properly**. When you enter a unique key combination, ensure that it really is unique within the table. To see the column or columns that Access uses as the unique key, look at the table in Design view. The column will have “key” symbol in the left most column.
- Access uses a different algorithm to access a table depending on whether a unique key exists or not. If it finds a unique key, it downloads the key values and then issues an SQL statement of the form “select...From...Where key=? Or key=?...”. It then displays a screenful of data and stops. (Note, however, that it may continue to download the keys in the background so long as the table is open in “table view”.) If a unique key is not defined, Access just does a regular Select statement to read the table. It will display a screenful of data but wills till continue to read the table in the background. You may view the SQL that Access (and any ODBC application) generates by turning on logging in the ODBC Setup Screen *For details see Section 7 - Troubleshooting - Page 59*.
- * To update a record in Access, you highlight the column you want to change, make the change, and then click on any other record. Access uses an algorithm called “Optimistic concurrency control” to verify that a record has not been changed by another user. This is a less secure method than the SELECT FOR UPDATE used by other applications, however it minimizes the number of locks that are held on the database. Optimistic

concurrency control works by Access generating and Update statement of the form "Update Table Set column1=?, column2=?,...WHERE column1=? AND column2=? AND column3=? And ....". This ensures that the update will fair if the record has been changed by another user since the time it was last read in by Access.

* The default type of join used by Access is the Inner Join. When creating a Query with a join between two tables, you can change this to a Left Outer Join by double-clicking on the link between the tables. You may also use the Edit Joins screen to change the default join between specific tables, so that you don't have to edit the join every time you create a new Query.
- You cannot join a local table to a remote table efficiently in Access unless there is a one-to-one correspondence between records in the local table and records in the remote table; in other words, you must have all the columns in the local table that are necessary to form a unique key on the remote table. If this is not the case, Access will attempt to download the remote table to the local machine.

Using MS-Query

Run MS-QUERY. Select File, New Query, and select a data source (you must have pre-configured a data source with ODBC administrator). After connecting, it should come up with a list of tables on your host.

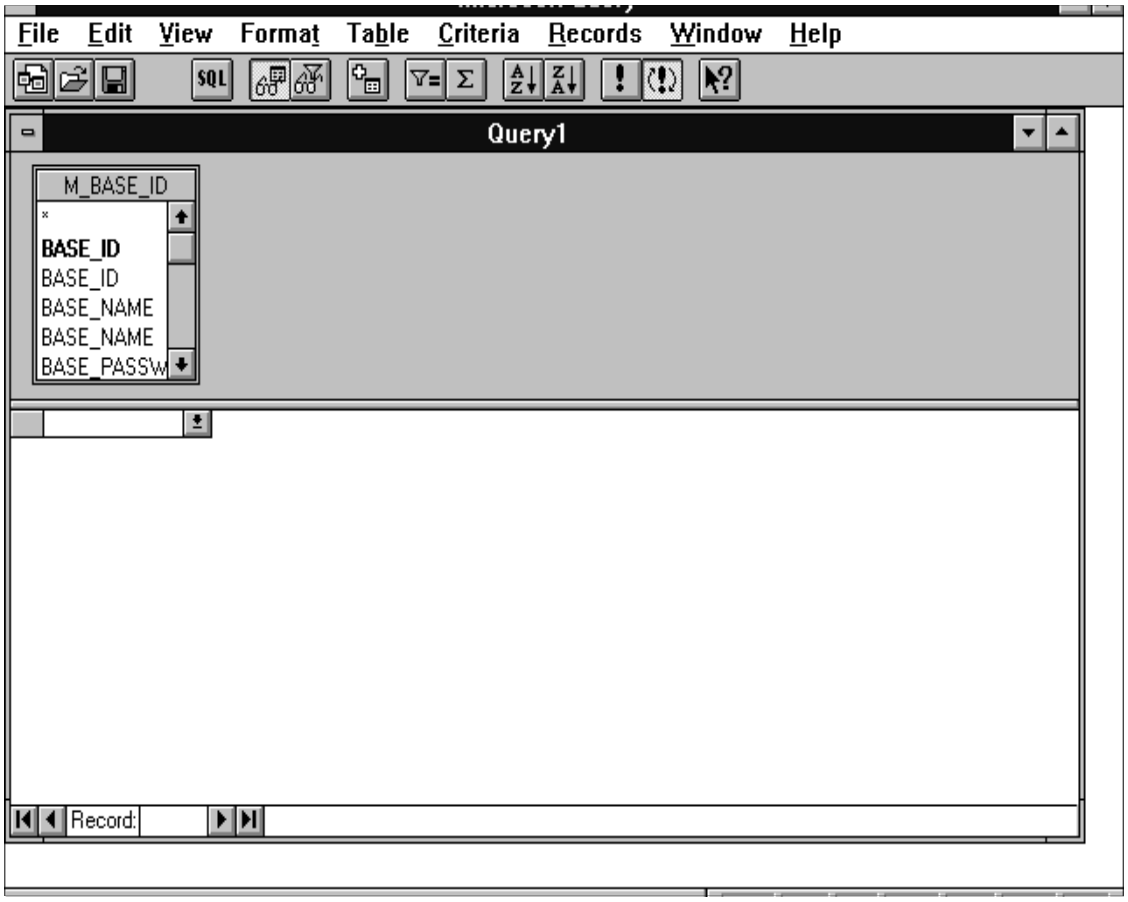

Figure 27

Select one or more tables and click OK. You will get a screen like the one in the Figure above. You may join tables by dragging the column you want to join on from one table to another in the upper part of the screen. You may view the data by double-clicking on any column name or by dragging it from the upper to the lower part of the screen.

Using Paradox

To configure Paradox to use ODBCLink/SE, run the Local Settings Utility and enter the path and name of your IDAPI Config file.

Run the IDAPI Configuration Utility, click on New Driver, enter ODBCLink/SE as the Default driver and enter the name of your data source in the Default data source field. Then select File, Save.

From Paradox, select File, New, Query, and enter the name of your database alias, as entered in the IDAPI Config Utility, in the drop-down list box "Drive (or Alias)". Refer to Figure 19 below.

The left-hand column of the screen should fill up with the names of your tables.

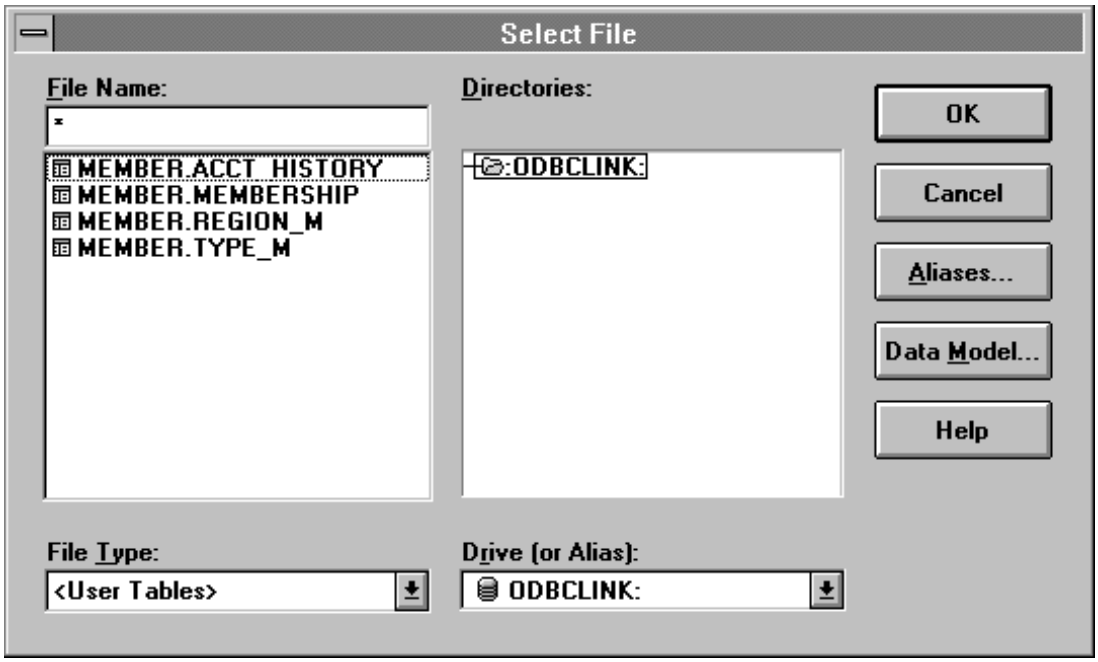

Figure 28

Then select any of the columns you want in your report. When done, click the “Run Query” button on the top of the screen.

Using PowerBuilder 3.0

To access HP3000 data from PowerBuilder, you first define a database and a database profile.

If you are working with several ODBC data sources, you may find it convenient to add the DB Profiles icon to the PowerBar. With the mouse over the PowerBar, click the right mouse button and select Customize. Then drag the DB Profiles icon from the "Selected palette" box into "Current Toolbar".

Now click on "DB Profiles" and create a new database profile.

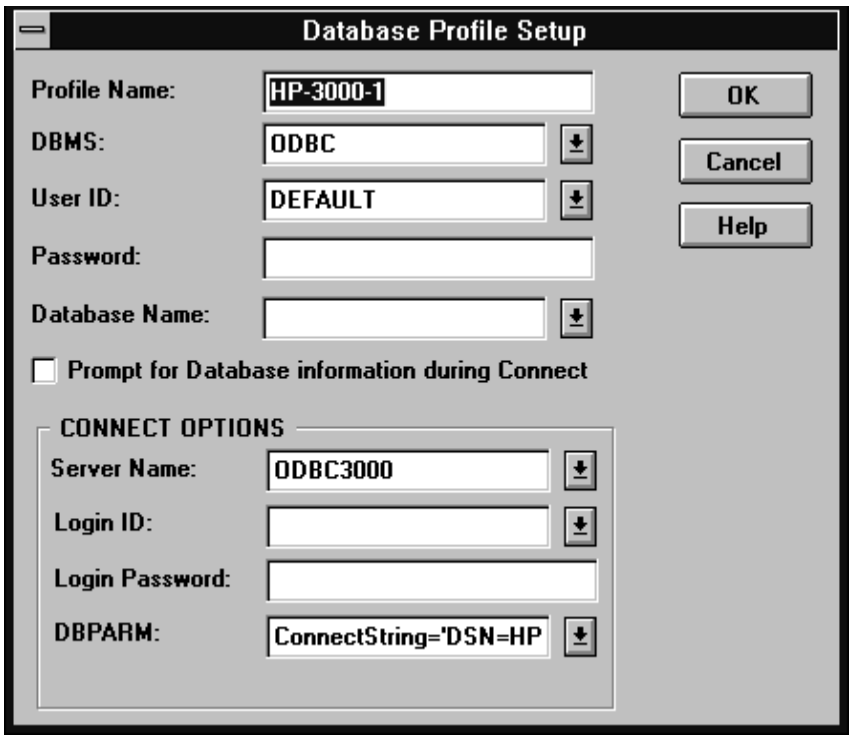

Figure 29

You must enter the following data:

Profile Name The name of your PowerBuilder database (you may have several databases with different User-ID's sharing the same data source name).

DBMS Enter ODBC.

DBParm This is your ODBC connect string. You must enter your data-source name.
"DSN<data source name>

Click OK and you should connect to the database.

Using Visual Basic (2.0 or higher)

You can use ODBCLink/SE either with VB Database controls or with Database and dynaset variables. In addition, of course, you may call ODBC functions directly from the DLL.

To use VB database controls, define a data control and assign it a Connect property of the form "DSN=HP3000" and a RecordSource property of the form "SELECT * FROM TableName". If you want to create a SnapShot (no update) set the Options property to 64 (SQL_PASSTHROUGH). You may then place text controls on the screen, set the DataSource property to the name of the data control, and set the DataField property to the name of the column in the Select statement. When the application is run, it will download all records qualified by the Select statement (or all primary key values, depending on the SQL_PASSTHROUGH option) and display the first record on the screen. You may scroll through the records by using the "Up", "Down", "Top", "Bottom" buttons on the data control. If it's a large table, it may take a long time to load (perhaps hours); you may prefer to use direct calls to ODBC.DLL like the VBDEMO application. There is no way to tell Visual Basic to stop loading a table once it has started other than by closing the form.

A second option is to use dynaset variables which gives the program more control, such as when loading a grid with data. Define a database object, assign it Connect and Options properties as above, and open it with an OpenDatabase call, as in SET DBVar=OpenDatabase(...). Then define a dynaset variable (DIM DSVar as DYNASET) and open it with a statement of the form "Set DSVar=DBVar.OpenDynaset (*SQLstatement*)." If *SQL Statement* is a Select statement, you can read the results from "DSVar.FieldName" and move to the next record with "DSVar.MoveNext". If *SQLStatement* is an Insert or Update statement, no results are returned. You can update records with the *Update* method, and you can get the selection count by calling SQLRowCount or with a query of the form "Select COUNT(*) From *Table* Where *Conditions*". To update the table, you will of course need a primary key defined and you will need to disable the SnapShot Only button.

A third option is to call ODBC functions directly. Generally you will do an SQLAllocEnv, SQLAllocConnect and SQLDriverConnect to open the database, followed by SQLExecDirect, followed by multiple SQLFetch and SQLGetData calls. If you plan to use SQLBindCol to bind fetch results to local variables you must use the following procedure:

(1) Declare strings variables as: Dim mystr as String
(2) Initialize your string variables before use: mystr=String(255,0)
(3) Use the ByVal keyword when passing the variable to the ODBC driver. Do not use the ByVal keyword in the “lpbBuf” parameter except for character data, as the goal is to pass the address of the parameter, not the value. Declare SQLBindCol as follows:

Declare Function SQLBindCol Lib “odbc.dll” (ByVal hstmt AS long, ByVal col As Integer, ByVal wConvType As Integer, lpdBuf As Any, ByVal dwbuflen As Long, lpcbout As Long) As Integer.

Using Visual C++ 1.51

Using ODBC with VC++ is beyond the scope of this manual. However, the following example, which is drawn from the book “Using ODBC 2", can be used to test.

1. Run VC++ and select Project, App Wizard.

2. Select Options, Use MFC250.DLL. (You will have to create this DLL by using the make file \MSVC\MFC\SRC\MFCDLL.MAK and then NMAKE -f MFCDLL.MAK. Then copy it to \Windows\Systems.)

3. Select Database options, Database support / No file support, then select a data source and a table.

4. Click “Create” to create an application.

5. Select Project, Build to compile your application and Project, Execute to run it. You will get a blank screen but you will be able to scroll through the record-set (although you won’t see any data).

6. Select Tools, AppStudio to enter the Windows resource editor. Select the main form and place a text control on it . Name your control IDC_CAR_NUMBER, for example, then save it and exit AppStudio.

7. Select Browse, ClassWizard, choose VCPPDemoView as the class name (or whatever you have called it). You should have a list of the controls on the form, including IDC_CAR_NUMBER. Select it and then select Add Member Variable. In the drop-down list you will see a list of your column names. Select, for instance, m_pSet>m_CarNumber.

8. You may now compile and run your application. This time you should see data on your main form as you scroll through the records.

Using Visual FoxPro

1. Select File, New, Project, enter a name, and Save.
2. Click on “Data”.
3. Select Database, New, enter a name, and Save.

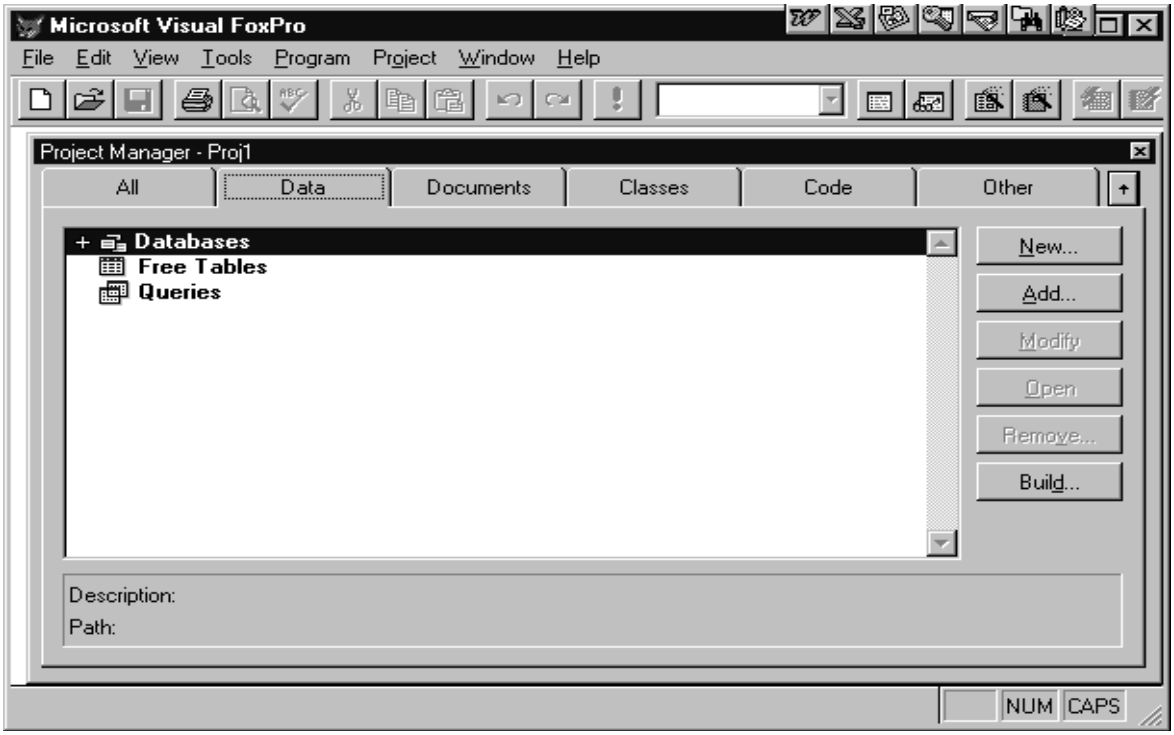

Figure 30

4. Click on “ODBC” icon in the database desktop.
5. Select “New Remote View”.
6. In “Select Connection or Data Source” screen click “New”.
7. Select a data source and click “OK”.
8. Select a connection name and click “OK”.
9. Select a table, which will get you into View Designer.
10. Select Columns.
11. Click on query button!

Section 7
Troubleshooting

Terminating an ODBC Connection

Occasionally it may be necessary to terminate an ODBC session that has deadlocked or is attempting to read a very large table (note that rebooting the PC will not be sufficient as the server has control at this point). You may do this from ISQL with the command TERMINATE USER user@account.

When an ODBC client terminates abnormally, the server process that handles the connection will normally go down after the timeout interval has expired. (*For details see - Connection Assurance Parameters - Page 7)* This is typically 30 minutes to 2 hours.

On an HP-UX system, you can kill the process manually (if you know the process-id) with the command "kill -16 *pid"* (*pid* is the process id).

On MPE/iX you must either use the ISQL TERMINATE USER command or abort the listener job using the ABORTJOB command.

Using Microsoft ODBC Test (32-bit)

You can use ODBC Test to verify correct functioning of the driver and to enter ad-hoc SQL commands. ODBCTEST and ODBCTE32 are part of the ODBC SDK that comes with the Microsoft Developers Network.

To run ODBCTE32.EXE:

1. Select Connect, Full Connect. You must enter a data source name, User-ID and a password, and click “OK”. You should get a message “Successfully connected to *databasename*”.

2. You can get a list of tables by selecting Catalog, SQL Tables, then click OK and then Results, Get Data All.

3. You can enter an SQL statement, such as “Select * from *tablename*” in the Query Window and then execute it by choosing Statement, SQLExecdirect and view the results with “Results, Get Data All”.

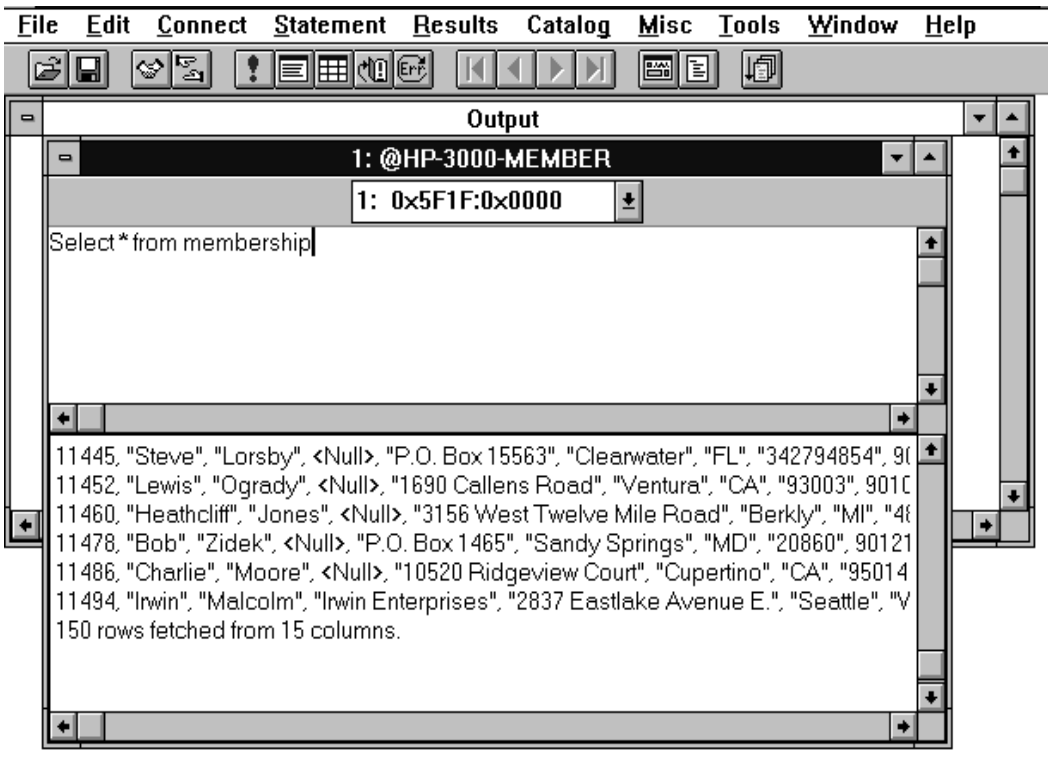

Figure 31

Verifying Software Version Numbers

Verifying the Client-Side Components

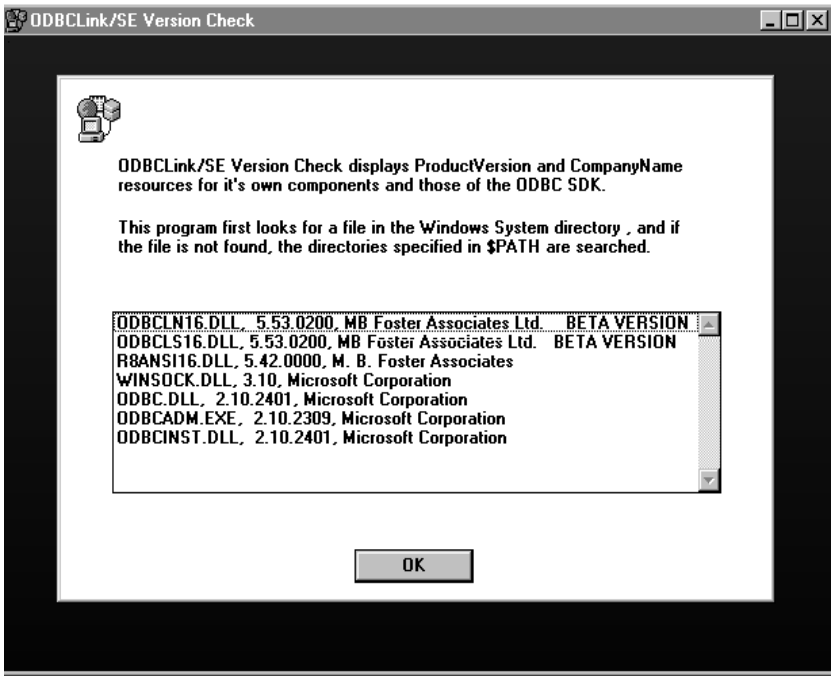

Figure 32

To determine which version of ODBCLink/SE is installed, run either WHAT16.EXE or WHAT32.EXE from the RUN window for from an MS-DOS prompt window.

NOTE: Press OK to exit before running the other program when checking both 16 and 32-bit versions.

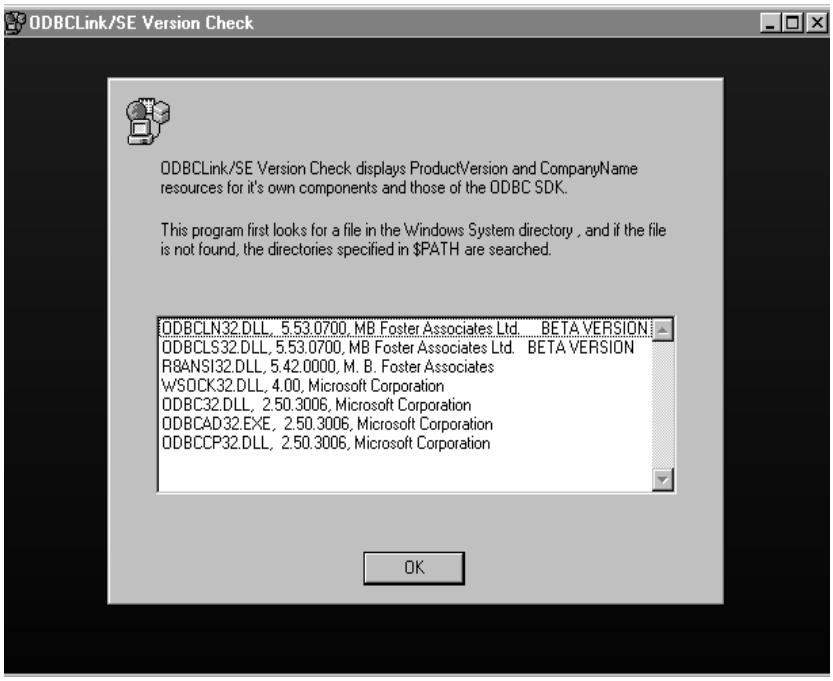

Figure 33

Deleting Unwanted Drivers

You can delete unwanted 16-bit drivers from the 16-bit ODBC Administrator on the Control Panel. Click on Drivers, highlight and unwanted driver and click on Delete.

To delete unwanted 32-bit drivers, run the driver delete tool DRVDEL32.EXE from either the RUN window or an MS-DOS prompt window.

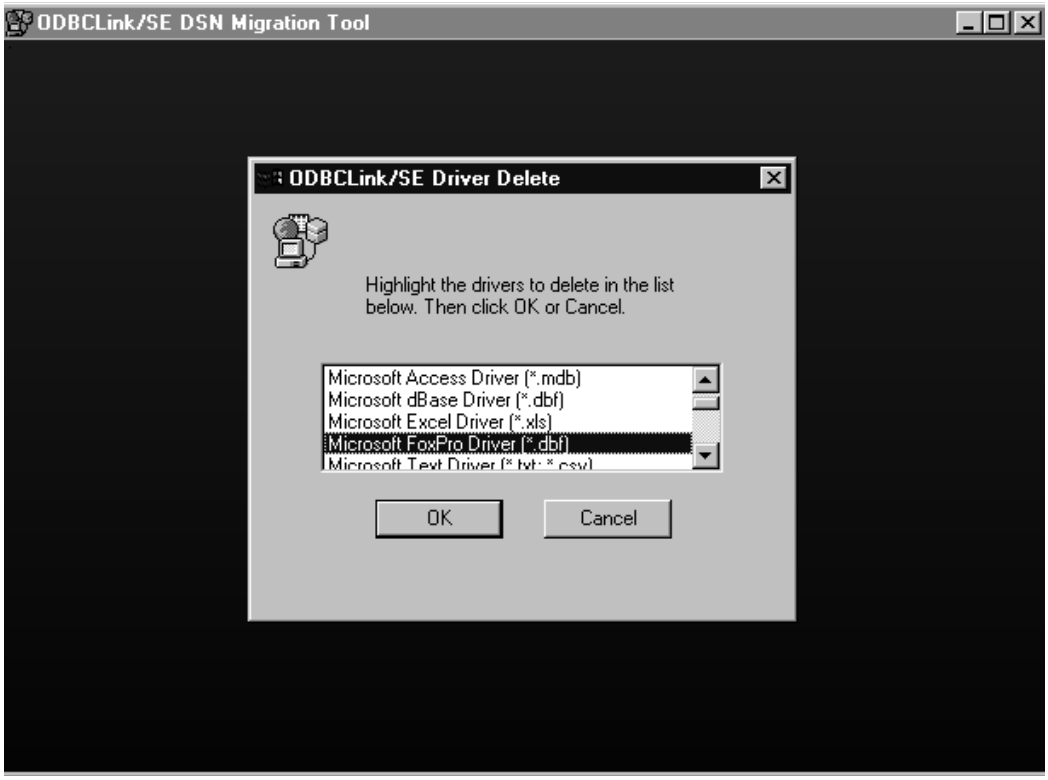

Figure 34

Highlight the drivers you wish to delete and click OK. The driver name and all related information will be removed from the system registry.

Deleting a Translation DLL

Run TRNDEL16.EXE or TRNDEL32.EXE from Program Manager by selecting FILE then RUN and entering the directory path and program name, or from File Manager by double clicking on the file name.

Figure 35

A welcome screen will be shown first. Click on OK to pop up the next screen.

Figure 36

Select the translator you want to uninstall and press OK. The translator information will then be deleted from the registry. If the program is successful it will show a window indicating the successful removal.

Figure 37

Once the translator has been deleted it will no longer be displayed in the translator list in the driver setup. To add the translator back in simply rerun the ODBC driver install program.

NOTE: The translator DLL file will still be in the windows directory after the program is run, but it will not be shown in the registries translator information.

Tools on the Database Server

Monitoring Activity with HP GlancePlus/XL (HP3000) Program.

HP GlancePlus/XL allows you to monitor a job, session, or process on MPE/iX. The GlancePlus screens display statistics on how different system resources are used. For information, refer to the HP GlancePlus/XL User's Manual.

Monitoring Activity with HP GlancePlus/UX (HP9000) Program.

HP GlancePlus/UX allows you to monitor a login or process on HP-UX. The GlancePlus screens display statistics on how different system resources are used. For information, refer to the HP GlancePlus/UX User's Manual.

Displaying Active Processes with HP-UX Process Status (ps) Command.

You can use the HP-UX ps command to display the status of your active processes. For information, refer to the HP-UX man page.

Monitoring ALLBASE/SQL Activity with SQLMON.

SQLMON is a component of ALLBASE/SQL and monitors the activity of an ALLBASE/SQL DBEnvironment. SQLMON summarizes the activity for the entire DBEnvironment, or focuses on individual sessions, programs, or database components. For information, refer to the ALLBASE/SQL Performance and Monitoring Guidelines.

Checking the Listener Log File on the HP3000

Look for ODBCLOG where the listener is running.

Checking the Listener Log File on the HP9000

Look for odbclog in the /tmp/ directory.

Using the Host Testing Utility ODBCUTSE

If you are having problems connecting or accessing a table on the host, you can access it directly on the host in the same way the ODBC server accesses it, by running the ODBCUTSE utility.

To run the program on MPE/iX, enter:

ODBCUTSE.ODBCSE.SYS “dbe name”

To run the program on HP/UX, enter:

/usr/bin/odbcse/odbcutse “dbe name” (for HP-UX V9.x systems)
or
/opt/allbase/bin/odbcutse “dbe name” (for HP-UX V10.x systems)

You should now be connected to the database. To see a menu of available commands, enter HELP:

```
0>HELP

  SHOW        Display tables and table structures
  EXIT        Exit program
  SELECT      SQL Select
  FETCH       SQL Fetch [ALL|##]
  INSERT      SQL Insert
  UPDATE      SQL Update
  DELETE      SQL Delete
  SQL         Execute immediate SQL command
  COMMIT      SQL Commit
  PARAM       Parameter substitution on a prepared statement
  STMT        Change current statement/cursor number
  ROLLBACK    SQL Rollback
  EXECUTE     Execute stored procedure
  QUIT Exit program
  FOREIGN     Display foreign Keys
  TABPRIV     Display table privileges
  0>
```

Enter the command you wish to use. For example, enter SHOW to see a list of available tables:

```
>show
Dataset/Table          Database               Type
MANUFDB.SUPPLYBATCHES  partsdbe.sql.mbftest   ALLBASE TABLE
MANUFDB.TESTDATA       partsdbe.sql.mbftest   ALLBASE TABLE
PURCHDB.PARTS          partsdbe.sql.mbftest   ALLBASE TABLE
PURCHDB.INVENTORY      partsdbe.sql.mbftest   ALLBASE TABLE
PURCHDB.SUPPLYPRICE    partsdbe.sql.mbftest   ALLBASE TABLE
PURCHDB.VENDORS        partsdbe.sql.mbftest   ALLBASE TABLE
PURCHDB.ORDERS         partsdbe.sql.mbftest   ALLBASE TABLE
PURCHDB.ORDERITEMS     partsdbe.sql.mbftest   ALLBASE TABLE
PURCHDB.PARTINFO       partsdbe.sql.mbftest   ALLBASE VIEW
PURCHDB.VENDORSTATS    partsdbe.sql.mbftest   ALLBASE VIEW
RECDB.CLUBS            partsdbe.sql.mbftest   ALLBASE TABLE
RECDB.MEMBERS          partsdbe.sql.mbftest   ALLBASE TABLE
RECDB.EVENTS           partsdbe.sql.mbftest   ALLBASE TABLE
```

To see the list of columns and indexes for a table, enter SHOW followed by the name of the table. For example:

```
>sho recdb.clubs
Field       ODBC-Datatype  HP-Datatype   Length  Offset  Nulls
CLUBNAME    CHAR(15)                     15      0
CLUBPHONE   SMALLINT                     2       16      1
ACTIVITY    CHAR(18)                     18      20      1

Indexed field       Index Type              Index Name
CLUBNAME            CONSTRAINT PRIMARY      CLUBS_PK
```

Host Logging

Normally, only connections and errors are logged to the ODBCLOG file on the host. If you check the box labelled “Trace SQL calls on the server” in the data-source setup dialogue, all the SQL sent from the host and the ALLBASE operation will be logged as well.

If a listener error occurs, an error message is logged in the ODBCLOG file

Here is the format of the ODBCLOG file (this file can be found in group and account of the listener job, typically ODBCLOG.ODBCSE.SYS on MPE systems and /tmp on Unix systems)

```
96/11/20 14:12:09 068ODBC listener started
96/11/20 14:13:13 068 Listener created new connection as SYS1 (PIN 77), IP=192.9.3.102
96/11/20 14:13:13 077 ODBCLNSE E.52.00 started from sockets
96/11/20 14:13:22 077 Allbase Prepare[49]SELECT NAME, OWNER, TYPE, NPAGES, NROWS FROM CATALOG.TABLE;
96/11/20 14:13:26 077 Allbase SetParams[49]
96/11/20 14:13:27 077 Allbase Fetch[49] Rows=119 MaxRows=200
96/11/20 14:13:27 077 Allbase Fetch[49] Rows=0 MaxRows=200
96/11/20 14:13:27 077 Allbase Close cursor[49]
96/11/20 14:13:27 077 Allbase Commit
96/11/20 14:13:27 077 Allbase Prepare[49]SELECT NAME,OWNER,NUMP,NUMR,MULTIRESULT FROM CATALOG.PROCEDURE;
96/11/20 14:13:27 077 Allbase SetParams[49]
96/11/20 14:13:28 077 Allbase Fetch[49] Rows=6 MaxRows=200
96/11/20 14:13:28 077 Allbase Fetch[49] Rows=0 MaxRows=200
96/11/20 14:13:28 077 Allbase Close cursor[49]
96/11/20 14:13:28 077 Allbase Commit
96/11/20 14:13:28 077 Connected to partsdbe, ClientVersion=E.52.00
96/11/20 14:13:35 077 Prepare[0]SELECT _TABLES
96/11/20 14:13:35 077 SetParams[0] nparam=0
96/11/20 14:13:53 077 FreeStmt[0]
96/11/20 14:13:54 077 Normal exit, Received 3 requests (133 bytes), Sent 0 records (5141 bytes), CPU=2 (5.6%)
```

The first three columns of the list are the date, time, and pin (process id number) of the requesting connection. The last column is the Allbase operation being performed, with the statement or cursor number in square brackets. The number is square brackets (as in Allbase prepare[49]) is the statement number. For a discussion on multiple statements, *refer to Section 5 - Application Development - Page 29*

Tools on the PC Client

The preferred method to trace ODBC calls on the client is by setting the “Trace ODBC Calls” check box in ODBC setup. However, Syware’s Dr. DeeBee Spy is installed with ODBCLink/SE (in the Windows directory). It can be used to trace calls to any ODBC driver.

ODBC Call Tracing using ODBCLink.LOG

To log all the ODBC calls made by the client, check the box “Trace ODBC calls on the client” on the data-source setup screen. This will create a file, called ODBCLINK.LOG, of all the ODBC calls made by the client. This file normally resides in the directory where the ODBC application was started from. Some applications however change their working directory, for MS-Access for instance look for the log file in “My Documents” directory.

Here is the format of the ODBCLINK.LOG file:

```
SQLDriverConnect(0) hstmt=0 Ver= 5.52.0000 szConnStrIn=<DSN=DBA MEMBER; UID=Admin; PWD=>,Completed_string=<DSN=DBA MEMBER; UID=#mpeix/192.9.3.10:MembrDBE.SQL.mbftest,,an#DBA,dba/.MBFTEST/password,work/> Login=<>

SQLError(100) hstmt=0 pfNativeError=0,szErrorMsg=
SQLGetInfo(0) hstmt=0 fInfoType=23, rgbInfoValue=  [1,0,0,0]
SQLSetConnectOption(0) hstmt=0 fOption=101 vParam=1
SQLAllocStmt(0) hstmt=0
SQLGetStmtOption(0) hstmt=0 fOption=0 vParam=0
SQLSetStmtOption(-1) hstmt=0 fOption=0 vParam=60
SQLError(0) hstmt=0 pfNativeError=-57,szErrorMsg=[ODBCLN32.DLL] Driver not capable
SQLError(100) hstmt=0 pfNativeError=0,szErrorMsg=

SQLExecDirect(0) hstmt=0 SELECT MEMBERDB.ACCTHIST.NUMBER,MEMBERDB.ACCTHIST.TRANTYPE,MEMBERDB.ACCTHIST.TRANDATE FROM MEMBERDB.ACCTHIST

SQLFetch(0) hstmt=0
SQLGetData(0) hstmt=0 icol=1 fCType=99 pcbValue=4 data= '[25,39,0,0]
SQLGetData(0) hstmt=0 icol=2 fCType=99 pcbValue=3 data=INV[73,78,86,0]
SQLGetData(0) hstmt=0 icol=3 fCType=99 pcbValue=6 data=È  [-56,7,1,0]
```

Each line starts with the name of the function being called, with the return code in brackets, followed by the different parameters of the call. Refer to the ODBC SDK manual for a description of the parameters of the call.

ODBC Call Tracing using Dr. DeeBee Spy

Dr. DeeBee Spy traces calls to the ODBC driver. Dr. DeeBee Spy logs each function call along with the input and output values. To turn on logging, do the following:

1. From the Program Manager Menu in Windows, click on RUN.

2. Type DRDBSP.

3. Choose OK.

4. Select the Data Source Name to be traced.

5. Choose OK.

The Dr. DeeBee Spy icon will appear at the bottom of the screen. It will flash when it is logging.

To turn off logging:

1. Click on the Dr. DeeBee Spy icon.

2. Choose CLOSE.

The log is located in C:\WINDOWS\DRDEEBEE.LOG.

"Dr. DeeBee Spy is © 1995 Syware, Inc., All rights reserved"

Modifying the System Registry and ODBC.INI Files

You only need to do this if you want to modify an option that you cannot configure in the ODBC control panel. An example of this is the MAXSTMT options. *For details see - Appendix A - Implementation Notes - Page 73.*

To add or change an option, either edit the ODBC.INI file (for 16-bit data-sources) or run the registry editor (regedt32.exe). In the registry editor, look under HKEY_CURRENT_USERS / Software / Microsoft / ODBC.INI.

Error Messages

Errors in ODBCLink/SE can come from several sources; all messages have a prefix which indicates the source of the error, as follows:

[ODBCLN16]	Generated by the 16-bit client DLL.
[ODBCLN32]	Generated by the 32-bit client DLL.
[WINSOCK]	Comes from the Winsock driver.
[ALLBASE]	Generated by the ALLBASE/SQL command interpreter.

The first action you should take in case of a problem is to attempt to execute the command manually, e.g. try to login using a terminal emulator, or else try issuing the SQL statement that causes a problem in ODBCUTSE or ISQL. If you do not know the command causing the error (because you are using Microsoft Access for instance), turn on host logging and examine your ODBCLOG file.

Errors from ALLBASE/SQL or IMAGE/SQL are followed by DBERR or DBWARN and a number. These error messages are documented in the ALLBASE/SQL Message Manual or in the IMAGE/SQL Administration Guide.

Error messages are returned by ODBCLink/SE to the application. Each message is followed by a CAUSE which gives information as to why the warning or error occurred. This is followed by an ACTION which gives information on how to resolve the problem.

For all other errors, refer to the appropriate documentation or manuals.

THIS PAGE INTENTIONALLY LEFT BLANK

Appendix A
Implementation Notes

Following is a list of features that are unique or specific to ODBCLink/SE and some details about them.

Primary key name returned by SQLStatistics

Most ODBC applications including Microsoft Access, Visual Basic and PowerBuilder require a unique primary key for a table to be updatable.

If a unique key has not be defined in ALLBASE/SQL the table will not be updatable.

Reading or writing to ALLBASE/SQL LongVarBinary items

Any file residing on the HP can be inserted into an ALLBASE/SQL LongVarBinary field with the SQL command:

```
INSERT INTO Table VALUES (..., '<BLOB  >%$', ...);
```

Where BLOB is the name of the file on the HP and the '>%$' tells ALLBASE/SQL to copy the contents to a memory location when retrieving the row. You can also have ALLBASE/SQL copy it to a file during retrieval ('<BLOB >OUTFILE'). In either case, the data can be retrieved in the standard way (SQLFetch followed by SQLGetData in chunks).

You may also create a binary column on the HP by using ODBC functions call sequence: SQLPrepare, SQLBindparameter (DATA_AT_EXECUTION), SQLExecute, SQLParamData, SQLPutData. *Refer to ODBC SDK manual for details.*

Maximum number of statements

ODBCLink/SE supports up to 50 concurrent statements, or cursors, per connection. However, SQLGetInfo with option SQL_MAX_STMT will only report 1 as the maximum number of concurrent statements. This is because multiple statements (or cursors) in ALLBASE/SQL on the same connection are not truly independent. Having SQLGetInfo return 1 for SQL_MAX_STMT forces MS-Access, and other applications to use multiple connections (multiple SQLDriverConnect's) instead of multiple statements on the same connection (one SQLDriverConnect and multiple SQLAllocStmt's).

You **can** use multiple statements from within the same connection (in fact, some applications ignore the SQLGetInfo and do this anyway). There is a performance advantage to doing things this way since a new statement (or cursor) in ALLBASE/SQL takes much less overhead than a new connection. However you should be aware of the following:

- In ALLBASE/SQL, a COMMIT or ROLLBACK operation closes all open cursors within

the connection. If you have multiple Select statements going on different statements, the next time you SQLFetch on one statement, after a Commit on another statement, you will get an error from ALLBASE/SQL “ALLBASE has closed the cursor for this statement”. You may not get this error immediately, as ODBCLink/SE caches a certain number of rows during a fetch.

- If you are in AUTOCOMMIT mode ON (this is the default option, set through SQLSetConnectOptions and in ODBC Administrator), there is an implicit Commit done whenever you fetch the last row. This is so that locks held on the database by the open cursor are released. This may destroy other open cursors. Note also that opening a table for the first time automatically triggers various initialization calls that cause a commit to be done when AUTOCOMMIT is ON.
- If you are using prepared statements with substitution parameters to insert or update you need not be concerned about this, as the prepared statement will still be valid after a Commit, and you may bind new parameters and SQLExecute again with no problem. Prepared statements are SQL statements with question marks ‘?’ that you SQLPrepare and then call SQLBindParam or SQLBindCol. *Refer to ODBC SDK for details.*

If you wish to use multiple statements, but are not writing your own SQL (i.e. you are using a 4GL application) you can tell ODBCLink/SE to have SQLGetInfo report more than 1 statement per connection. To do this, modify the ODBC.INI or the registry manually, *(For details see - Modifying the System Registry and ODBC.INI Files - Page 70)* and add a new parameter “MAXSTMT=50".

ALLBASE/SQL and IMAGE/SQL Restrictions on the ODBC Grammar

The following table summarizes the ALLBASE/SQL restrictions on the ODBC grammar:

Statement	Programming Considerations
CREATE TABLE	UNIQUE PRIMARY KEY must follow NOT NULL
DROP TABLE	ALLBASE/SQL does not provide CASCADE or RESTRICT
REVOKE	ALLBASE/SQL does not provide RESTRICT.

ALLBASE/SQL provides a DATETIME data type that is similar to TIMESTAMP.

ALLBASE/SQL does not implement optimistic locking. Optimistic locking means that the DBMS does not lock data until just before an update is made, thus improving concurrency because locks are not held for long.

Using the ANSI Character Set

The HP3000 and HP9000 both use the default ROMAN8 character set. Most applications running under Microsoft Windows use the 8-bit ANSI (ISO 8859/1) character set to provide support for Western European languages (including American). The first 127 characters are the same for both ROMAN8 and ANSI character sets, so only the extended and special characters are different.

If your PC client application uses these extended or special characters, then the data from the database server must be converted from ROMAN8 to ANSI, and the data returning to the database server must be converted from ANSI to ROMAN8. To activate the conversion, select the “Roman8 to PC-ANSI translation DLL during the data-source setup. The conversion takes place on the PC client.

Bind variables of datatype CHAR and VARCHAR are also converted. A bind variable in an SQL statement is associated (bound) to variables defined in a program. A bind variable is another name for a dynamic parameter.

For more information, refer to the ALLBASE/SQL Reference Manual.

A few characters cannot be converted to ROMAN8 and then back to ANSI. For example, when the copyright sign in ANSI is converted to ROMAN8, it is changed to a lowercase "c." When the data is returned to the PC client, the copyright sign cannot be recognized, as it was converted to a "c." The following characters, sent from the PC client, cannot be properly converted from ANSI to ROMAN8 in a round-trip fashion. (The hex designation is listed along with the character name.)

- A6 broken bar
- A9 copyright sign
- AC not sign
- AE registered trade mark
- B2 superscript 2
- B3 superscript 3
- B8 cedilla
- B9 superscript 1
- D7 multiply sign
- F7 divide sign

Similarly, the following characters, sent from the database server, cannot be properly converted from ROMAN8 to ANSI in a round-trip fashion.

- A9 accent grave
- AA circumflex accent
- AC tilde accent
- BE Dutch guilder
- EB uppercase S caron

- EC lowercase s caron
- EE uppercase Y umlaut
- F6 long dash
- FC solid box

Unsupported ALLBASE/SQL and IMAGE/SQL Statements

There are several reasons why some ALLBASE/SQL and IMAGE/SQL statements are not supported in the ODBCLink/SE environment. They are explained here:

- Several ALLBASE/SQL statements work only with embedded SQL; ODBCLink/SE does not support embedded SQL.
- Some ALLBASE/SQL statements relate to functions, such as connection, that have been replaced by functions resident on the PC client.
- Some ALLBASE/SQL and IMAGE/SQL functions are best restricted to the DBA; statements which control these functions are accessible only through Interactive Structured Query Language (ISQL) on the server.
- ODBCLink/SE internally prepares an SQL statement to be dynamically preprocessed. Therefore, you cannot use some statements such as EXECUTE IMMEDIATE. *Refer to the "PREPARE" section of the "SQL Statements" chapter in the ALLBASE/SQL Reference Manual for a list of statements that cannot be prepared.*

ALLBASE/SQL and IMAGE/SQL Statements That Work Only with Embedded SQL

PC client application software that accesses ALLBASE/SQL and IMAGE/SQL databases through ODBCLink/SE do not support embedded SQL. The application software calls the database functions directly; there is no need for preprocessing.

Because the following ALLBASE/SQL and IMAGE/SQL statements function only with embedded SQL, they are not supported by ODBCLink/SE:

CLOSE CURSOR
BEGIN DECLARE SECTION
DECLARE CURSOR
DESCRIBE
END DECLARE SECTION
EXECUTE
EXECUTE IMMEDIATE
FETCH
INCLUDE
OPEN
PREPARE
REFETCH
SQLEXPLAIN
WHENEVER

ALLBASE/SQL and IMAGE/SQL Statements Replaced by Functions on the PC Client

Some of the functionality has been moved from the database server to the PC client with ODBCLink/SE.

The following ALLBASE/SQL and IMAGE/SQL statements are not supported because they have been replaced by other functions resident on the PC client:

CONNECT
DISCONNECT
RELEASE
RESET
SET CONNECT
SET MULTITRANSACTION
START DBE
START DBE NEW
START DBE NEWLOG
STOP DBE

If you attempt to use any of these unsupported statements, you will receive an error message.

THIS PAGE INTENTIONALLY LEFT BLANK

Appendix B
Creating a DBEnvironment

On the HP3000 Database Server

To enable a connection between a PC client and database server, you must set up an ALLBASE/SQL DBEnvironment. If you do not have an existing database for testing purposes, you can use the ALLBASE/SQL SQLSetup tool to create a DBEnvironment called PartsDBE. If you are using IMAGE/SQL, you can use the IMSQL.SAMPLEDB.SYS command to create an IMAGE/SQL database called MusicDBE. *For more information, refer to Getting Started with HP IMAGE/SQL.*

Follow these steps to create the PartsDBE database on the HP3000:

1. Logon to the group and account where you want to create the DBEnvironment.
2. From the command prompt, enter the following command:

   :SQLSETUP.SAMPLEDB.SYS

A menu like the following will appear on your screen:

```
        Options for Setting Up ALLBASE/SQL Sample DBEnvironments

==============================================================

    Choose one:

    1. Create PartsDBE without sample programs
    2. Create PartsDBE, copy, preprocess and compile sample programs
    3. Copy, preprocess and compile sample programs only
    4. Generate a schema for PartsDBE
    5. Display schema for PartsDBE
    6. Purge PartsDBE and sample programs
    7. Help
    0. Exit

==============================================================

    Enter your choice=>
```

Figure 38

3. Choose option 1 to create PartsDBE. This option creates the DBEnvironment, defines all of

its tables, views, indexes, and security structure, and then loads it with data. As the system creates PartsDBE, you see several messages displayed. At the end of the creation process, you see the following message:

Creation and Loading of PartsDBE is now complete!
Press Return to continue...

4. Choose 0 to exit the menu.

If you have designed your own database for testing purposes, you can consult your database administrator and complete the following tasks:

1. Plan the security structure of the DBEnvironment. Decide which users will be granted access to the DBEnvironment, databases within the DBEnvironment, and tables within the databases.
2. Create the account where the DBEnvironment is to reside.
3. Grant authorizations to users who will be using the test DBEnvironment.
4. Create the DBEnvironment, including databases and tables.

After you setup the DBEnvironment, ensure that you can connect to it. For example,

```
: isql
isql=> CONNECT TO 'PartsDBE.SomeGrp.SomeAcct';
isql=> SELECT * FROM SYSTEM.TABLE;
:
At this point you should see the system table information.
:
U[p],d[own],l[eft],r[ight],t[op],b[ottom],pr[int]<n>,or e[nd]

    > e;

     isql=> exit;
```

For a complete explanation of the procedures necessary to create the PartsDBE DBEnvironment, refer to the appendix "Sample DBEnvironment" in the ALLBASE/SQL Reference Manual.

On the HP9000 Database Server

To enable a connection between a PC client and database server, you must set up an ALLBASE/SQL DBEnvironment. If you do not have an existing database for testing purposes, you can use the ALLBASE/SQL SQLSetup tool to create a DBEnvironment called PartsDBE.

Follow these steps to create the PartsDBE database on the HP9000:

1. Login to the directory where you want to create the DBEnvironment.
2. From the C shell, enter the following command:

   $ /usr/lib/allbase/hpsql/sqlsetup Return

   From the Korn shell or Bourne shell, enter the following command:

   $ csh /usr/lib/allbase/hpsql/sqlsetup Return

   A menu like the following will appear on your screen:

```
        Options for Setting Up ALLBASE/SQL Sample DBEnvironments

=======================================
Choose one:

    1. Create PartsDBE without sample programs
    2. Create PartsDBE, copy, preprocess and compile sample programs
    3. Copy, preprocess and compile sample programs only
    4. Generate a schema for PartsDBE
    5. Display schema for PartsDBE
    6. Purge PartsDBE and sample programs
    7. Help
    0. Exit
=======================================

Enter your choice=>
```

Figure 39

3. Choose option 1 to create PartsDBE. This option creates the DBEnvironment, defines all of its tables, views, indexes, and security structure, and then loads it with data. As the system creates PartsDBE, you see several messages displayed. At the end of the creation process, you see the following message:

   Creation and Loading of PartsDBE is now complete!

Press Return to continue...

4. Choose 0 to exit the menu.

If you have designed your own database for testing purposes, you can consult your database administrator and complete the following tasks:

1. Plan the security structure of the DBEnvironment. Decide which users will be granted access to the DBEnvironment, databases within the DBEnvironment, and tables within the databases.
2. Create the account where the DBEnvironment is to reside.
3. Grant authorizations to users who will be using the test DBEnvironment.
4. Create the DBEnvironment, including databases and tables.

After you setup the DBEnvironment, ensure that you can connect to it. For example,

```
$ isql
isql=> CONNECT TO '/users/hpsql/sampledb/PartsDBE';
isql=> SELECT * FROM SYSTEM.TABLE;
:
At this point you should see the system table information.
:
U[p],d[own],l[eft],r[ight],t[op],b[ottom],pr[int]<n>,or e[nd]

> e;

isql=> exit;
```

For a complete explanation of the procedures necessary to create the PartsDBE DBEnvironment, refer to the appendix "Sample DBEnvironment" in the ALLBASE/SQL Reference Manual.

Appendix C
List of Installed Files

16 Bit Driver Client - For Windows 3.1 or Windows 3.11

NOTE: "..\" is a short-hand that represents the location of the windows installation (e.g. "c:\windows\")

Install Disc	Windows 3.x	Description
_bootstp.exe	not installed to hard drive	
_bootstp.obj	not installed to hard drive	
mssetup.ex	not installed to hard drive	
b5ccdc16.dl_	..\system\b5ccdc16.dll	Used for character translation for Chinese characters
cpn16ut.dl_	..\system\cpn16ut.dll	16 bit installer Universal Thunking DLL. Used by 32-bit applications with 16-bit installer DLL
ctl3dv2.dl_	..\system\ctl3dv2.dll	The DLL for 3D controls
drvsetup.exe	not installed to hard drive	16-bit ODBC driver setup program
dsnmig.ex_	...\system\dsnmig.exe	Migrates 16bit PCAPI clients to ODBCLink/SE client
drdbsp16.ex_	...\system\drdbsp16.exe	16-bit Dr. DeeBee spy used for debugging
drdbsp16.dl_	...\system\drdbsp16.dll	16-bit Dr. DeeBee spy used for debugging
odbc.dl_	...\system\odbc.dll	16-bit driver manager (Microsoft)
odbc.inf	not installed to hard drive	Setup information file
odbc16ut.dl_	[win]\system\odbc16ut.dll	16-bit ODBC Universal Thunking DLL. Used by 32-bit applications with 16-bit driver manager
odbc32.dl_	..\system\odbc32.dll	32-bit Thunking Driver manager
odbcadm.ex_	..\system\odbcadm.exe	ODBC Administrator application
odbccp32.dl_	..\system\odbccp32.dll	Windows control Panel ODBC Installer Tool
odbccurs.dl_	..\system\odbccurs.dll	Microsoft cursor library DLL. Can be used by client to implement scrollable cursors
odbcinst.dl_	..\system\odbcinst.dll	ODBC administrator DLL. Called by ODBC Control Panel Application
odbcinst.hl_	..\system\odbcinst.hlp	Help file for odbcinst.dll
odbcln16.dl_	..\system\odbcln16.dll	16-bit version of the ODBCLink/SE driver
odbcls16.dl_	..\system\odbcls16.dll	16-bit version of ODBCLink/SE setup
r8ansi16.dl_	..\system\r8ansi16.dll	16-bit character translation DLL for translation between the ANSI and roman-8 character set.
setup.exe	not installed to hard drive	Microsoft ODBC Setup program
setup.lst	not installed to hard drive	
trndel16.exe	...\system\trndel16.exe	Used to delete/uninstall the 16-bit translation DLL
what16.ex_	...\system\what16.exe	Utility to determine what versions of drivers required by ODBC application are currently loaded on the PC

32 Bit Driver Client - For Windows 95

NOTE: "..\" is a short-hand that represents the location of the windows installation (e.g. c"\windows\")

Install Disc	Windows 95	Description
_bootstp.exe	not installed to hard drive	
mssetup.ex	not installed to hard drive	
ctl3d95.dl_	..\system\ctl3d95.dll	DLL for 3D controls in Windows95
b5ccdc32.dl_	..\system\b5ccdc32.dll	Used for character translation for Chinese characters
drdbsp32.ex_	..\system\drdbsp32.exe	32-bit Dr. DeeBee spy used for debugging
drdbsp32.dl_	..\system\drdbsp32.dll	32-bit Dr. DeeBee spy used for debugging
drvstp32.exe	not installed to hard drive	32-bit ODBC driver setup program
dvrdel32.ex_	..\system\dvrdel32.exe	Utility used to delete a 32-bit driver from the registry
ds16gt.dl_	..\system\ds16gt.dll	16-bit Driver Setup Generic Thunking DLL. 16-bit applications use this in conjunctions with the 32-bit Installer DLL
ds32gt.dl_	..\system\ds32gt.dll	32-bit Driver Setup Generic Thunking DLL.
mfc30.dl_	..\system\mfc30.dll	Microsoft Foundation Class Library
msvcrt20.dl_	..\system\msvcrt20.dll	Microsoft Run-Time Library
odbc.inf	not installed to hard drive	Installation control file
odbc16gt.dl_	..\system\odbc16gt.dll	16-bit ODBC Generic Thunking DLL
odbc32.dl_	..\system\odbc32.dll	32-bit ODBC Driver Manager
odbc32gt.dl_	..\system\odbc32gt.dll	32-bit ODBC Generic Thunking DLL
odbcad32.ex_	..\system\odbcad32.exe	ODBC Administrator Application
odbccp32.cp_	..\system\odbccp32.cpl	32-bit ODBC Installer Control Panel Tool
odbccp32.dl_	..\system\odbccp32.dll	32-bit ODBC Control Panel Installer DLL
odbccr32.dl_	..\system\odbccr32.dll	32-bit cursor library
odbcinst.hl_	..\system\odbcinst.hlp	Installation help file
odbcln32.dl_	..\system\odbcln32.dll	32-bit ODBCLink/SE Driver
odbcls32.dll	..\system\odbcls32.dll	32-bit ODBCLink/SE Driver Setup DLL
r8ansi32.dl_	..\system\r8ansi32.dll	32-bit character translation DLL for translations between the ANSI and roman-8 character sets.
setup.exe	not installed to hard drive	Microsoft ODBC Setup program drive
setup.lst	not installed to hard drive	
trndel32.exe	...\system\trndel32.exe	Translator uninstall utility to "de-register" translator information
what32.ex_	...\system\what32.exe	Utility to determine what versions of drivers required by ODBC application are currently loaded on the PC

32 Bit Driver Client - For Windows NT

NOTE: "..\" is a short-hand that represents the location of the windows installation (e.g. c"\windows\")

Install Disc	Windows NT	Description
_bootstp.exe	not installed to hard drive	
mssetup.ex	not installed to hard drive	
b5ccdc32.dl_	..\system32\b5ccdc32.dll	Used for character translation for Chinese characters
ctl3dnt.dl_	..\system32\ctl3dnt.dll	DLL for 3D controls in WindowsNT
drdbsp32.ex_	..\system32\drdbsp32.exe	32-bit Dr. DeeBee spy used for debugging
drdbsp32.dl_	..\system32\drdbsp32.dll	32-bit Dr. DeeBee spy used for debugging
drvstp32.exe	not installed to hard drive	32-bit ODBC driver setup program
dvrdel32.ex_	..\system32\dvrdel32.exe	Utility used to delete a 32-bit driver from the registry
ds16gt.dl_	..\system32\ds16gt.dll	16-bit Driver Setup Generic Thunking DLL. 16-bit applications use this in conjunctions with the 32-bit Installer DLL
ds32gt.dl_	..\system32\ds32gt.dll	32-bit Driver Setup Generic Thunking DLL.
mfc30.dl_	..\system32\mfc30.dll	Microsoft Foundation Class Library
msvcrt20.dl_	..\system32\msvcrt20.dll	Microsoft Run-Time Library
odbc.inf	not installed to hard drive	Installation control file
odbc16gt.dl_	..\system32\odbc16gt.dll	16-bit ODBC Generic Thunking DLL
odbc32.dl_	..\system32\odbc32.dll	32-bit ODBC Driver Manager
odbc32gt.dl_	..\system32\odbc32gt.dll	32-bit ODBC Generic Thunking DLL
odbcad32.ex_	..\system32\odbcad32.exe	ODBC Administrator Application
odbccp32.cp_	..\system32\odbccp32.cpl	32-bit ODBC Installer Control Panel Tool
odbccp32.dl_	..\system32\odbccp32.dll	32-bit ODBC Control Panel Installer DLL
odbccr32.dl_	..\system32\odbccr32.dll	32-bit cursor library
odbcinst.hl_	..\system32\odbcinst.hlp	Installation help file
odbcln32.dl_	..\system32\odbcln32.dll	32-bit ODBCLink/SE Driver
odbcls32.dll	..\system32\odbcls32.dll	32-bit ODBCLink/SE Driver Setup DLL
r8ansi32.dl_	..\system32\r8ansi32.dll	32-bit character translation DLL for translation between the ANSI and roman-8 character set.
setup.exe	not installed to hard drive	Microsoft ODBC Setup program drive
setup.lst	not installed to hard drive	
trndel32.exe	...\system32\trndel32.exe	Translator uninstall utility to "de-register" translator information
what32.ex_	...\system32\what32.exe	Utility to determine what versions of drivers required by ODBC application are currently loaded on the PC

MPE/iX Host

ODBCJOB.ODBCSE.SYS	sample jobstream to run the listener process
ODBCLNSE.ODBCSE.SYS	the listener program
ODBCXLSE.ODBCSE.SYS	an XL library containing common code
ODBCUTSE.ODBCSE.SYS	utility for technical support purposes
ODBCCL16.ODBCSE.SYS	a self-extracting archive for the 16bit client software
ODBCCL32.ODBCSE.SYS	a self-extracting archive for the 32bit client software
ODBCLOG.ODBCSE.SYS	log file (when created).

HP-UX 9.x Host

/usr/bin/odbcse/odbclnse	the listener program
/usr/bin/odbcse/odbcutse	utility for technical support purposes
/usr/bin/odbcse/odbccl16	a self-extracting archive of the 16bit client software
/usr/bin/odbcse/odbccl32	a self-extracting archive of the 32bit client software

HP-UX 10.x Host

/opt/allbase/bin/odbcse/odbclnse	the listener program
/opt/allbase/bin/odbcse/odbcutse	utility for technical support purposes
/opt/allbase/bin/odbcse/odbccl16	a self-extracting archive of the 16bit client software
/opt/allbase/bin/odbcse/odbccl32	a self-extracting archive of the 32bit client software
/tmp/odbclog	log file (when created).

GLOSSARY

BLOB An acronym for Binary Large OBject. This is a logical datatype that is usually associated with large, uniquely formatted data considered to be binary in nature. An example of this might be a bit mapped graphic image.

Column In relational database terms, this refers to a data item within a table.

Data Source This term is used to describe a unique instance of the configuration of an ODBC driver. For example, you could configure a data source as "HP3000-SERIAL" using the ODBCLink driver as well as a data source called "HP3000-LAN" using the Hewlett-Packard supplied ODBCLink/SE driver.

Entry In relational database terms, this refers to a collection of related columns.

File A collection of data within the "File System" of an operating system such as MPE.

ISQL This is an interactive program, supplied with ALLBASE/SQL, that allows the user to construct ad-hoc enquiries against the database.

Record A collection of related data items.

Row In relational database terms, this refers to an entry within a table. *See Entry, Record.*

SQL An acronym for Structured Query Language. This is an emerging industry standard way of constructing commands to a relational database. It is often used synonymously with "Relational Database" although this is not strictly correct.

Table A collection of commonly structured data entries. This term is usually used when referring to Relational databases. *See File, Set.*

Winsock Winsock is a Microsoft standard based on Berkeley sockets. It is a definition of a protocol that can be used to communicate between co-operating processes.

THIS PAGE INTENTIONALLY LEFT BLANK

INDEX