SQL Quick Syntax Guide

Version 7.2
April 1996
Part No. 000-7880A

Published by INFORMIX® Press

Informix Software, Inc.
4100 Bohannon Drive
Menlo Park, CA 94025

The following are worldwide trademarks of Informix Software, Inc., or its subsidiaries, registered in the United States of America as indicated by an "®," and in numerous other countries worldwide:

INFORMIX®; INFORMIX®-OnLine Dynamic Server™; C-ISAM®

The following are worldwide trademarks of the indicated owners or their subsidiaries, registered in the United States of America as indicated by an "®," and in numerous other countries worldwide:

X/Open Company Ltd.: UNIX®; X/Open®
Adobe Systems Incorporated: PostScript®

Some of the products or services mentioned in this document are provided by companies other than Informix. These products or services are identified by the trademark or servicemark of the appropriate companies. If you have a question about one of these products or services, please contact the company in question directly.

Documentation Team: Diana Chase, Geeta Karmarkar, Tom Noronha

RESTRICTED RIGHTS LEGEND

Software and accompanying materials acquired with United States Federal Government funds or intended for use within or for any United States federal agency are provided with "Restricted Rights" as defined in DFARS 252.227-7013(c)(1)(ii) or FAR 52.227-19.

Introduction

The following statements and segments for Version 7.2 are presented in this guide:

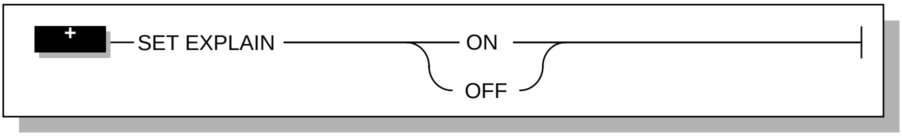
- SQL statements
- SQL segments
- Stored Procedure Language (SPL) statements
- INFORMIX-OnLine / Optical statements

This guide presents the syntax diagrams for statements and segments but does not provide any explanatory information. For complete descriptions of SQL and SPL statements and segments, see the *Informix Guide to SQL: Syntax*. For complete descriptions of INFORMIX-OnLine / Optical statements, see the *INFORMIX-OnLine/Optical User Manual*. These manuals provide the following information for each statement:

- A brief introduction that explains the purpose of the statement
- A syntax diagram that is identical to the syntax diagram in the *SQL Quick Syntax Guide*
- A syntax table that provides the purpose, restrictions, and syntax for all italicized elements in a syntax diagram
- Rules and examples of usage
- References to related statements

Syntax Conventions

This section describes conventions for syntax diagrams. Each diagram displays the sequences of required and optional keywords, terms, and symbols that are valid in a given statement, command line, or other specification, as in the following diagram of the SET EXPLAIN statement.



Each syntax diagram begins at the upper left corner and ends at the upper right corner with a vertical terminator. Between these points, any path that does not stop or reverse direction describes a possible form of the statement. (For a few diagrams, notes in the text identify path segments that are mutually exclusive.)

Syntax elements in a path represent terms, keywords, symbols, and segments that can appear in your statement. Except for separators in loops, which the path approaches counterclockwise from the right, the path always approaches elements from the left and continues to the right. Unless otherwise noted, at least one blank character separates syntax elements.

Elements That Can Appear on the Path

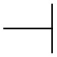
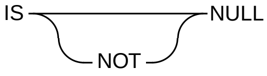
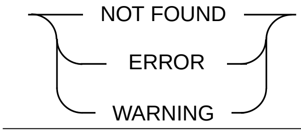
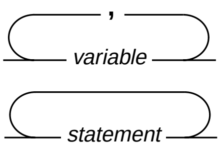
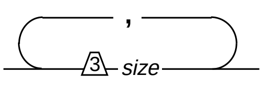
You might encounter one or more of the following elements on a path.

Element	Description
KEYWORD	A word in UPPERCASE characters is a keyword. You must spell the word exactly as shown; however, you can use either uppercase or lowercase characters.
(. , ; @ + * - /)	Punctuation and other non-alphanumeric characters are literal symbols that you must enter exactly as shown.
' '	Single quotes are literal symbols that you must enter as shown.
variable	A word in *italics* represents a value that you must supply. A table immediately following the diagram explains the value.
ADD Clause p. 1-14 ADD Clause	A term shown in a rectangle represents a subdiagram on the same page (if no page number is supplied) or a specified page, as if the subdiagram were spliced into the diagram at this point. The same subdiagram can be represented by rectangles of different shapes, as in these symbols for the ADD Clause subdiagram.
CREATE OPTICAL CLUSTER Statement *see* INFORMIX-OnLine/Optical User Manual	This reference represents a statement described in the *INFORMIX-OnLine/Optical User Manual*. Imagine that the statement is spliced into the diagram at this point.
OL	An icon is a warning that this path is valid only for some products, or only under certain conditions. Characters on the icons indicate what products or conditions support the path. These icons might appear in a syntax diagram: OL This path is valid only for INFORMIX-OnLine Dynamic Server. SE This path is valid only for INFORMIX-SE.

(1 of 3)

Element	Description
	D/B This path is valid only for DB-Access. **ESQL** This path is valid only for SQL statements in INFORMIX-ESQL / C and INFORMIX-ESQL / COBOL. **E/C** This path is valid only for INFORMIX-ESQL / C. **E/CO** This path is valid only for INFORMIX-ESQL / COBOL. **SPL** This path is valid only for only if you are using Informix Stored Procedure Language (SPL). **OP** This path is valid only for INFORMIX-OnLine / Optical. **+** This path is an Informix extension to ANSI SQL-92 entry-level standard SQL. If you initiate Informix extension checking and include this syntax branch, you receive a warning. If you have set the **DBANSIWARN** environment variable at compile time, or have used the **-ansi** compile flag, you receive warnings at compile time. If you have **DBANSIWARN** set at runtime, or if you compiled with the **-ansi** flag, warning flags are set in the **sqlwarn** structure. **GLS** This path is valid only only if your database or application is GLS compliant.
— ALL -	A shaded option is the default. If you do not specify any of the available options, this option is in effect by default.
——►——►	Syntax that is enclosed between a pair of arrows is a subdiagram.

(2 of 3)

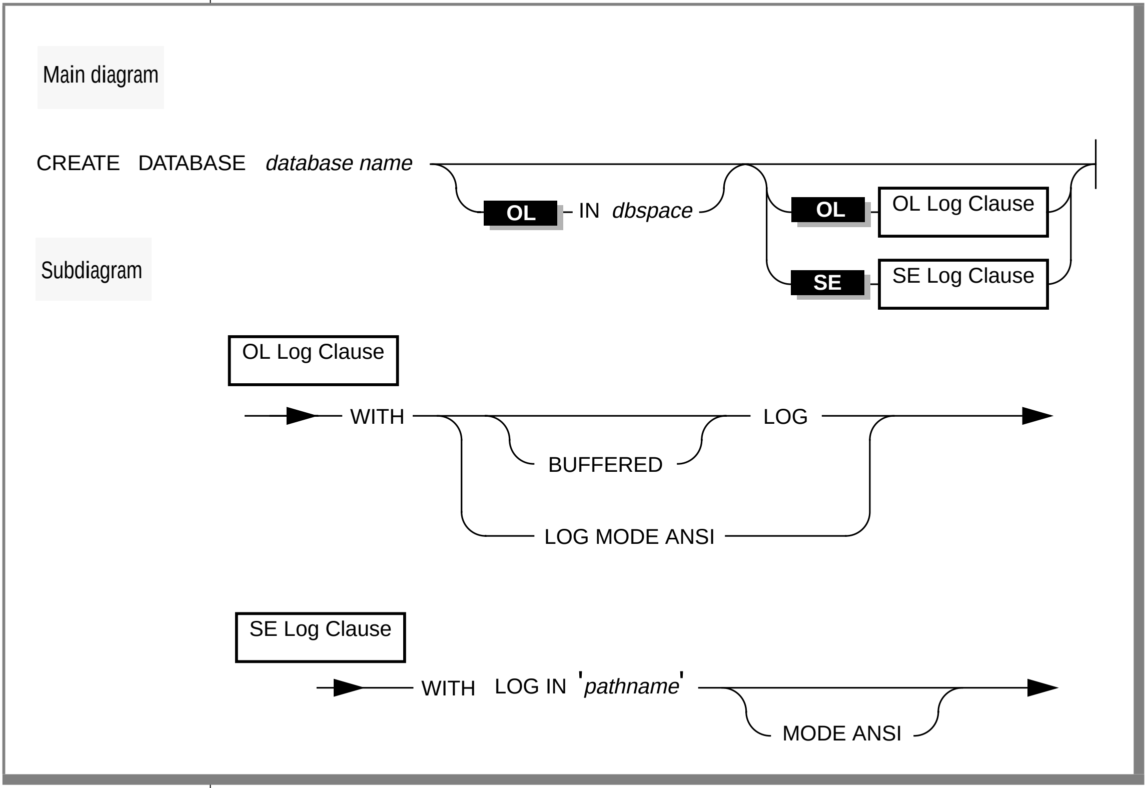
Element	Description
	The vertical line is a terminator. This symbol only appears at the right, indicating that the syntax diagram is complete.
	A branch below the main path indicates an optional path. (Any term on the main path is required, unless a branch can circumvent it.)
	A set of multiple branches indicates that a choice among more than two different paths is available.
	A loop indicates a path that you can repeat. Punctuation along the top of the loop indicates the separator symbol for list items, as in this example. If no symbol appears, a blank space is the separator,
	A gate (3) on a path indicates that you can only use that path the indicated number of times, even if it is part of a larger loop. Here you can specify *size* no more than three times within this statement segment.

(3 of 3)

How to Read a Syntax Diagram

Figure 1 shows a syntax diagram that uses most of the path elements that are listed in the previous table.

Figure 1
Elements of a Syntax Diagram



To use this diagram to construct a statement, start at the top left with the keywords CREATE DATABASE. Then follow the diagram to the right, proceeding through the options that you want.

To read the example syntax diagram

1. You must type the words CREATE DATABASE.
2. You must supply a *database name*.
3. You can stop, taking the direct route to the terminator, or you can take one or more of the optional paths.
4. If desired, you can designate a dbspace by typing the word IN and a dbspace name.
5. If desired, you can specify logging. Here, you are constrained by the database server with which you are working:
   - ❑ If you are using INFORMIX-OnLine Dynamic Server, go to the subdiagram named *OL Log Clause*. Follow the subdiagram by typing the keyword WITH, then choosing and typing either LOG, BUFFERED LOG, or LOG MODE ANSI. Then follow the arrow back to the main diagram.
   - ❑ If you are using INFORMIX-SE, go to the subdiagram named *SE Log Clause*. Follow the subdiagram by typing the keywords WITH LOG IN, typing a quote, supplying a pathname, and closing the quotes. You can then choose the MODE ANSI option below the line or continue to follow the line across.
6. Once you are back at the main diagram, you come to the terminator. Your CREATE DATABASE statement is complete.

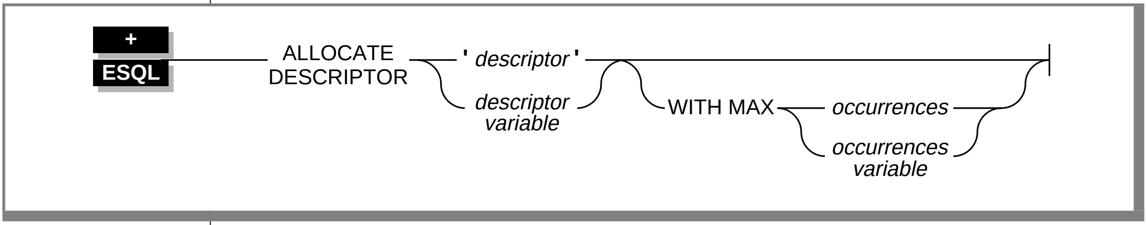
Sample-Code Conventions

Examples of SQL code occur throughout this manual. Except where noted, the code is not specific to any single Informix application development tool. If only SQL statements are listed in the example, they are not delimited by semicolons. To use this SQL code for a specific product, you must apply the syntax rules for that product. For example, if you are using the Query-language option of DB-Access, you must delimit multiple statements with semicolons. If you are using an SQL API, you must use EXEC SQL and a semicolon (or other appropriate delimiters) at the start and end of each statement, respectively.

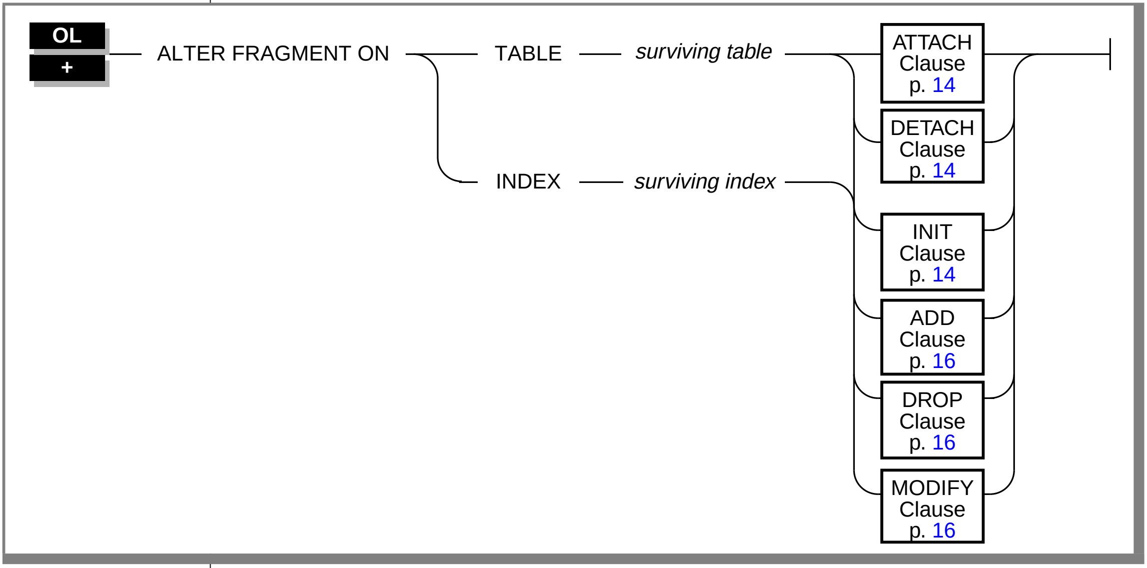
SQL Statements

Section

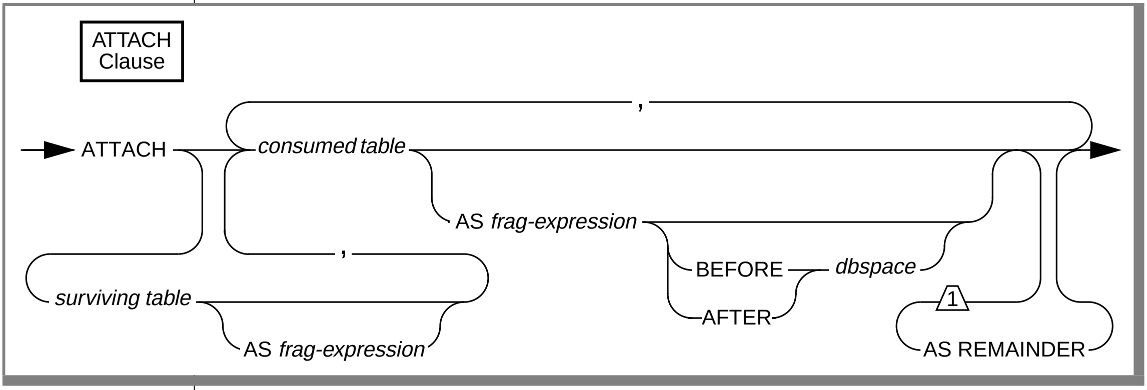
ALLOCATE DESCRIPTOR



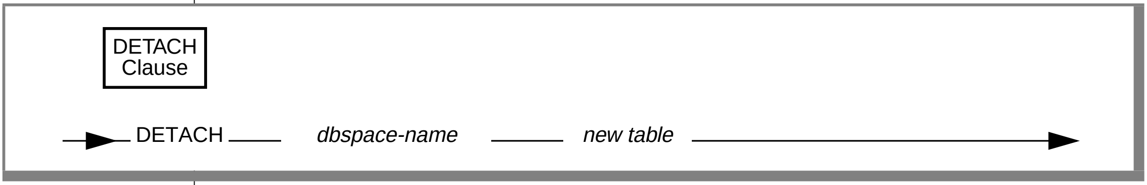
ALTER FRAGMENT



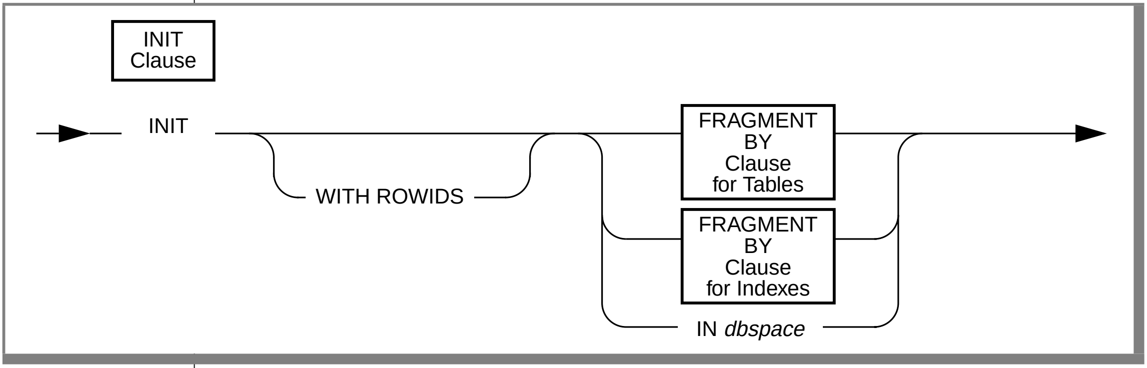
ATTACH Clause



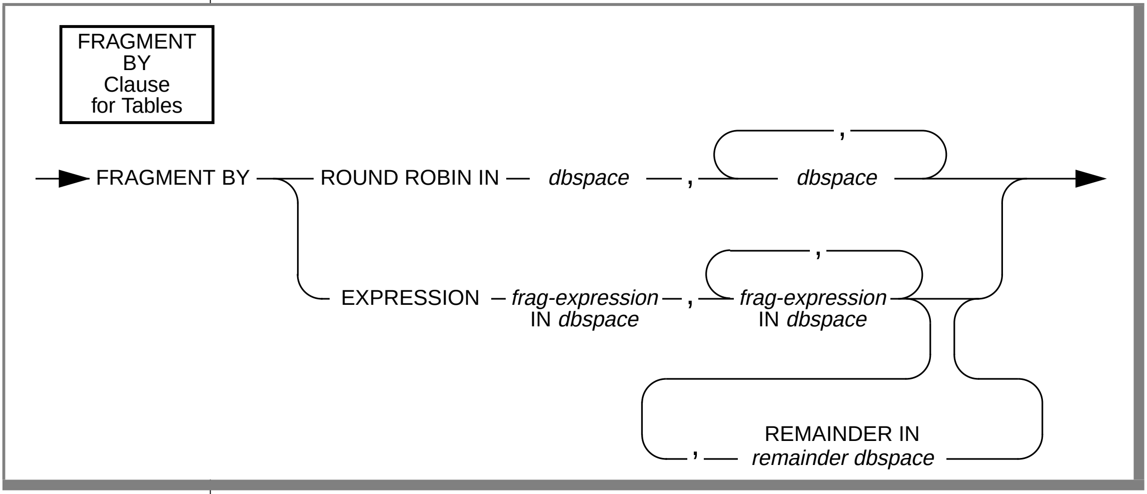
DETACH Clause



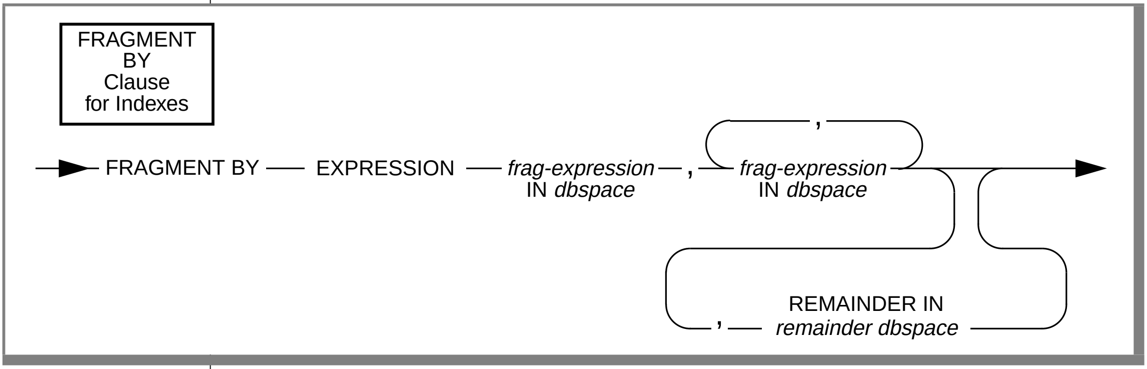
INIT Clause



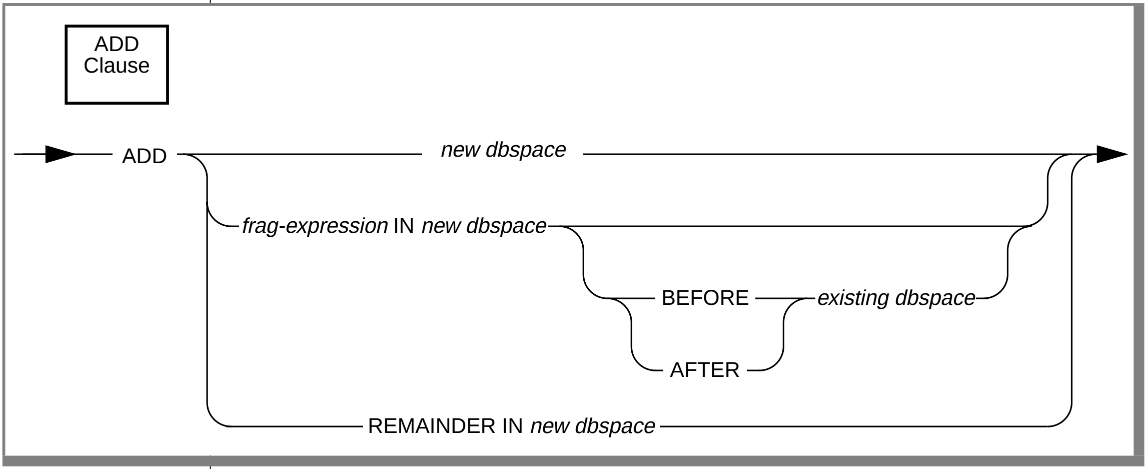
FRAGMENT BY Clause for Tables


FRAGMENT BY Clause for Tables
FRAGMENT BY
ROUND ROBIN IN
dbspace
,
dbspace
,
EXPRESSION
frag-expression IN dbspace
,
frag-expression IN dbspace
,
,
REMAINDER IN remainder dbspace

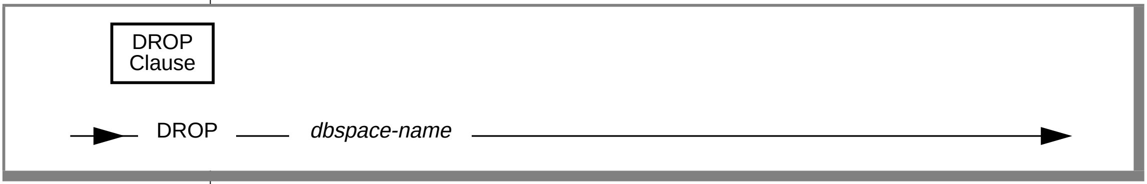
FRAGMENT BY Clause for Indexes


FRAGMENT BY Clause for Indexes
FRAGMENT BY
EXPRESSION
frag-expression IN dbspace
,
frag-expression IN dbspace
,
,
REMAINDER IN remainder dbspace

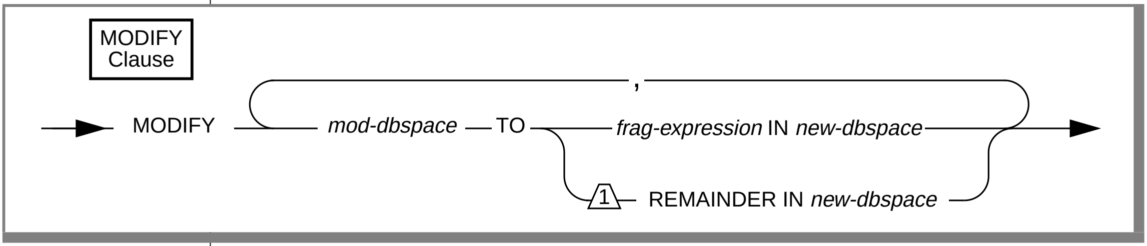
ADD Clause



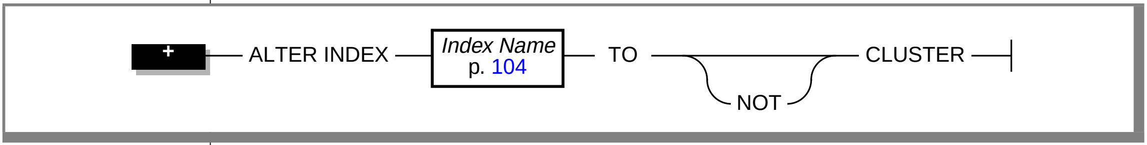
DROP Clause



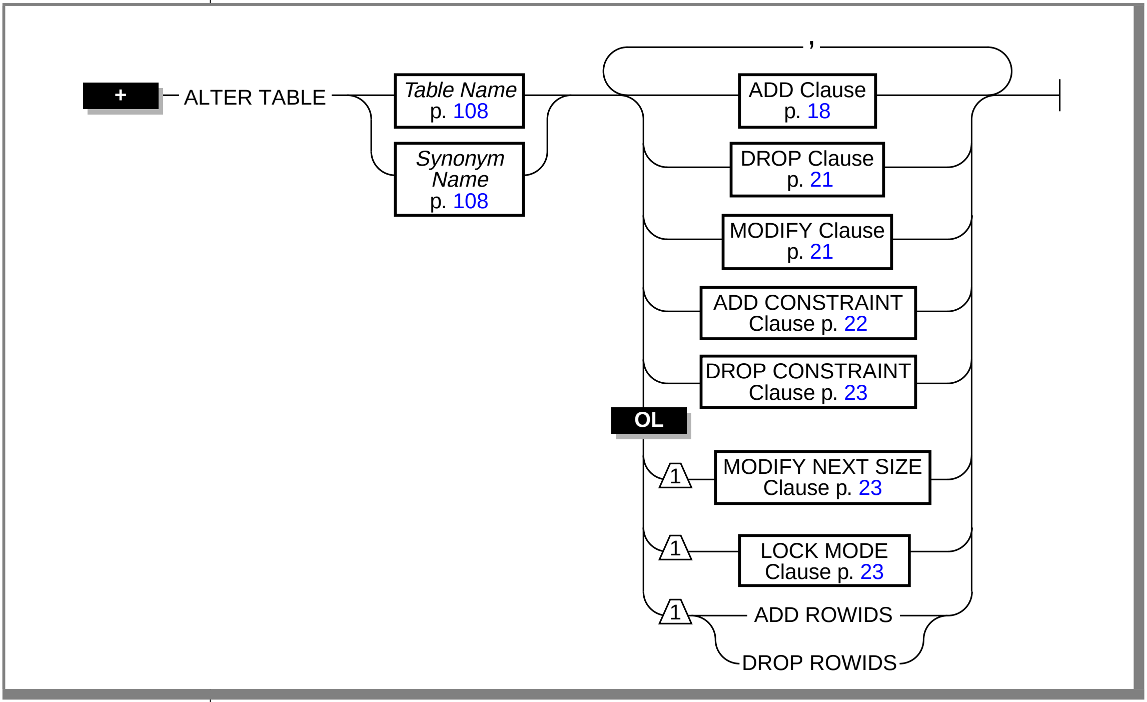
MODIFY Clause



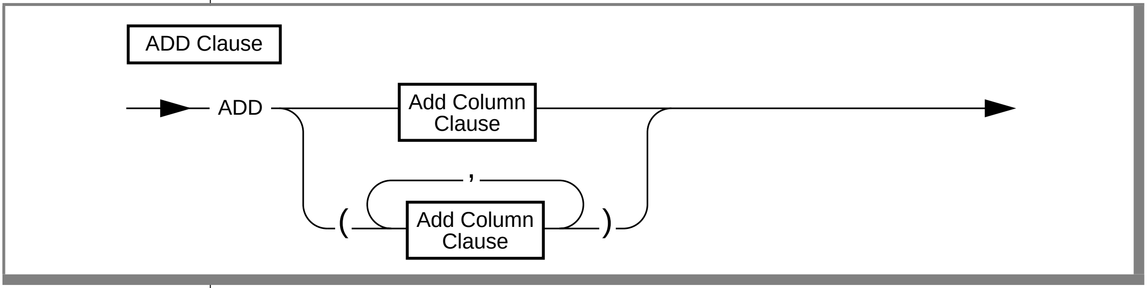
ALTER INDEX



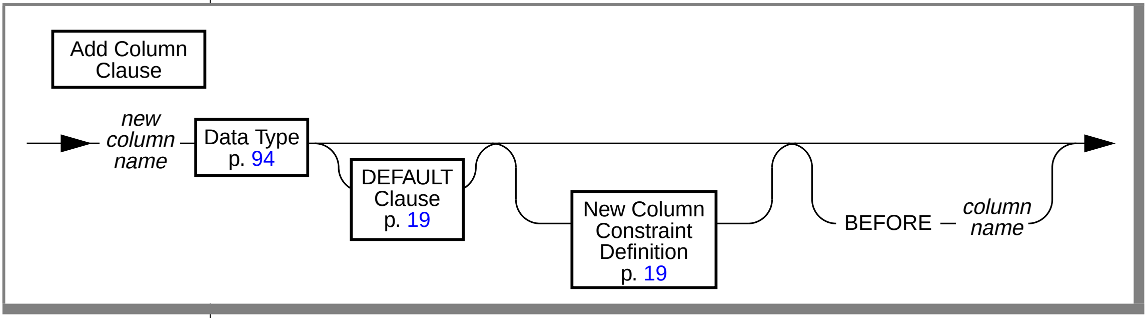
ALTER TABLE



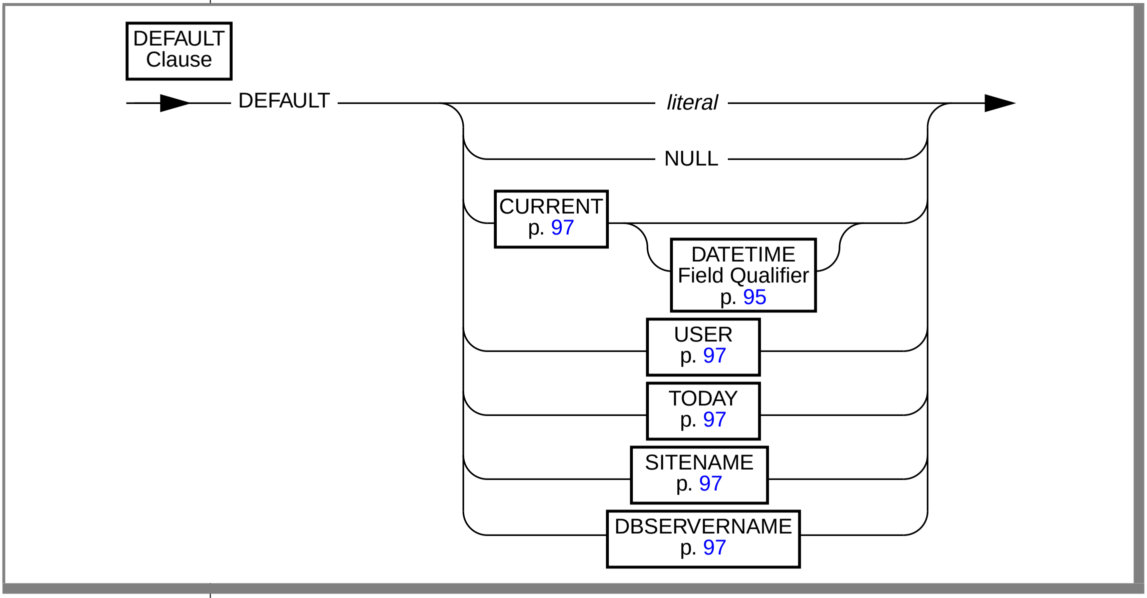
ADD Clause



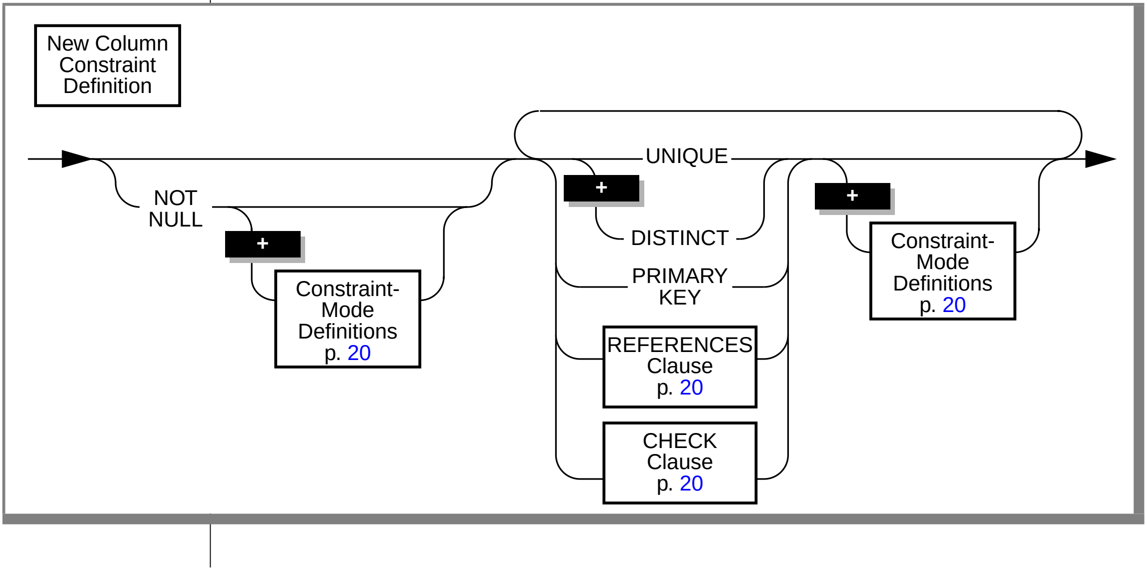
Add Column Clause



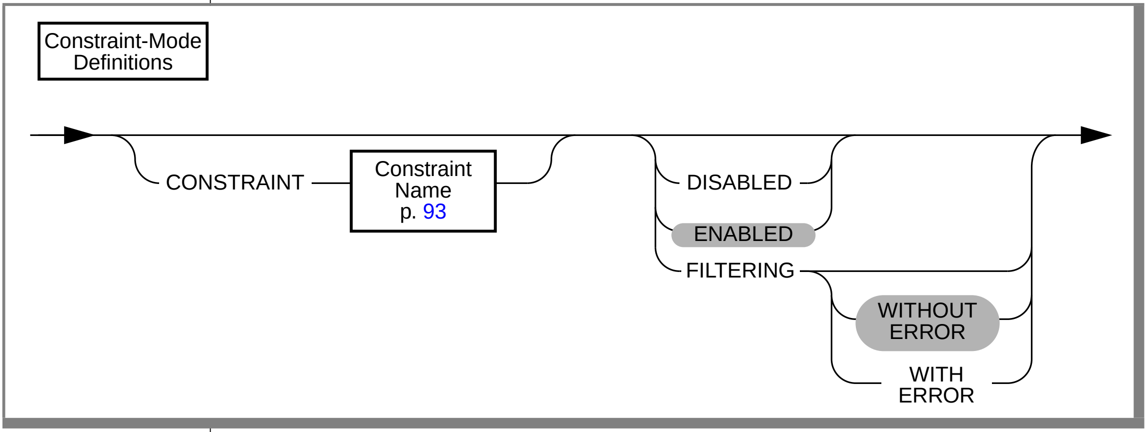
DEFAULT Clause



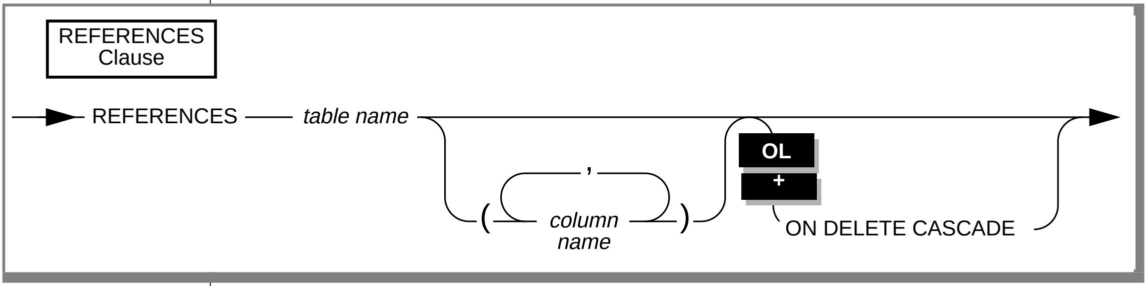
New Column Constraint Definition


New Column Constraint Definition
NOT NULL
+
Constraint-Mode Definitions p. 20
UNIQUE
+
DISTINCT
PRIMARY KEY
REFERENCES Clause p. 20
CHECK Clause p. 20
+
Constraint-Mode Definitions p. 20

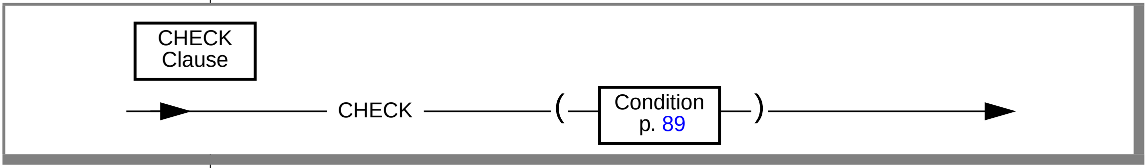
Constraint-Mode Definitions



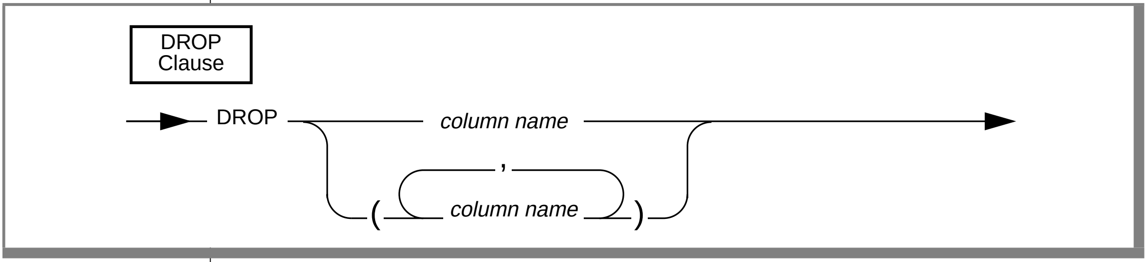
REFERENCES Clause



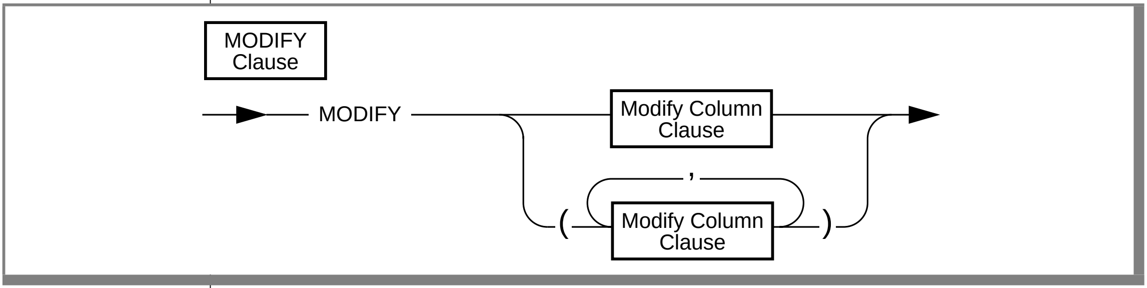
CHECK Clause



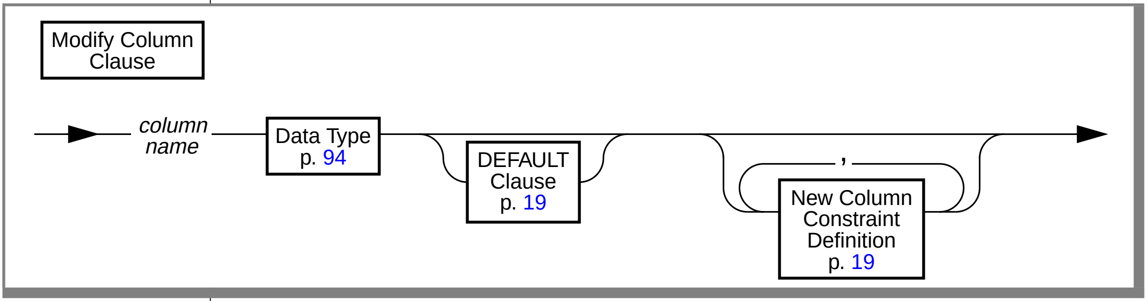
DROP Clause



MODIFY Clause



Modify Column Clause



ADD CONSTRAINT Clause

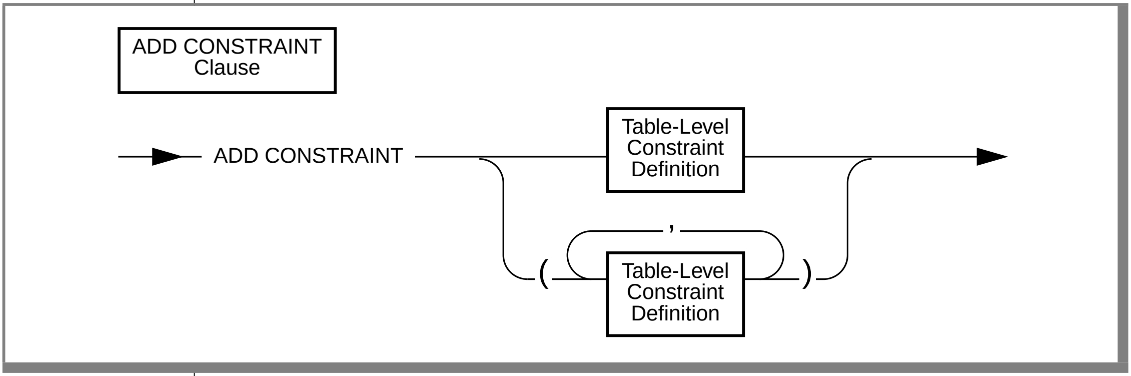
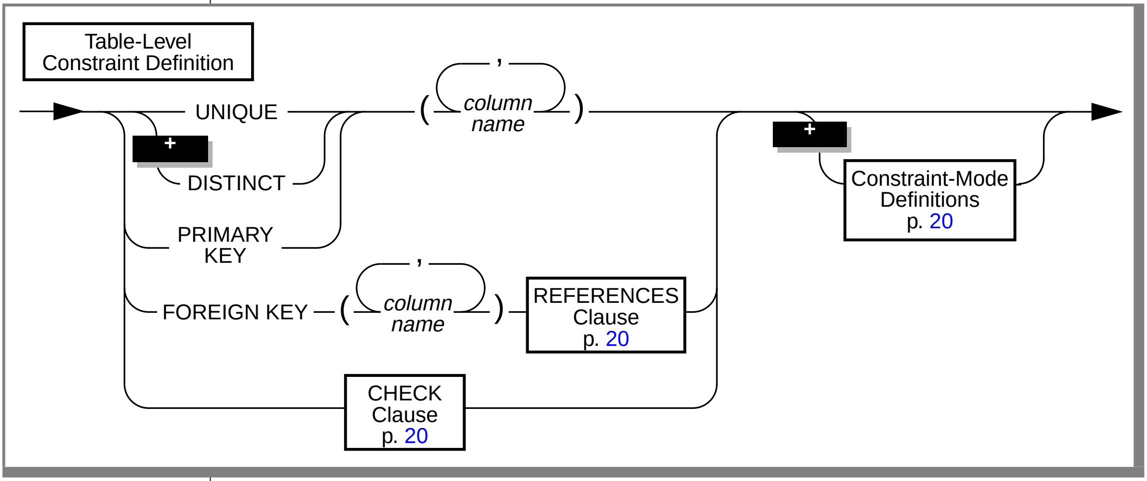
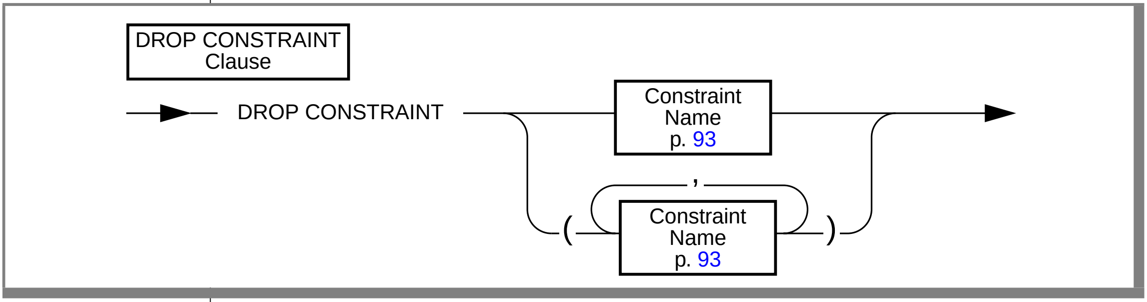

ADD CONSTRAINT Clause
ADD CONSTRAINT
Table-Level Constraint Definition
,
(
Table-Level Constraint Definition
)

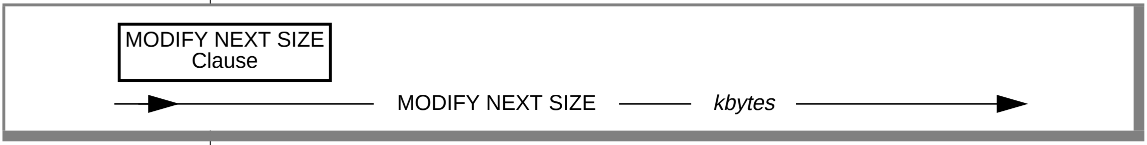
Table-Level Constraint Definition



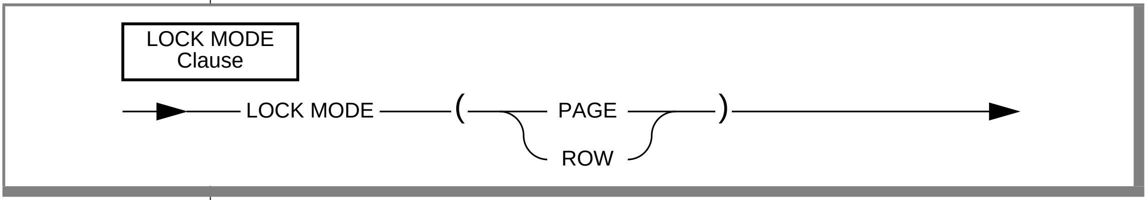
DROP CONSTRAINT Clause



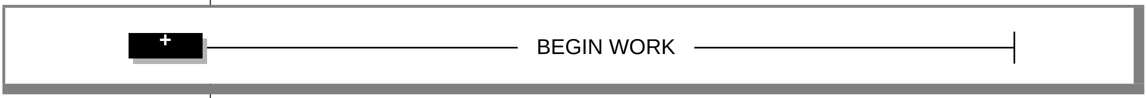
MODIFY NEXT SIZE Clause



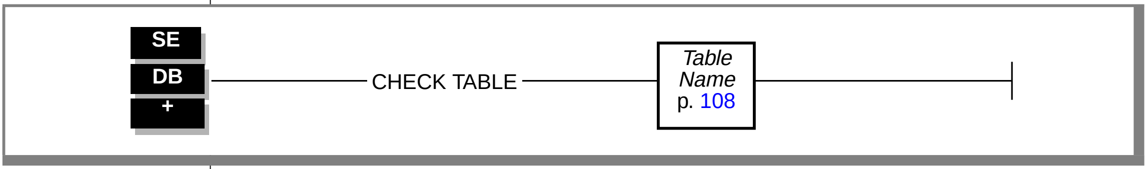
LOCK MODE Clause



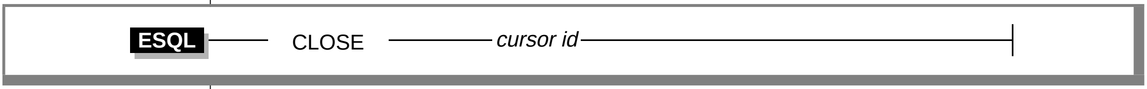
BEGIN WORK



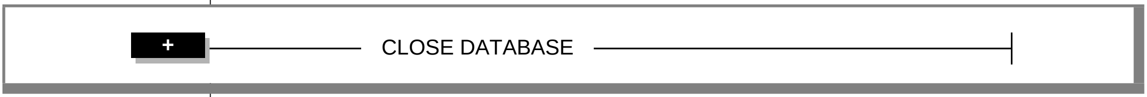
CHECK TABLE



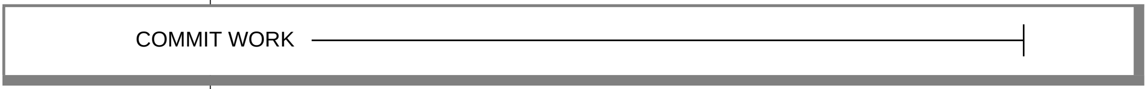
CLOSE



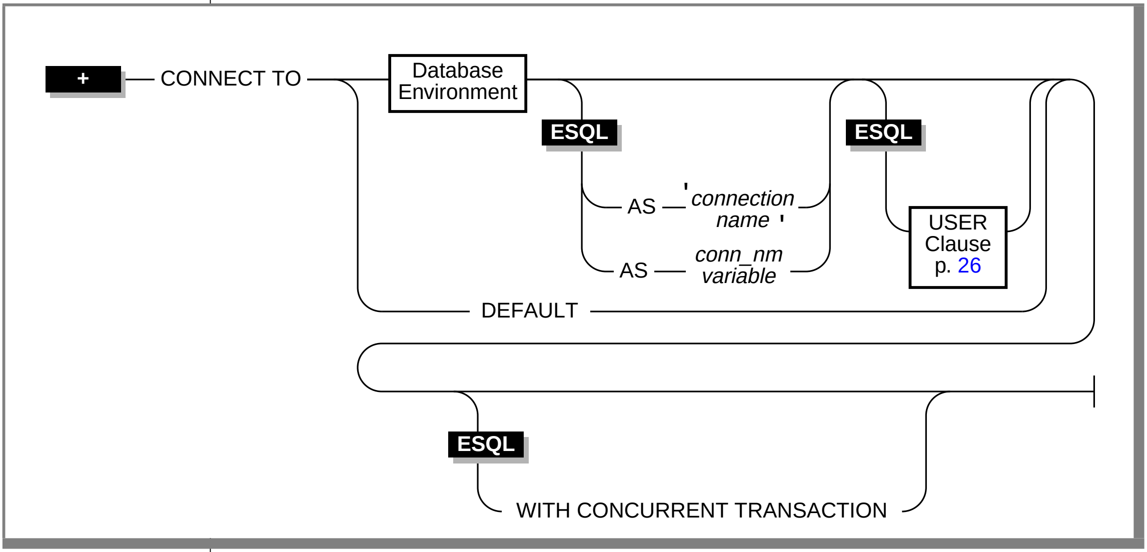
CLOSE DATABASE



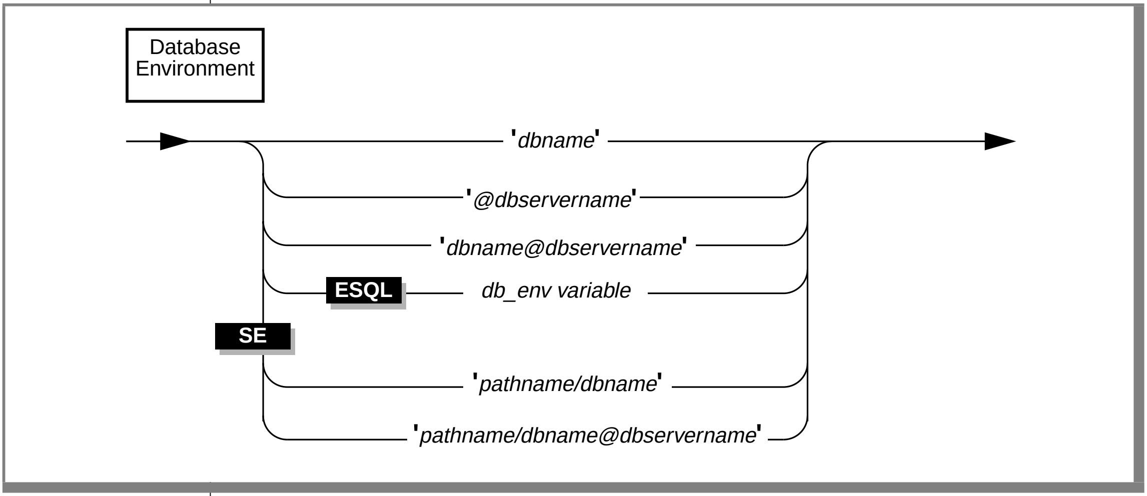
COMMIT WORK



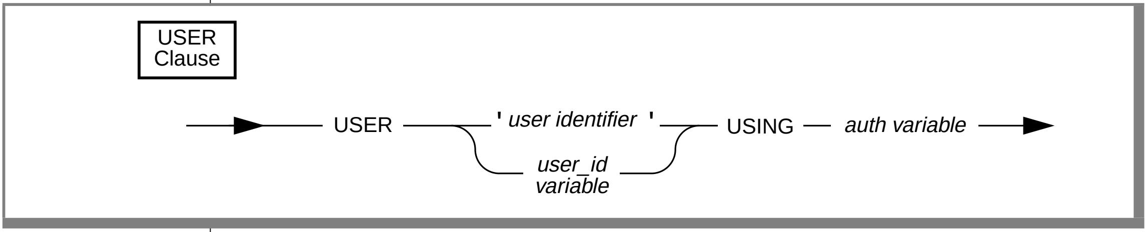
CONNECT



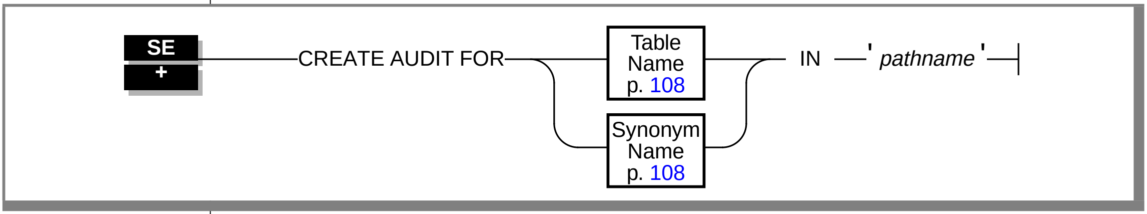
Database Environment



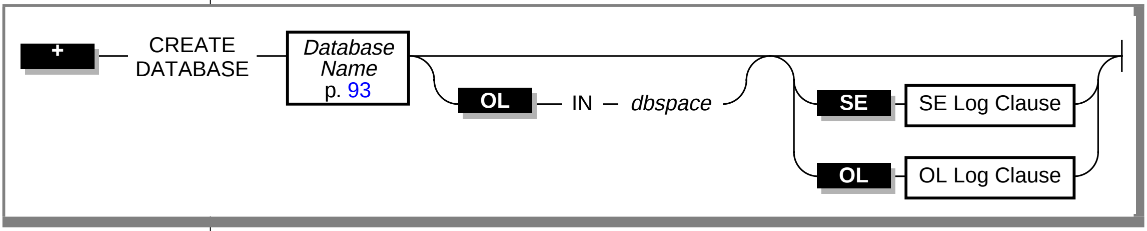
USER Clause



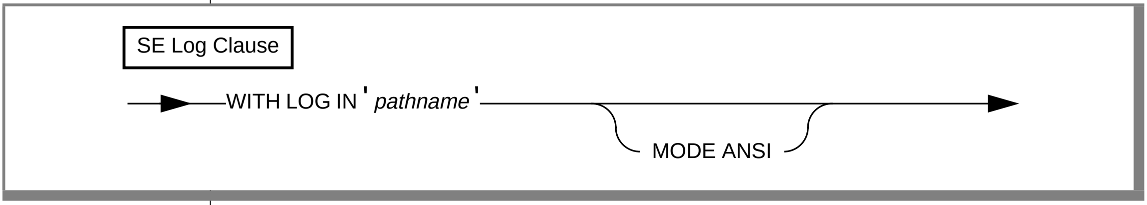
CREATE AUDIT



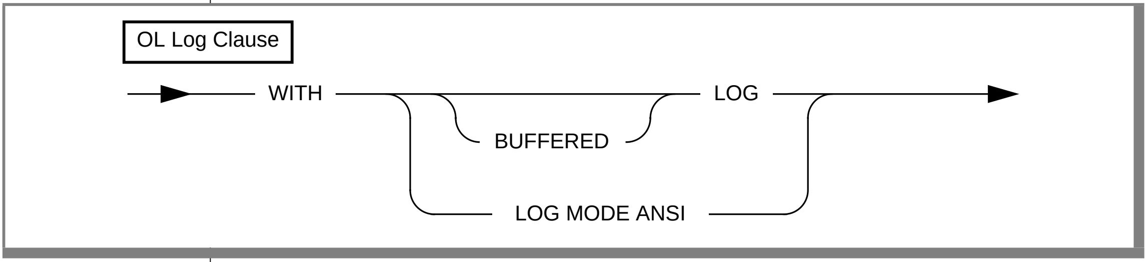
CREATE DATABASE



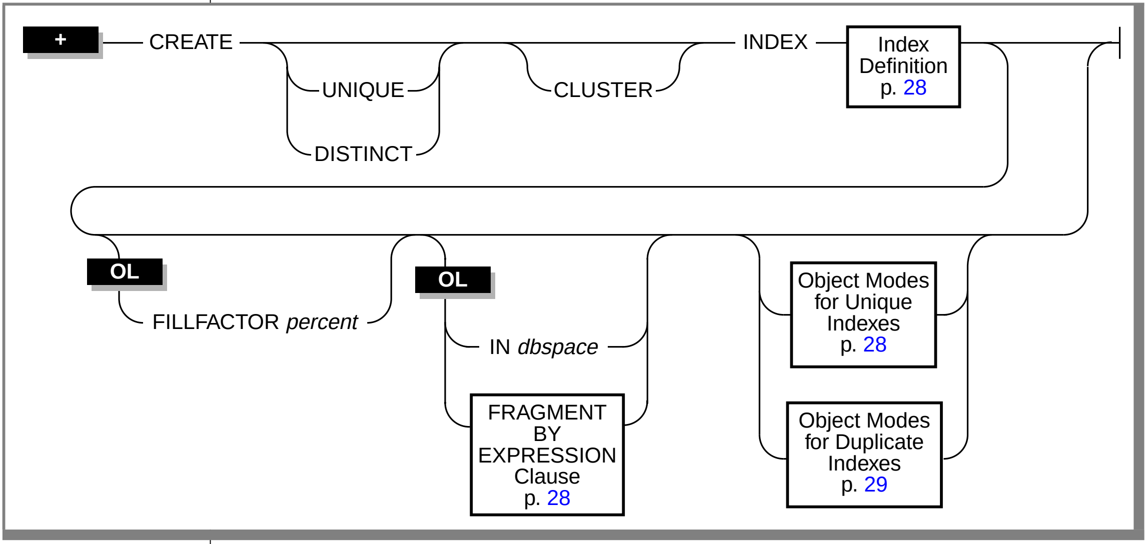
SE LOG Clause



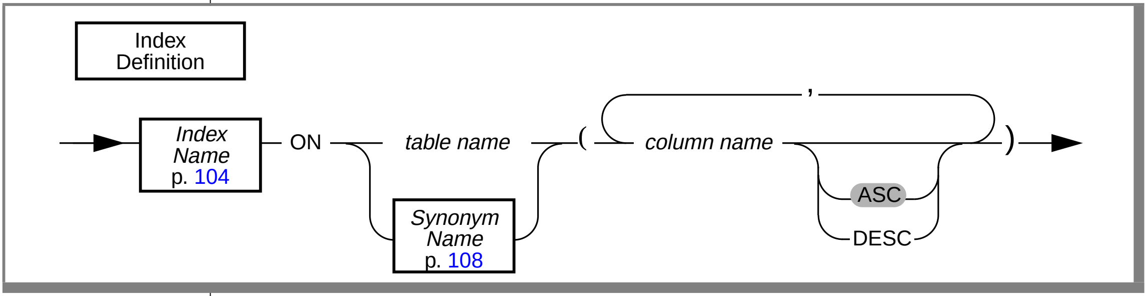
OL LOG Clause



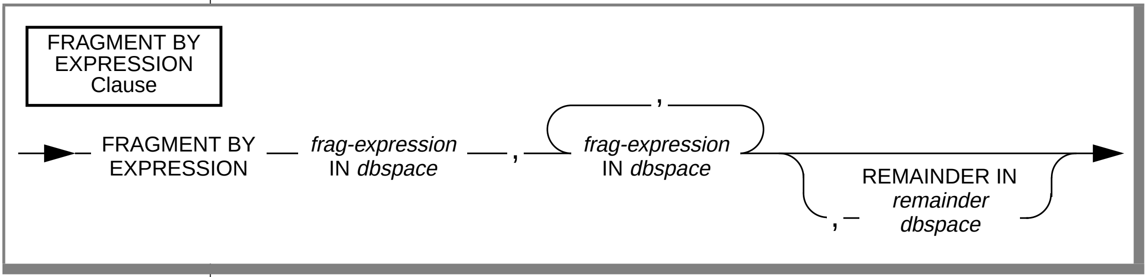
CREATE INDEX


+
CREATE
UNIQUE
DISTINCT
CLUSTER
INDEX
Index Definition p. 28
OL
FILLFACTOR *percent*
OL
IN *dbspace*
FRAGMENT BY EXPRESSION Clause p. 28
Object Modes for Unique Indexes p. 28
Object Modes for Duplicate Indexes p. 29

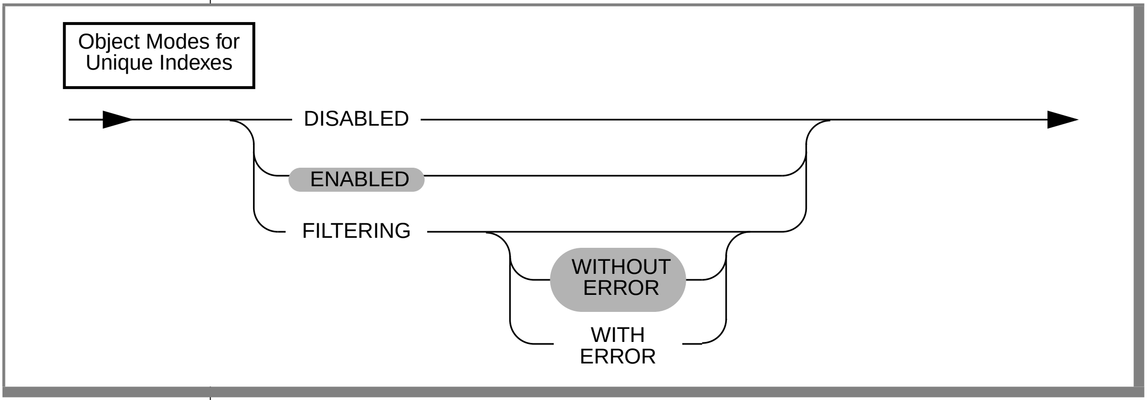
Index Definition



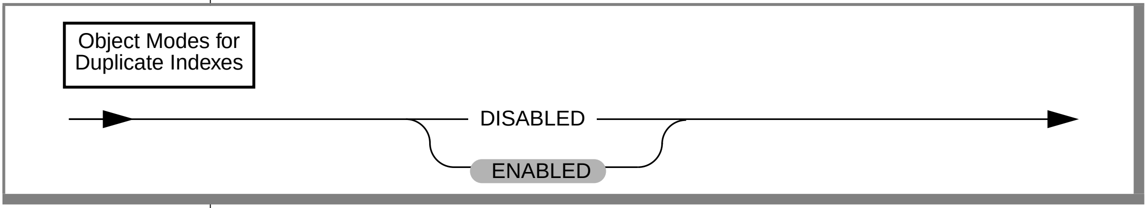
FRAGMENT BY EXPRESSION Clause


FRAGMENT BY EXPRESSION Clause
FRAGMENT BY EXPRESSION
frag-expression IN dbspace
,
frag-expression IN dbspace
,
REMAINDER IN remainder dbspace
,

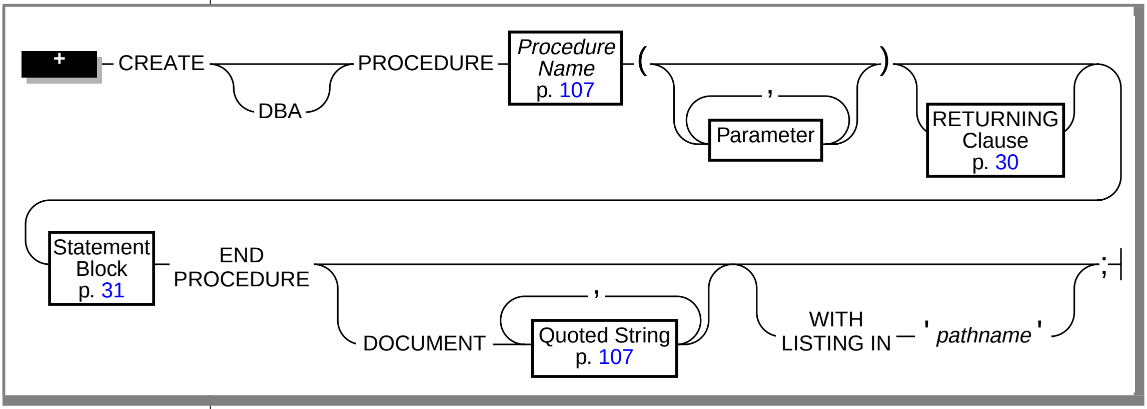
Object Modes for Unique Indexes



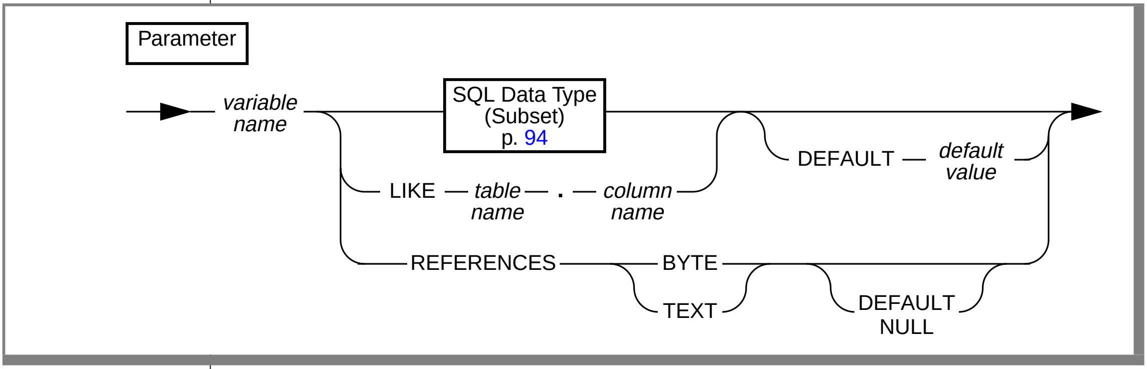
Object Modes for Duplicate Indexes



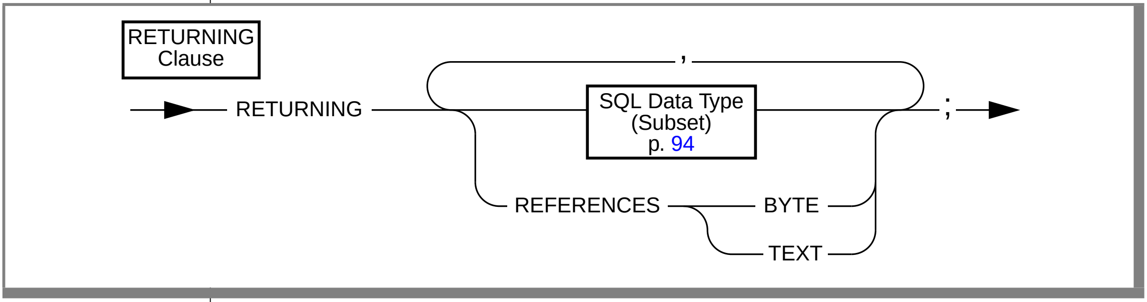
CREATE PROCEDURE



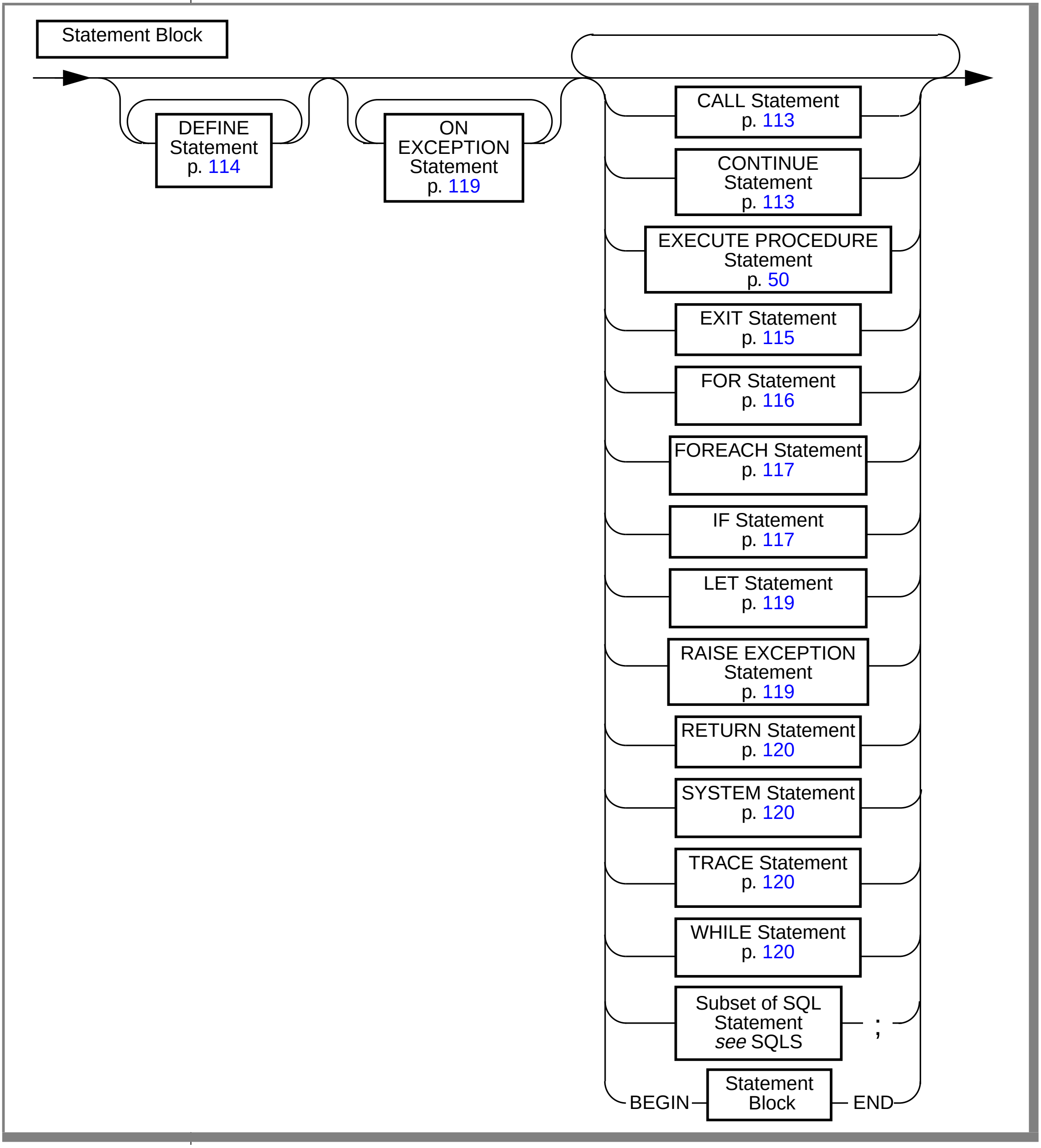
Parameter Option



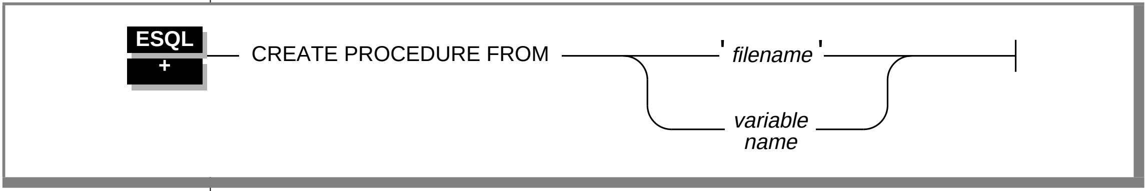
RETURNING Clause



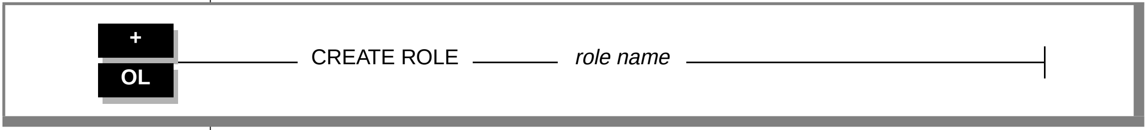
Statement Block



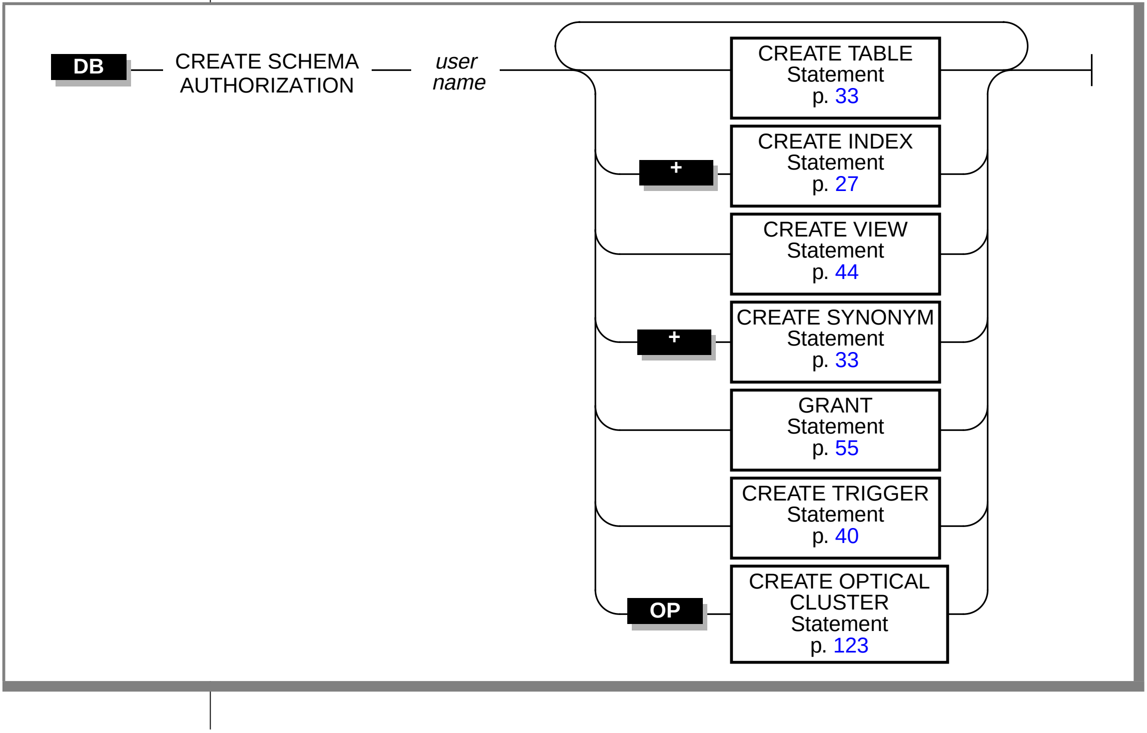
CREATE PROCEDURE FROM



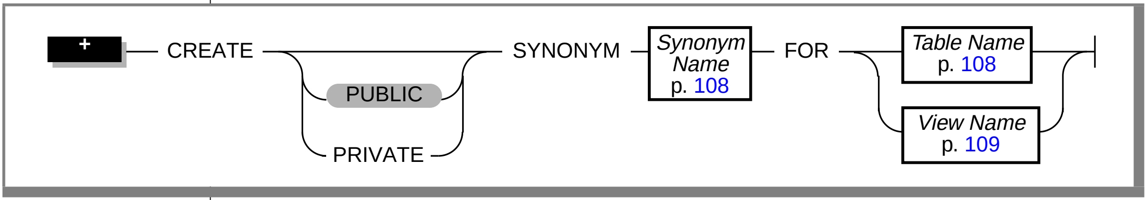
CREATE ROLE



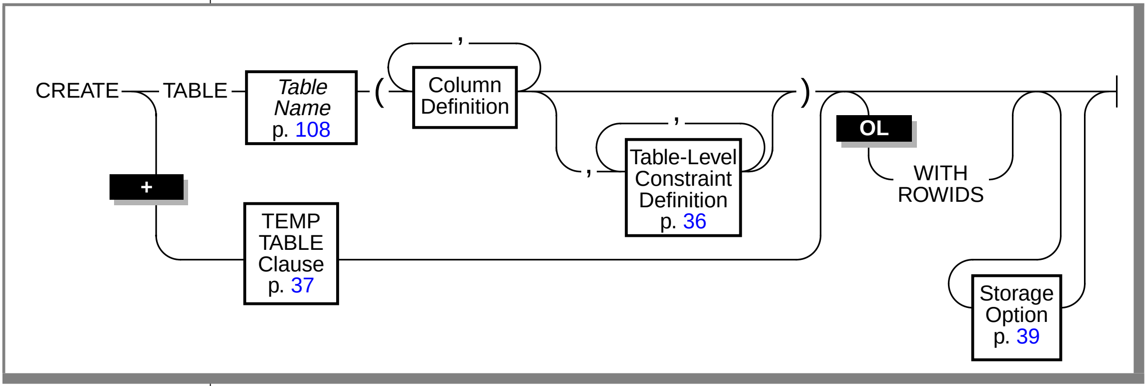
CREATE SCHEMA


DB
CREATE SCHEMA AUTHORIZATION
user name
CREATE TABLE Statement p. 33
+
CREATE INDEX Statement p. 27
CREATE VIEW Statement p. 44
+
CREATE SYNONYM Statement p. 33
GRANT Statement p. 55
CREATE TRIGGER Statement p. 40
OP
CREATE OPTICAL CLUSTER Statement p. 123

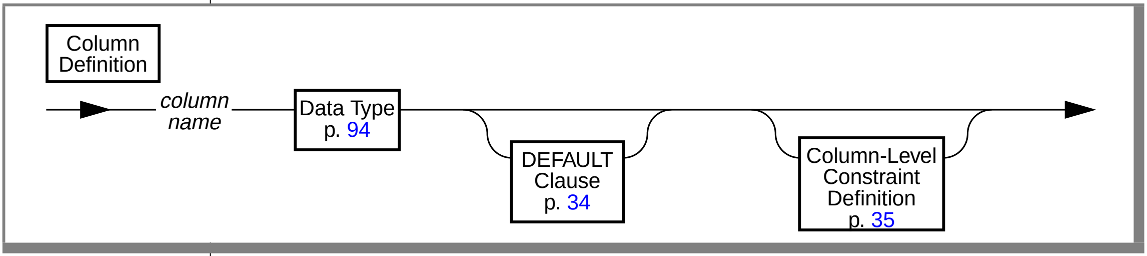
CREATE SYNONYM



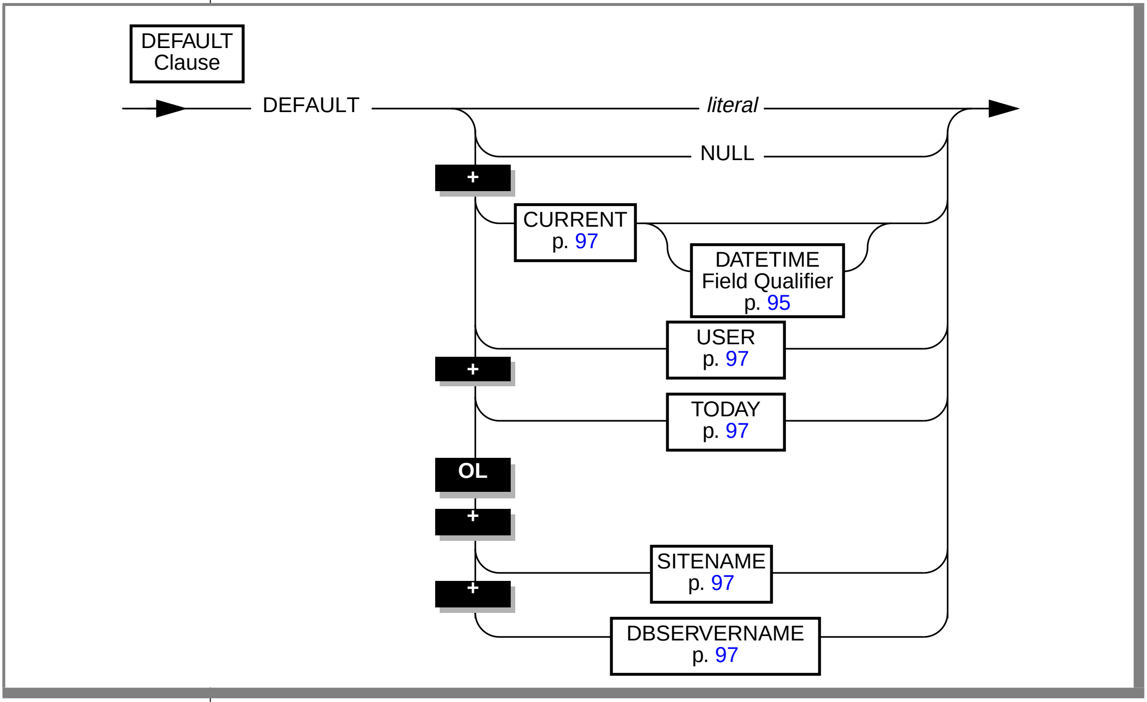
CREATE TABLE



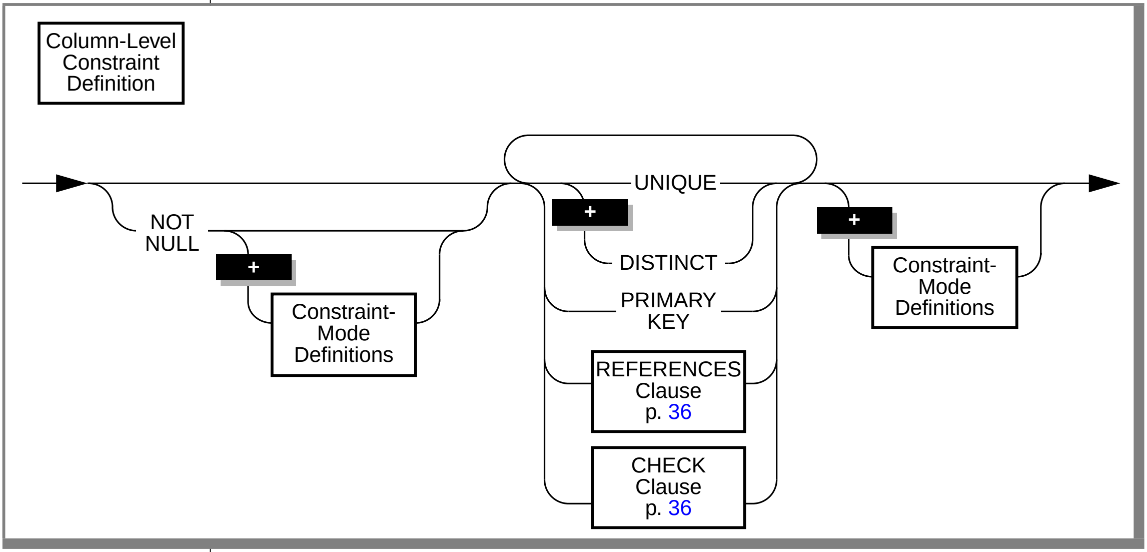
Column-Definition Option



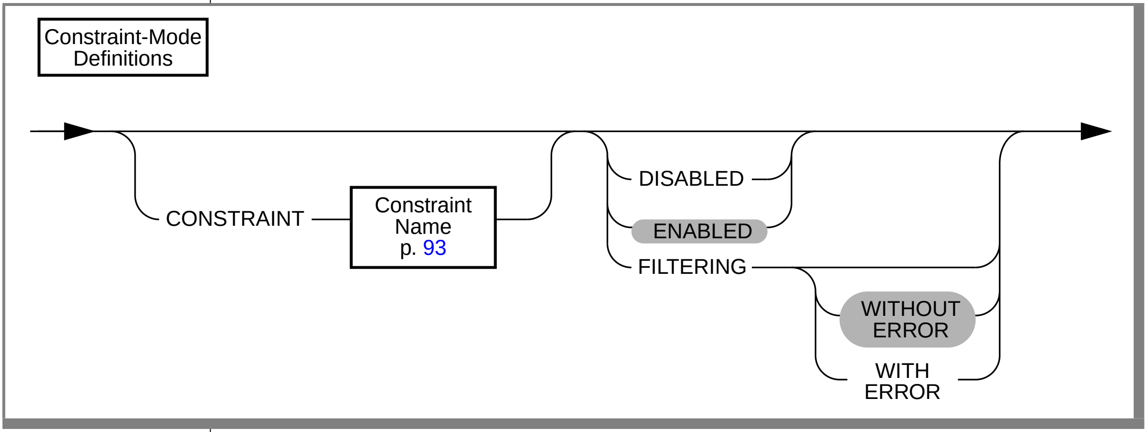
DEFAULT Clause



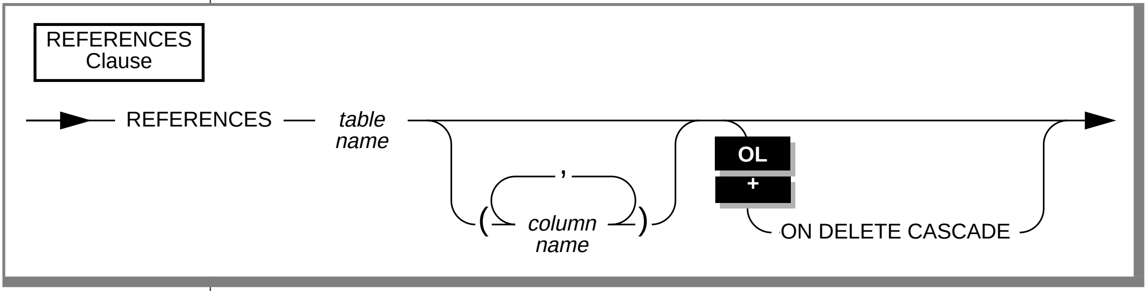
Column-Level Constraint-Definition Option


Column-Level Constraint Definition
NOT NULL
+
Constraint-Mode Definitions
UNIQUE
+
DISTINCT
PRIMARY KEY
REFERENCES Clause p. 36
CHECK Clause p. 36
+
Constraint-Mode Definitions

Constraint-Mode Definitions



REFERENCES Clause



CHECK Clause

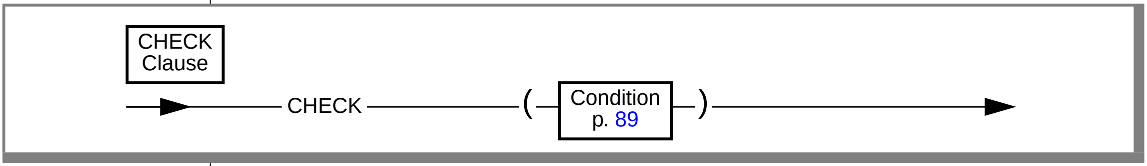
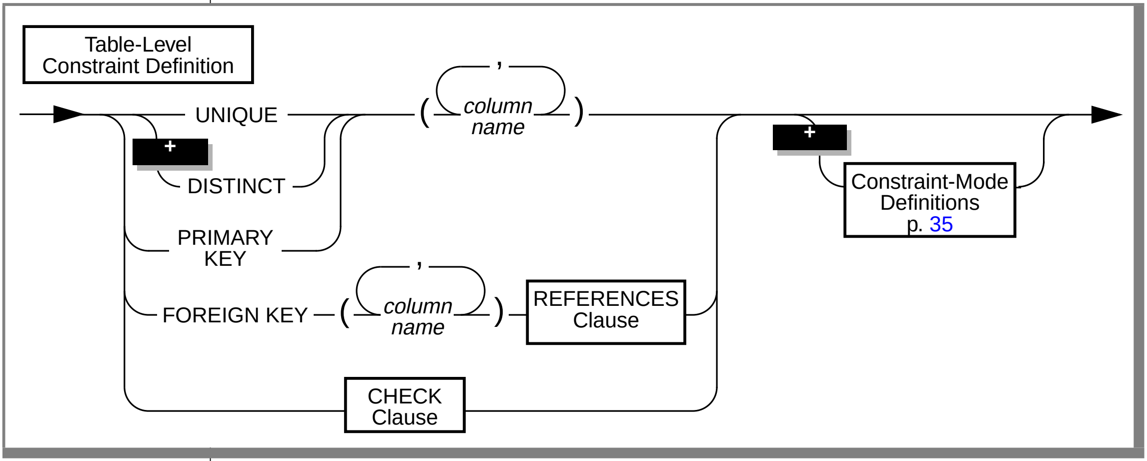
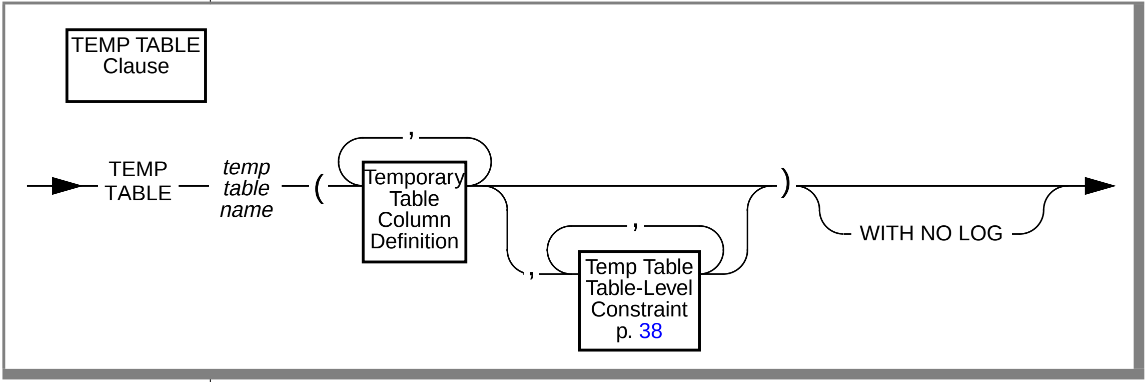


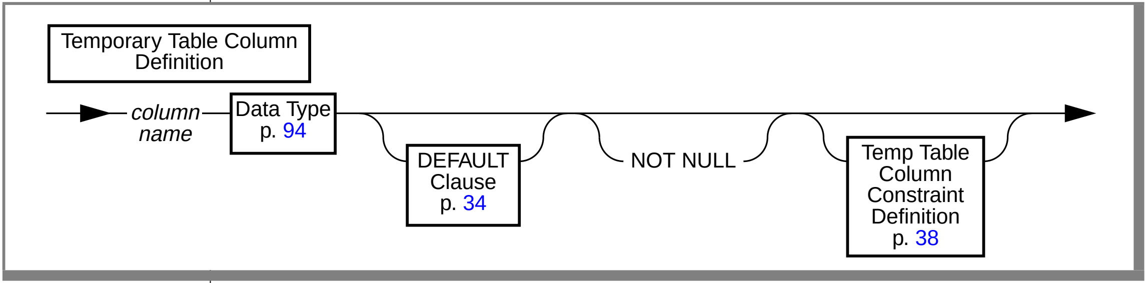
Table-Level Constraint-Definition Option



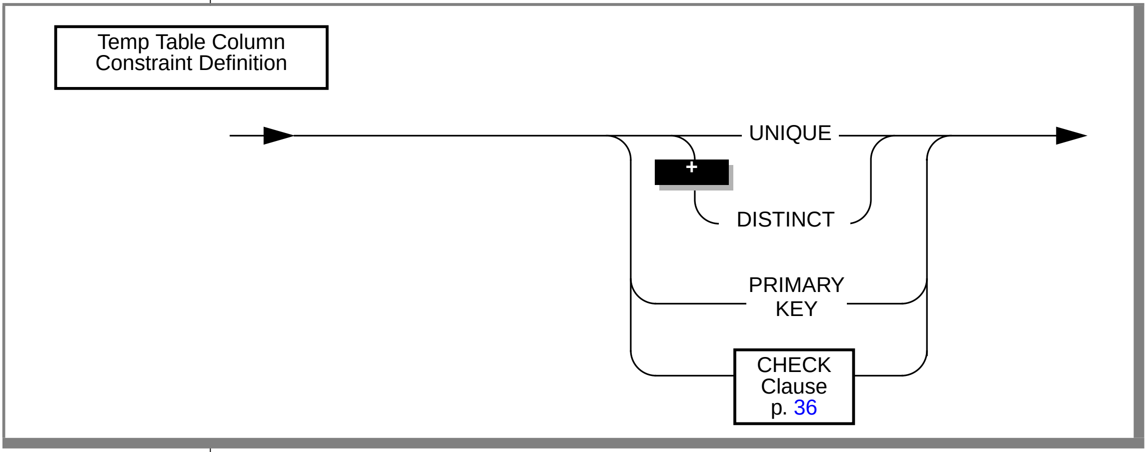
TEMP TABLE Clause


TEMP TABLE Clause
TEMP TABLE
temp table name
(
,
Temporary Table Column Definition
,
,
Temp Table Table-Level Constraint p. 38
)
WITH NO LOG

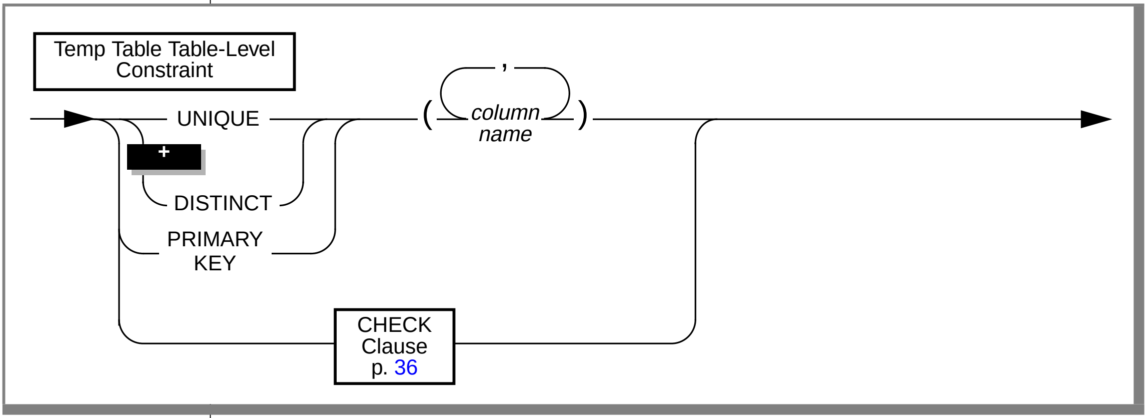
Temporary Table Column Definition


Temporary Table Column Definition
column name
Data Type p. 94
DEFAULT Clause p. 34
NOT NULL
Temp Table Column Constraint Definition p. 38

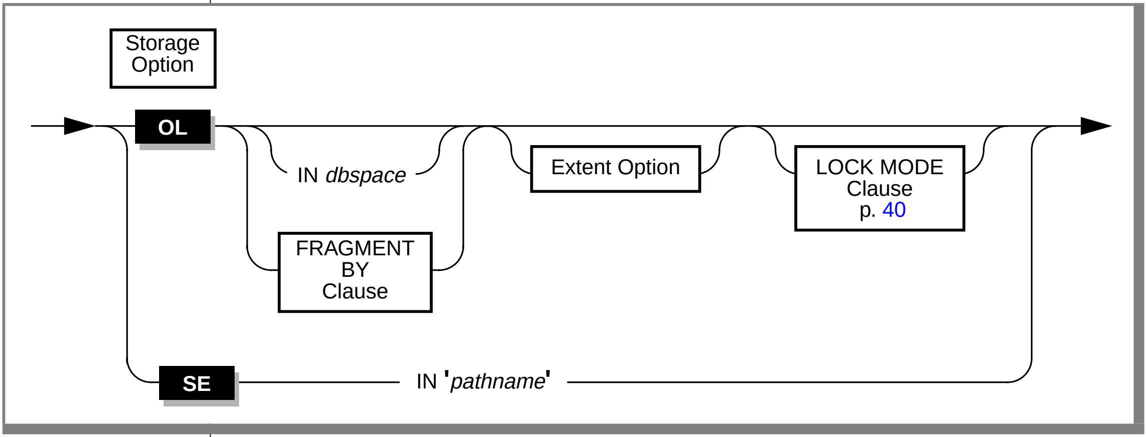
Temp Table Column Constraint Definition



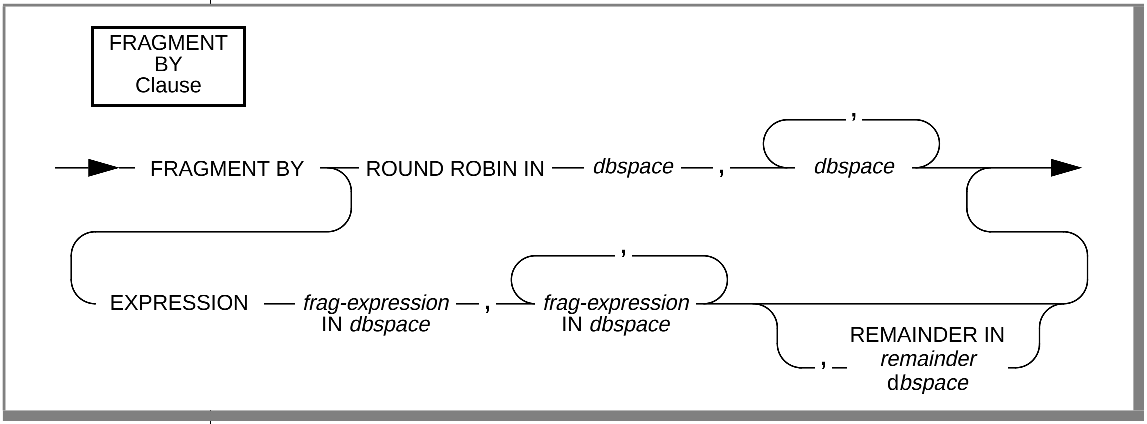
Temp Table Table-Level Constraint



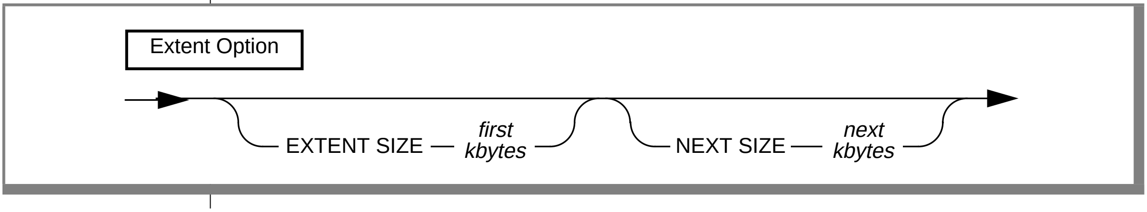
Storage Option



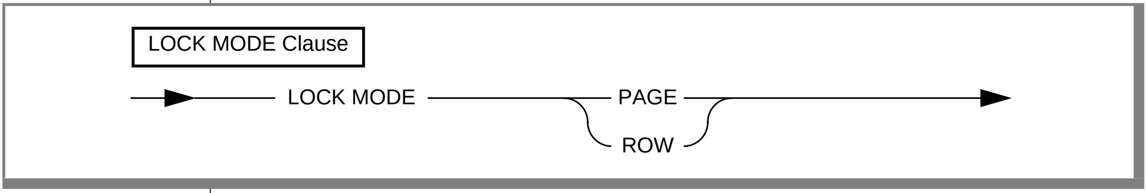
FRAGMENT BY Clause


FRAGMENT BY Clause
FRAGMENT BY
ROUND ROBIN IN
dbspace
,
dbspace
,
EXPRESSION
frag-expression IN dbspace
,
frag-expression IN dbspace
,
REMAINDER IN remainder dbspace
,

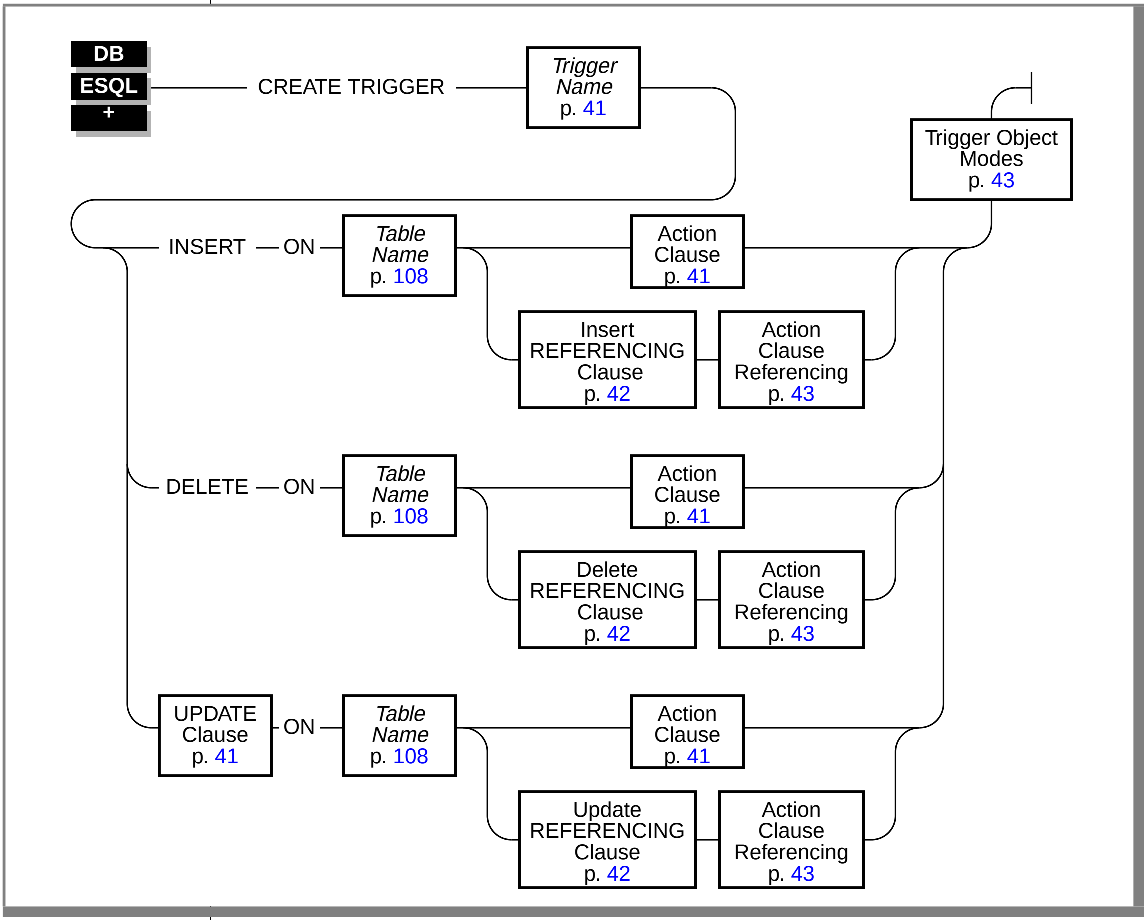
Extent Option



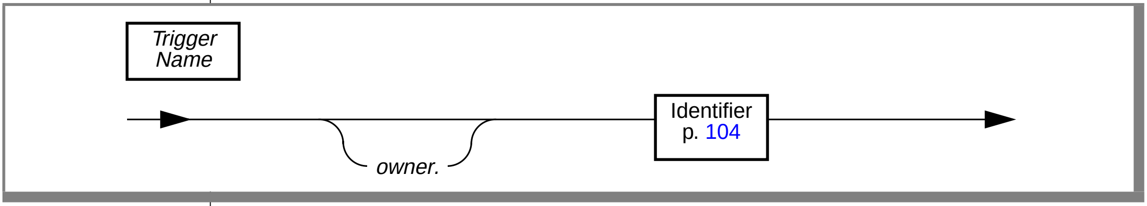
LOCK MODE Clause



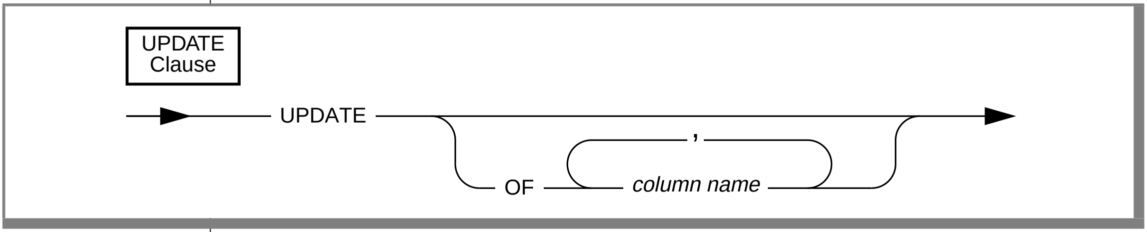
CREATE TRIGGER


DB
ESQL
+
CREATE TRIGGER
Trigger Name p. 41
Trigger Object Modes p. 43
INSERT
ON
Table Name p. 108
Action Clause p. 41
Insert REFERENCING Clause p. 42
Action Clause Referencing p. 43
DELETE
ON
Table Name p. 108
Action Clause p. 41
Delete REFERENCING Clause p. 42
Action Clause Referencing p. 43
UPDATE Clause p. 41
ON
Table Name p. 108
Action Clause p. 41
Update REFERENCING Clause p. 42
Action Clause Referencing p. 43

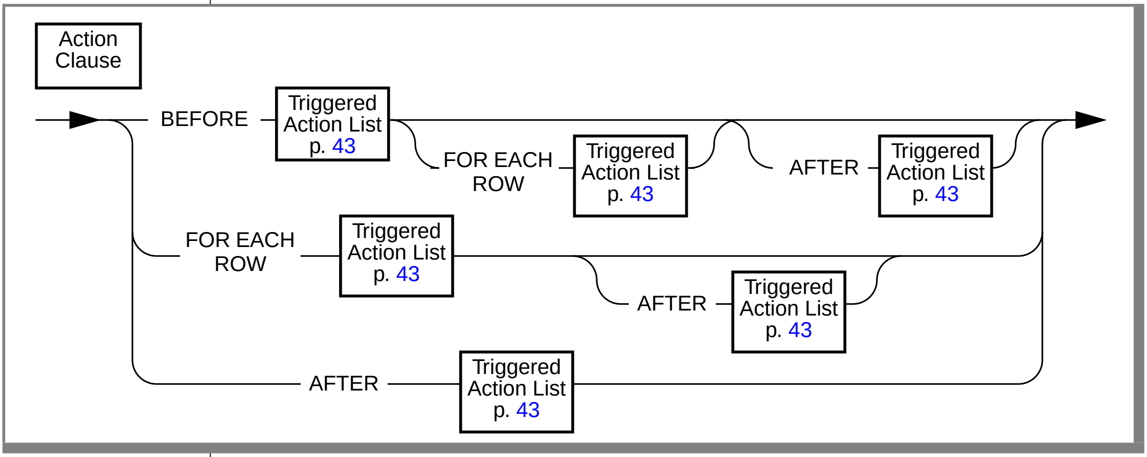
Trigger Name



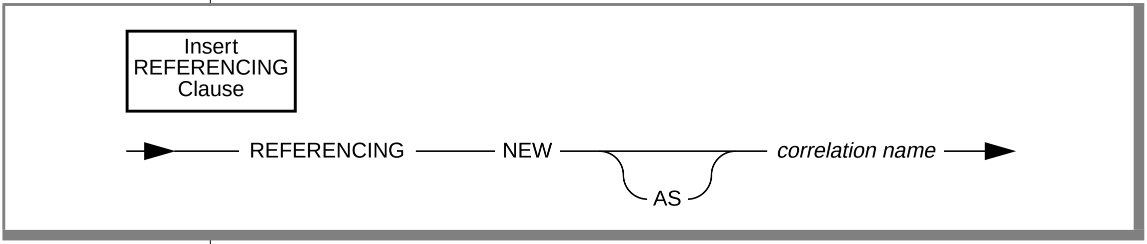
UPDATE Clause



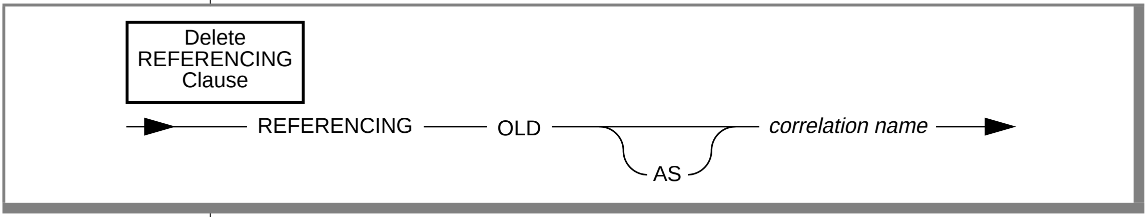
Action Clause


Action Clause
BEFORE
Triggered Action List p. 43
FOR EACH ROW
Triggered Action List p. 43
AFTER
Triggered Action List p. 43
FOR EACH ROW
Triggered Action List p. 43
AFTER
Triggered Action List p. 43
AFTER
Triggered Action List p. 43

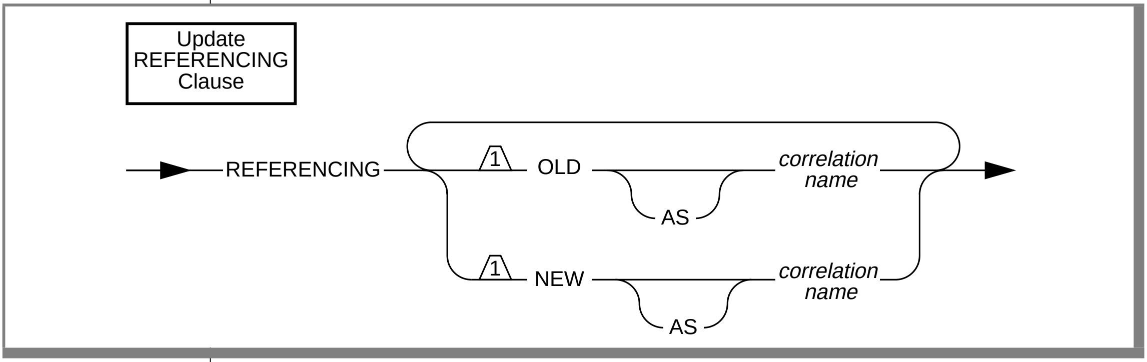
Insert REFERENCING Clause



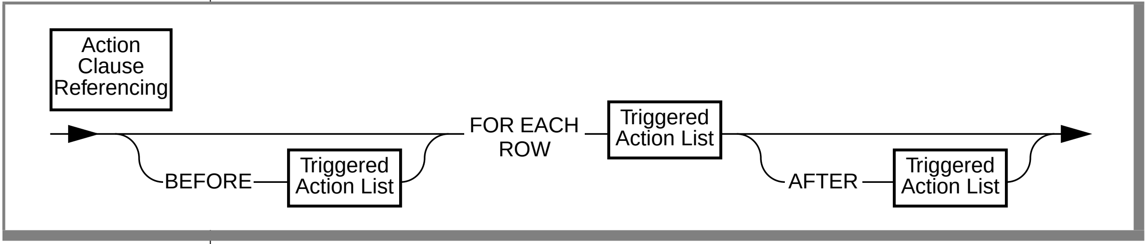
Delete REFERENCING Clause



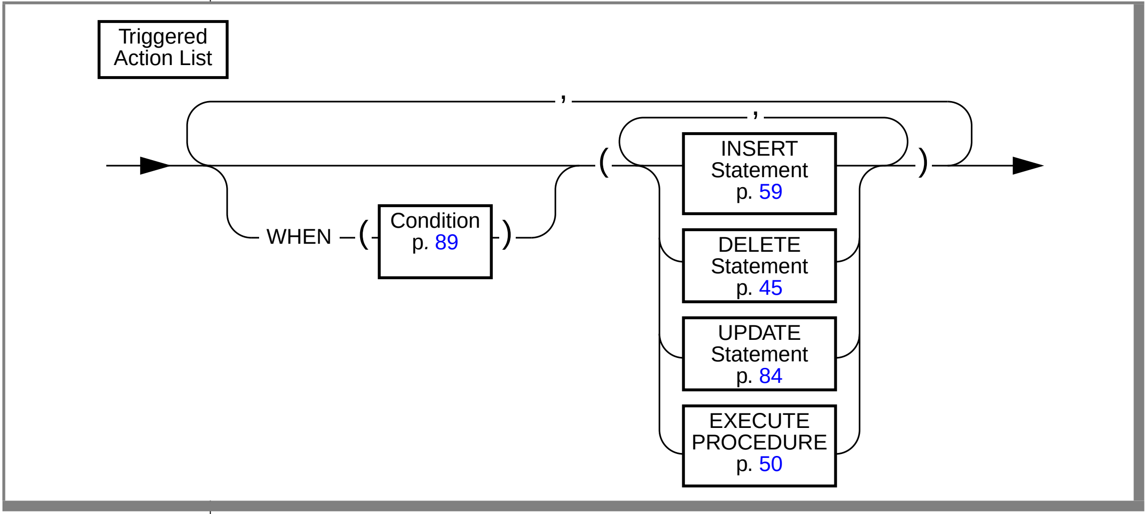
Update REFERENCING Clause



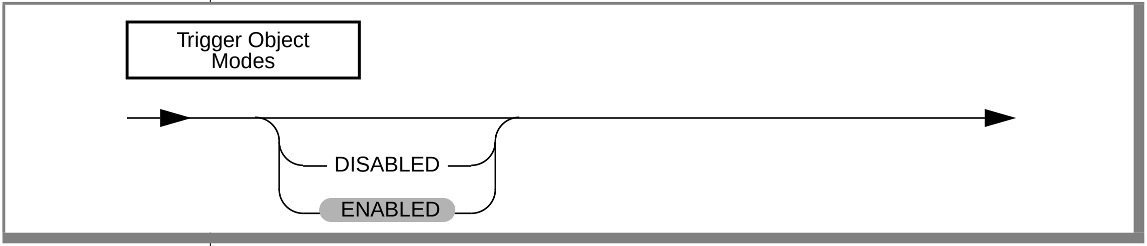
Action Clause Referencing



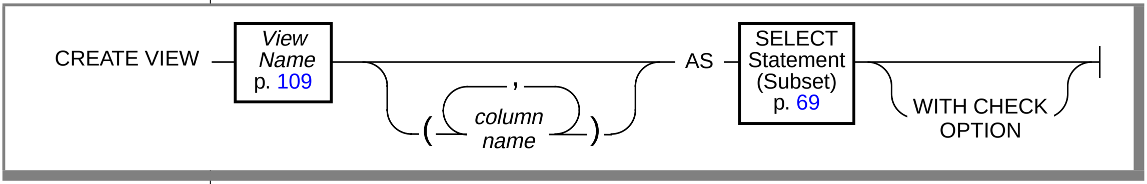
Triggered Action List



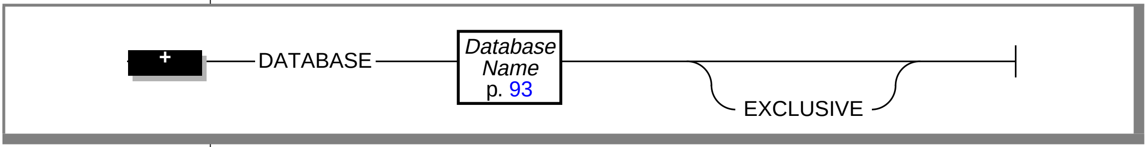
Trigger Object Modes



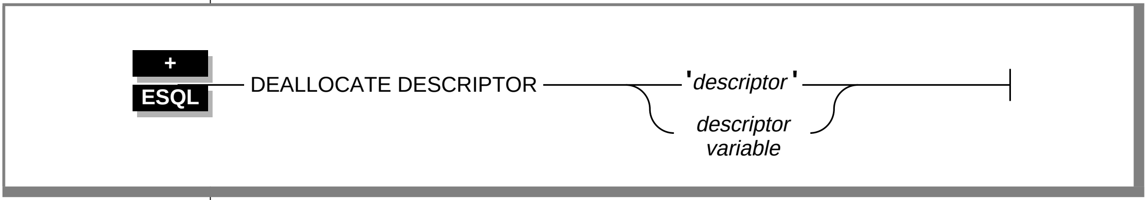
CREATE VIEW



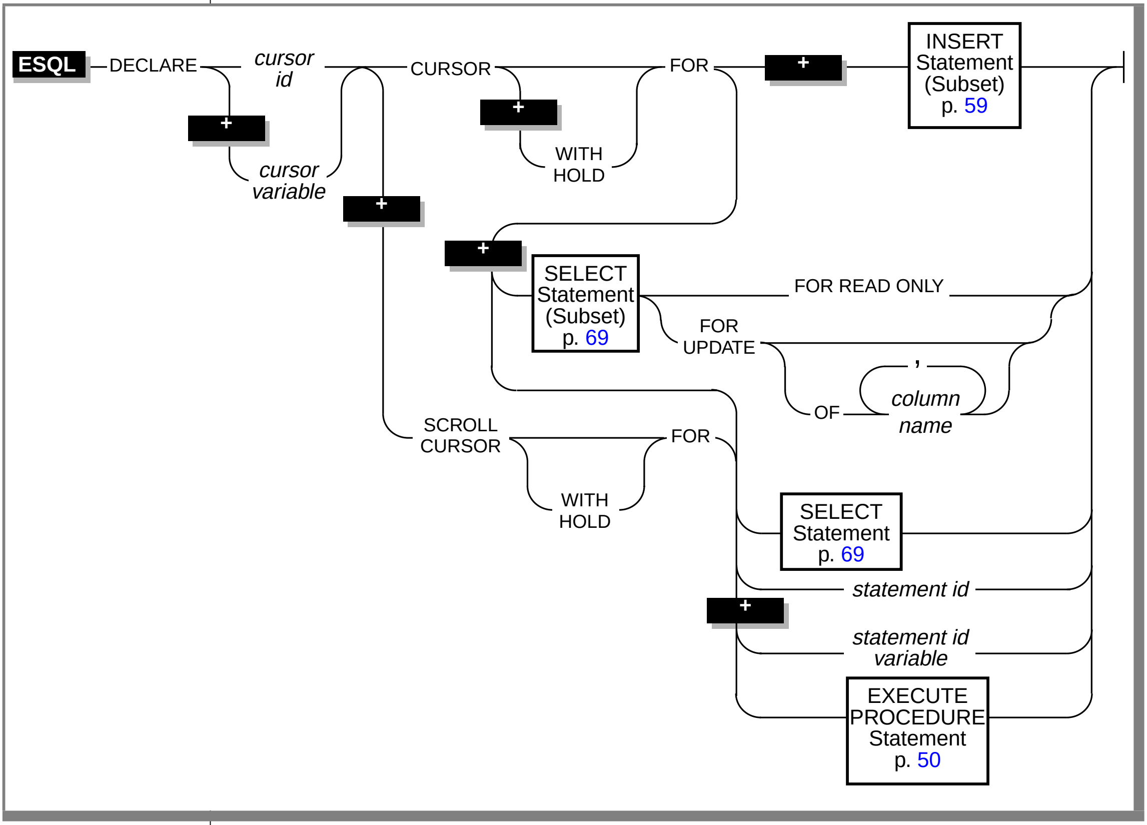
DATABASE



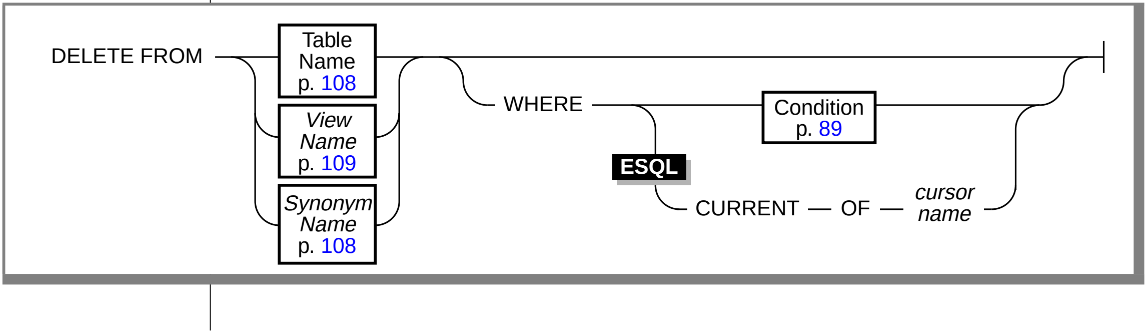
DEALLOCATE DESCRIPTOR



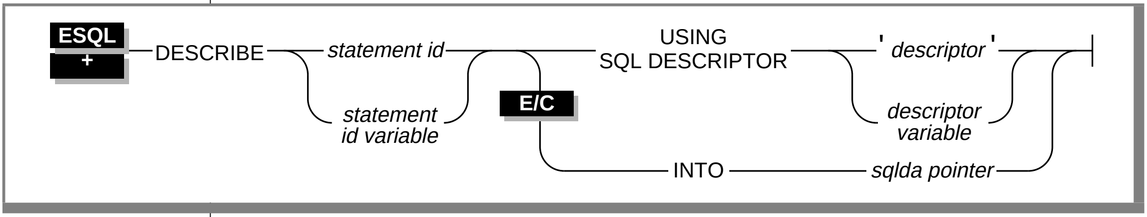
DECLARE



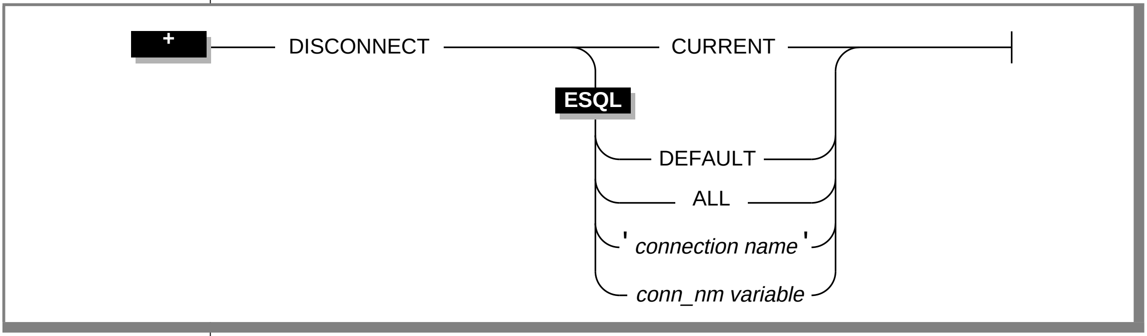
DELETE



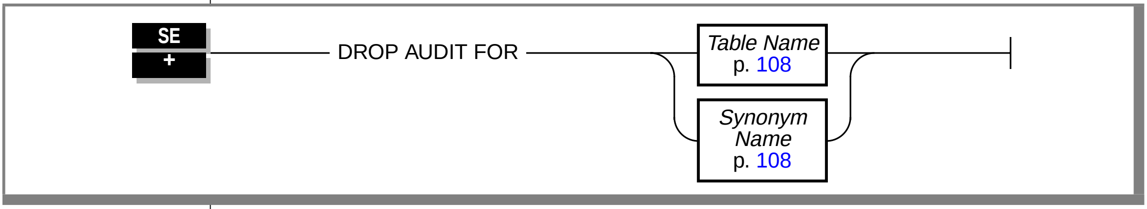
DESCRIBE



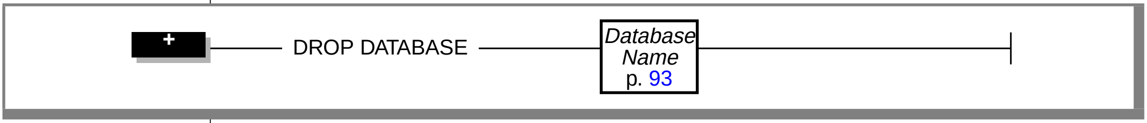
DISCONNECT



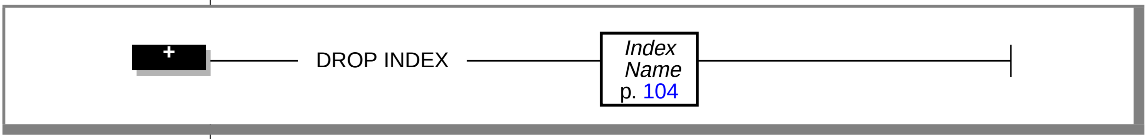
DROP AUDIT



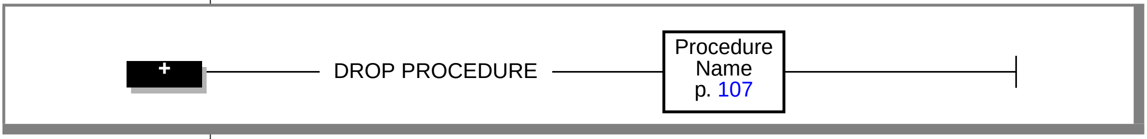
DROP DATABASE



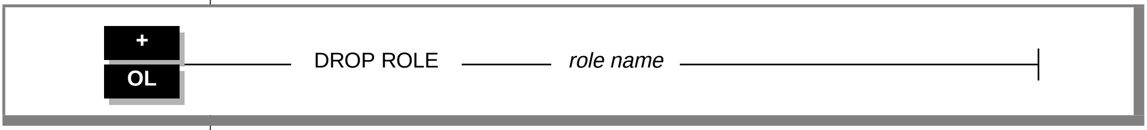
DROP INDEX



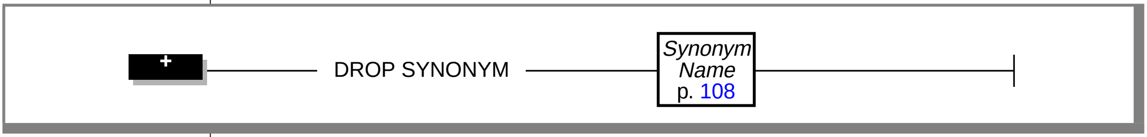
DROP PROCEDURE



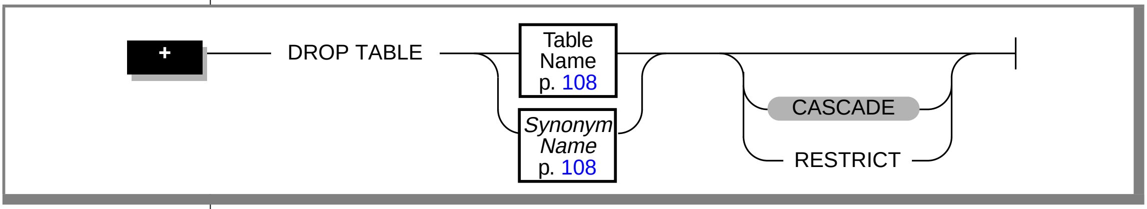
DROP ROLE



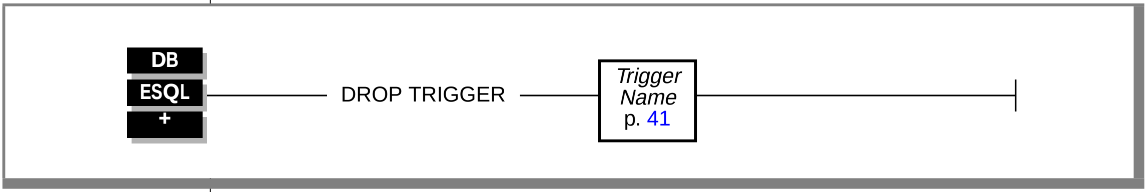
DROP SYNONYM



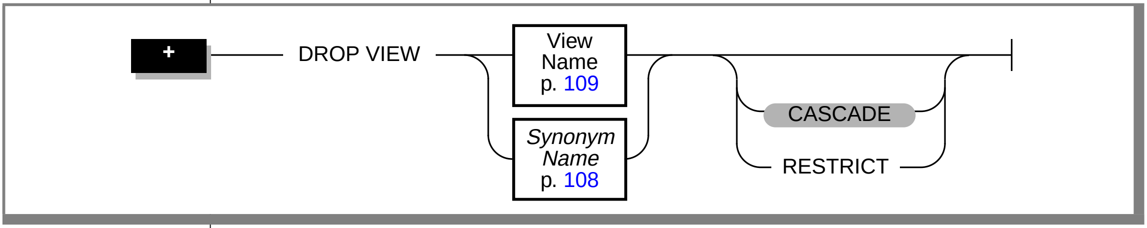
DROP TABLE



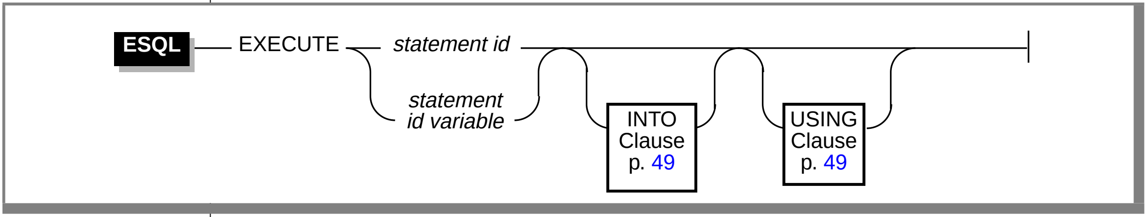
DROP TRIGGER



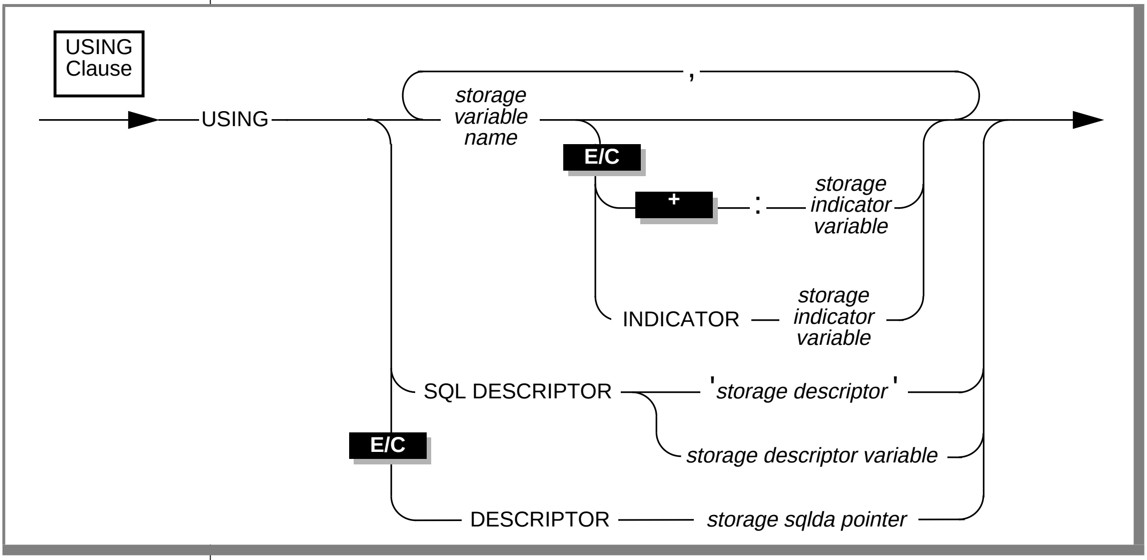
DROP VIEW



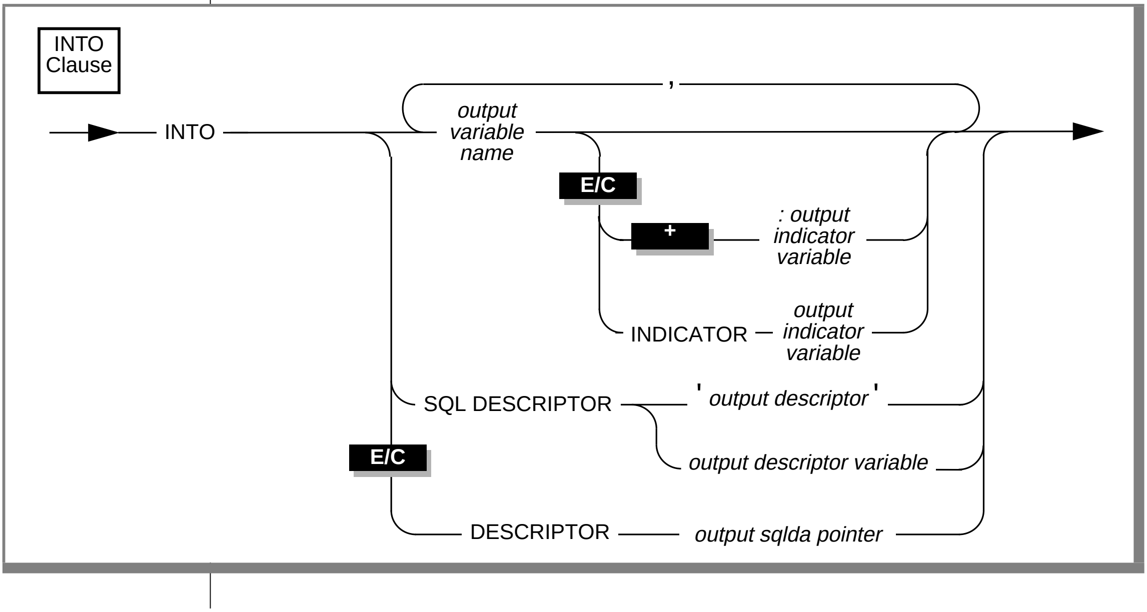
EXECUTE



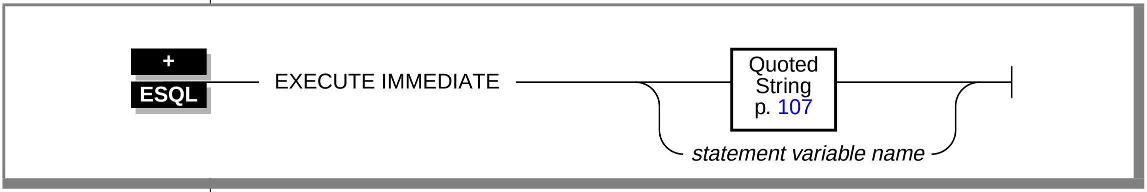
USING Clause



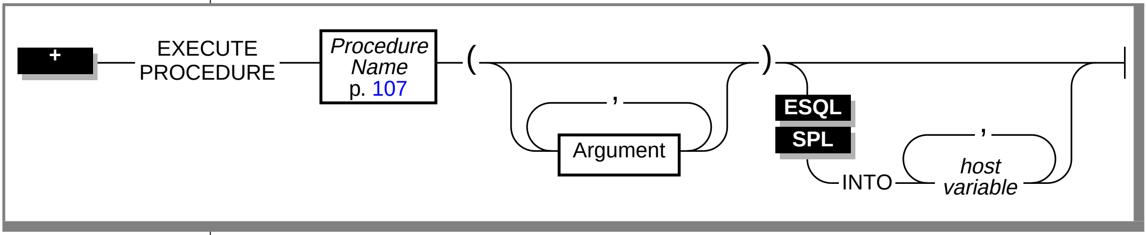
INTO Clause



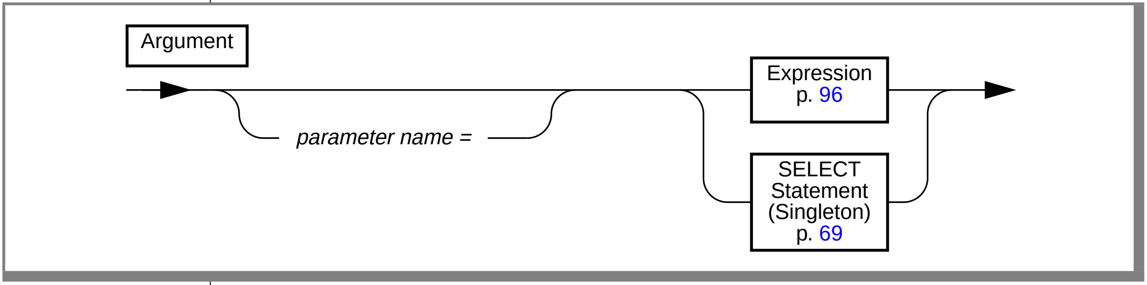
EXECUTE IMMEDIATE



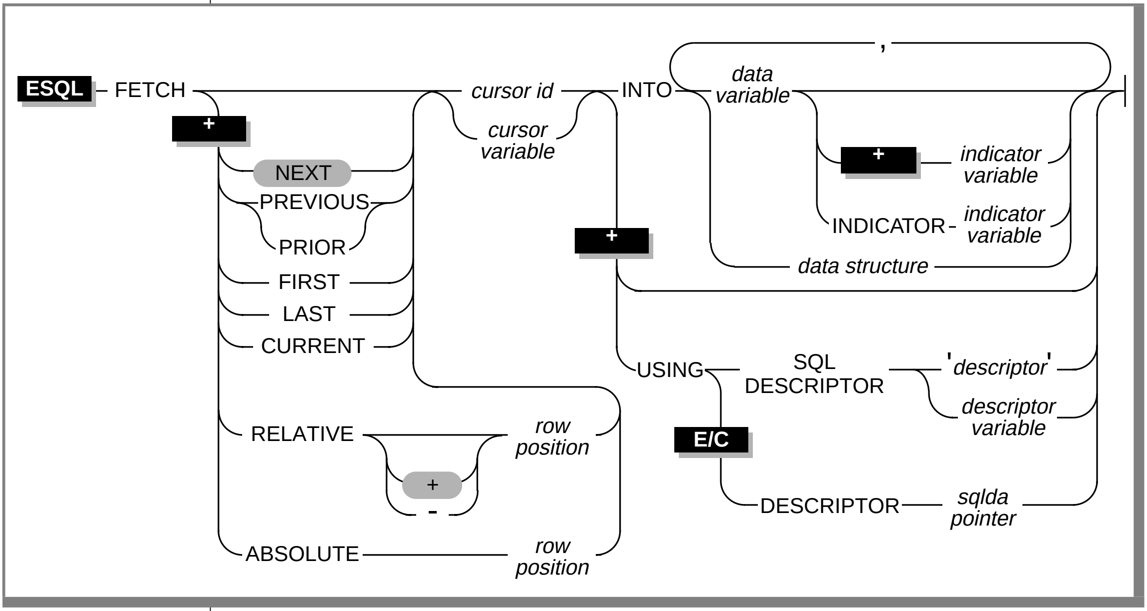
EXECUTE PROCEDURE



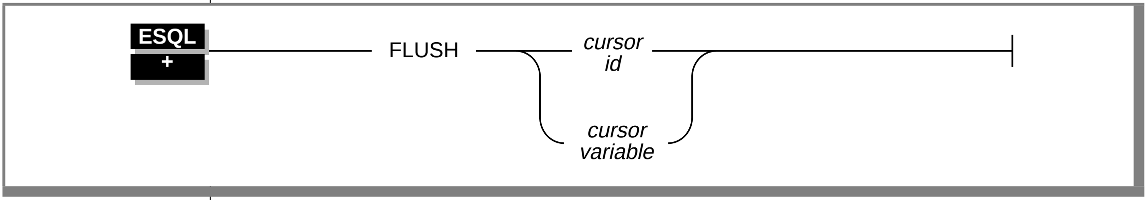
Argument



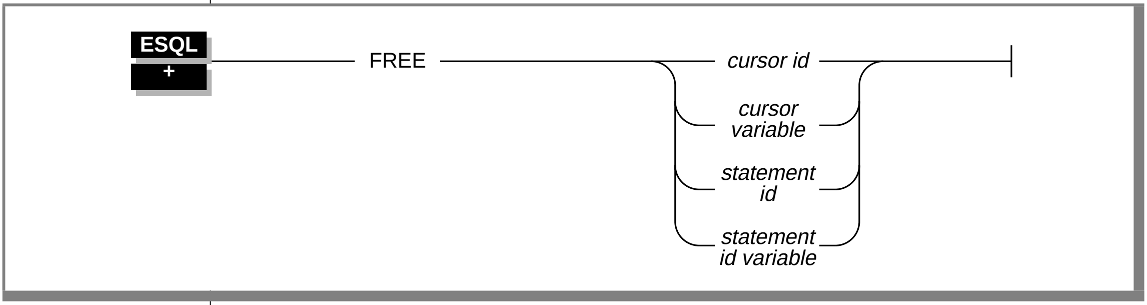
FETCH


ESQL
FETCH
+
NEXT
PREVIOUS
PRIOR
FIRST
LAST
CURRENT
RELATIVE
+
-
row position
ABSOLUTE
row position
cursor id
cursor variable
INTO
+
,
data variable
+
indicator variable
INDICATOR
indicator variable
data structure
USING
SQL DESCRIPTOR
'descriptor'
descriptor variable
E/C
DESCRIPTOR
sqlda pointer

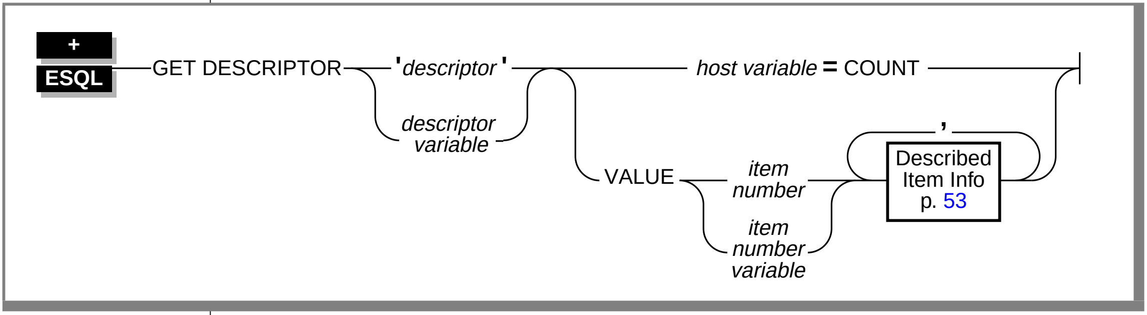
FLUSH


ESQL
+
FLUSH
cursor id
cursor variable

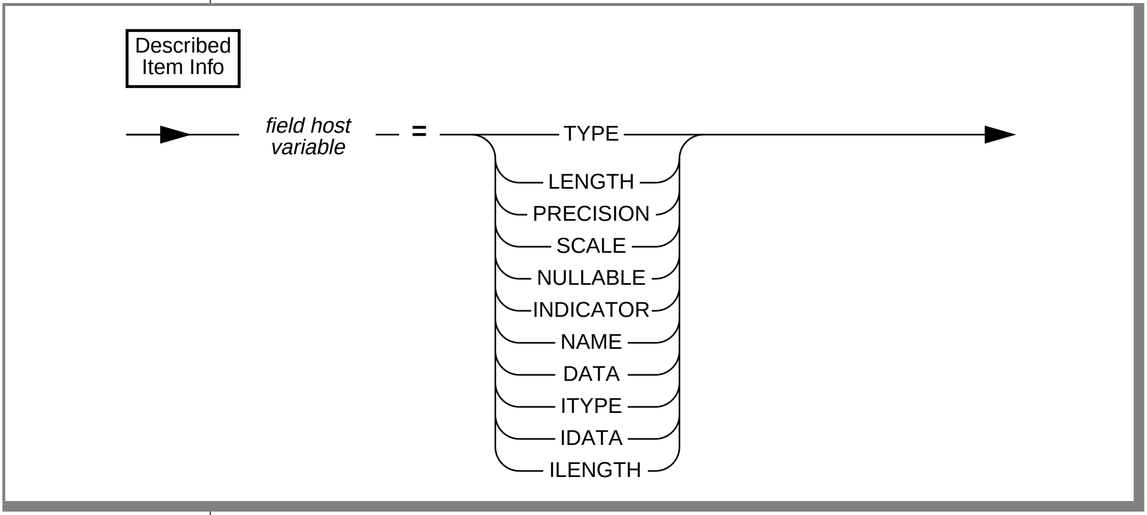
FREE



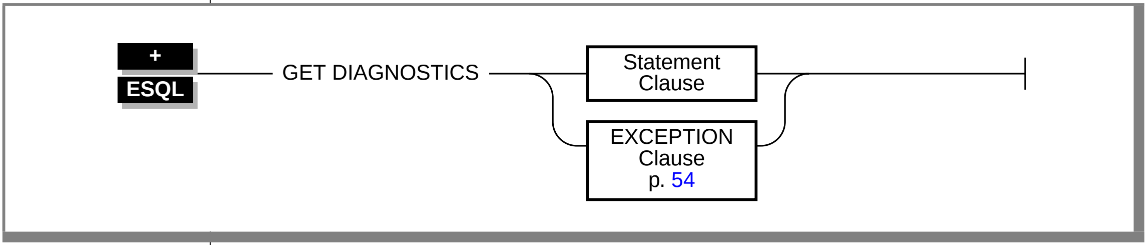
GET DESCRIPTOR



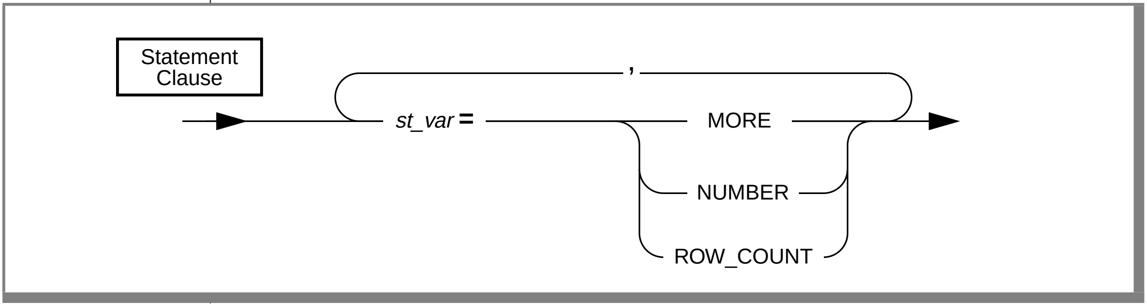
Described Item Info



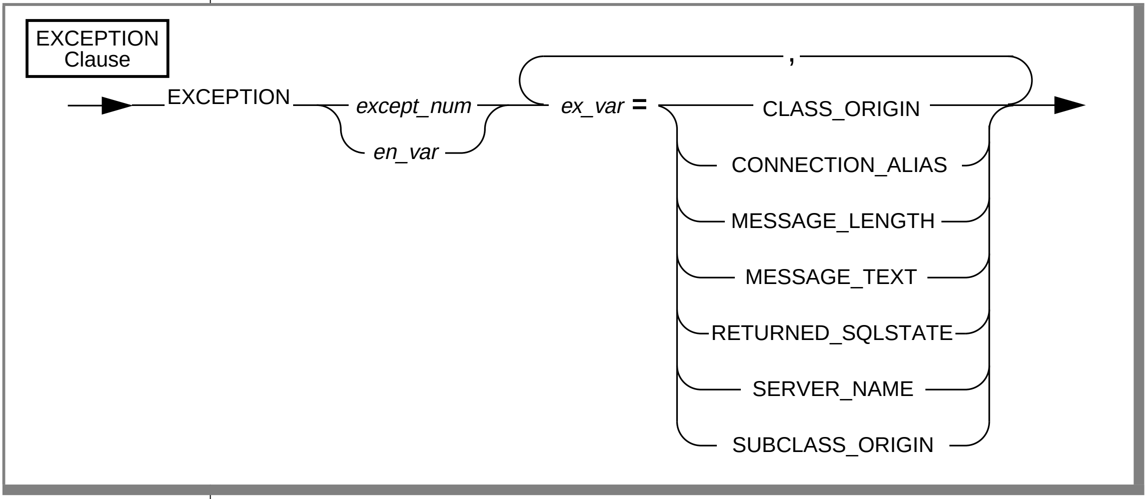
GET DIAGNOSTICS



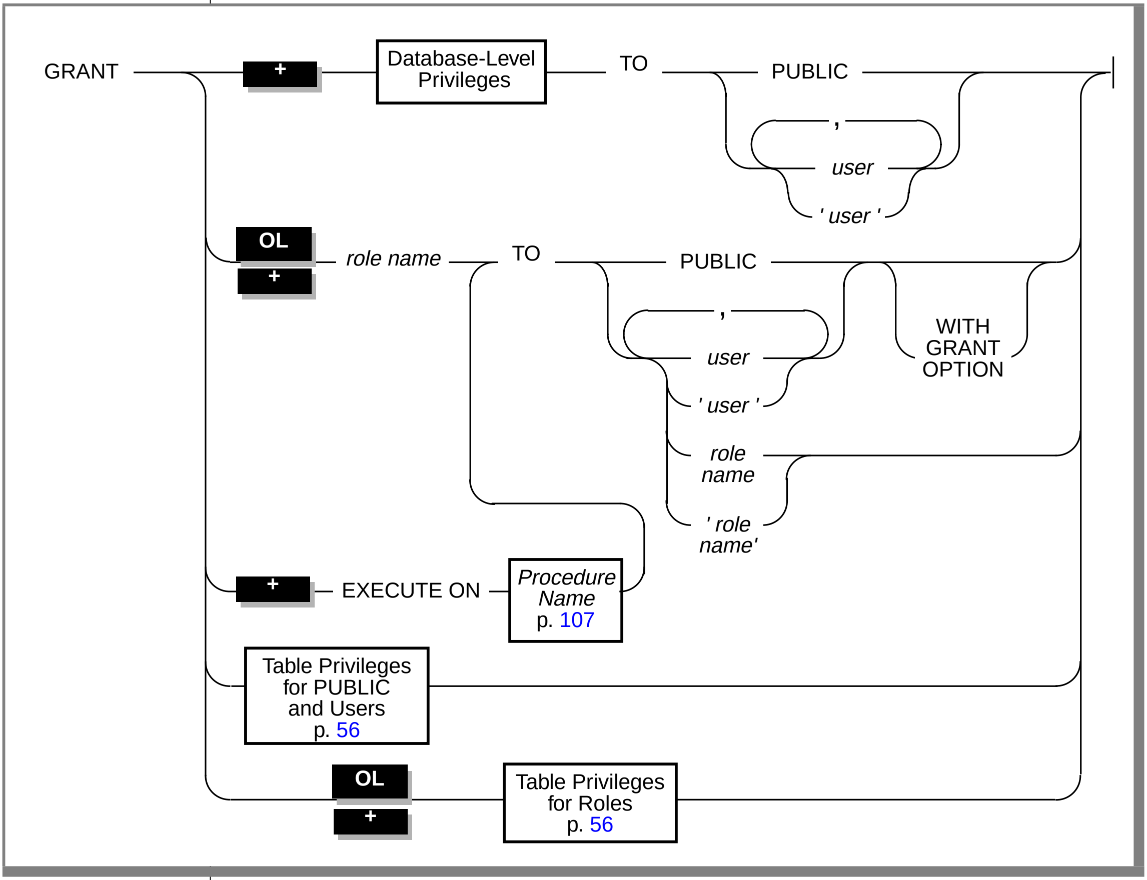
Statement Clause



EXCEPTION Clause



GRANT



Database-Level Privileges

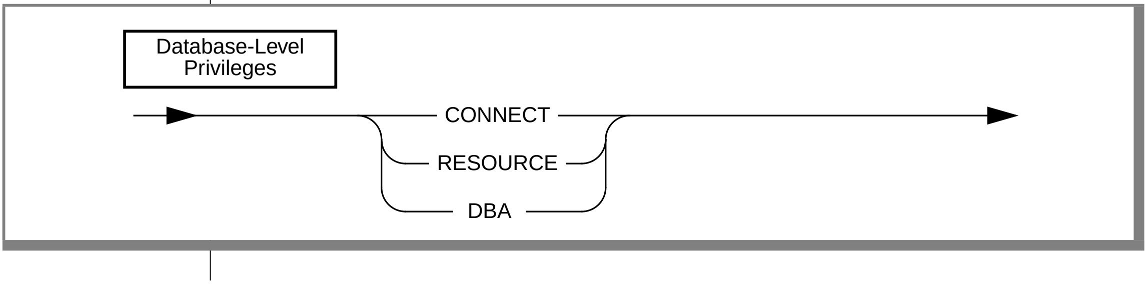


Table Privileges for PUBLIC and Users

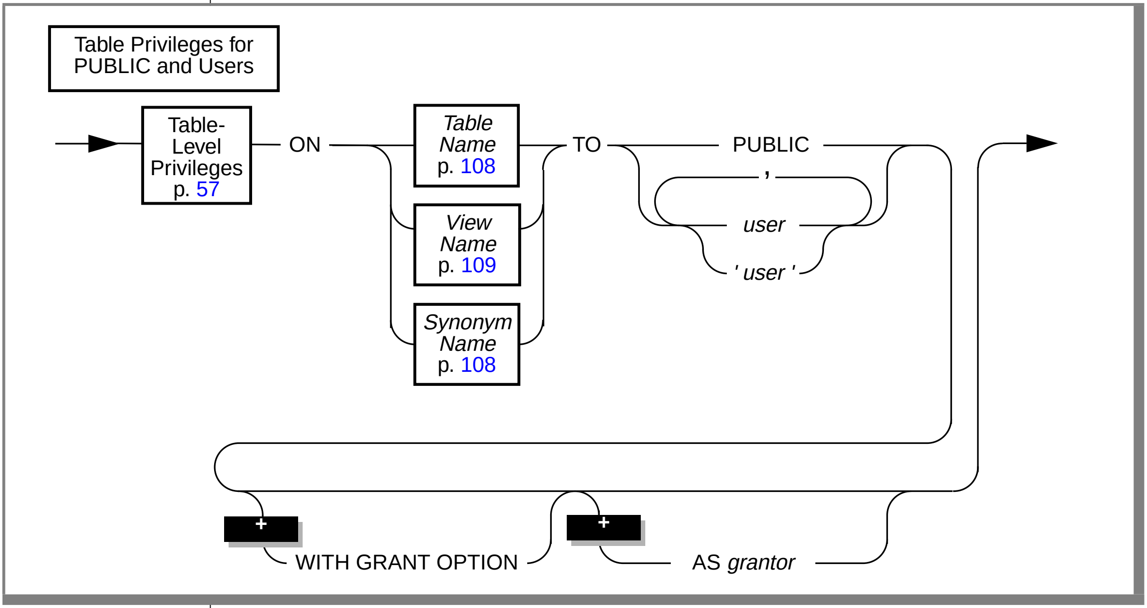


Table Privileges for Roles

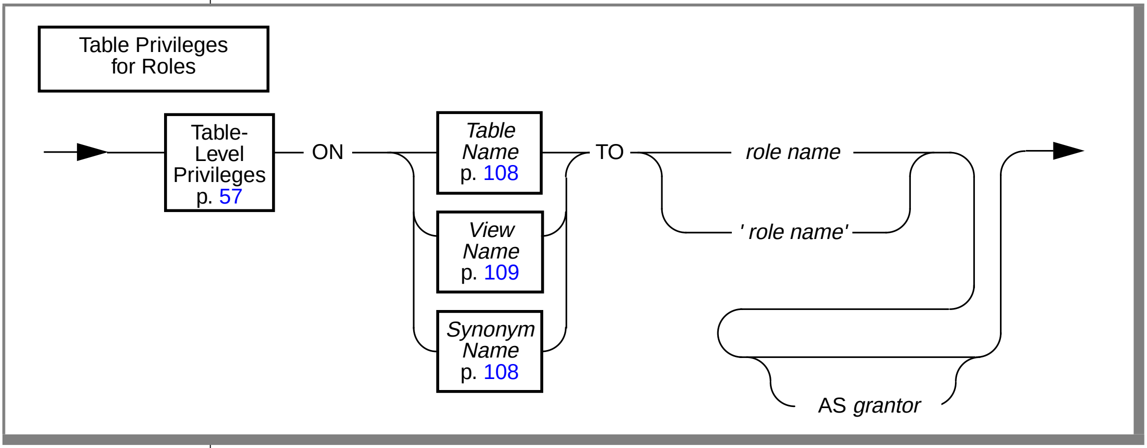
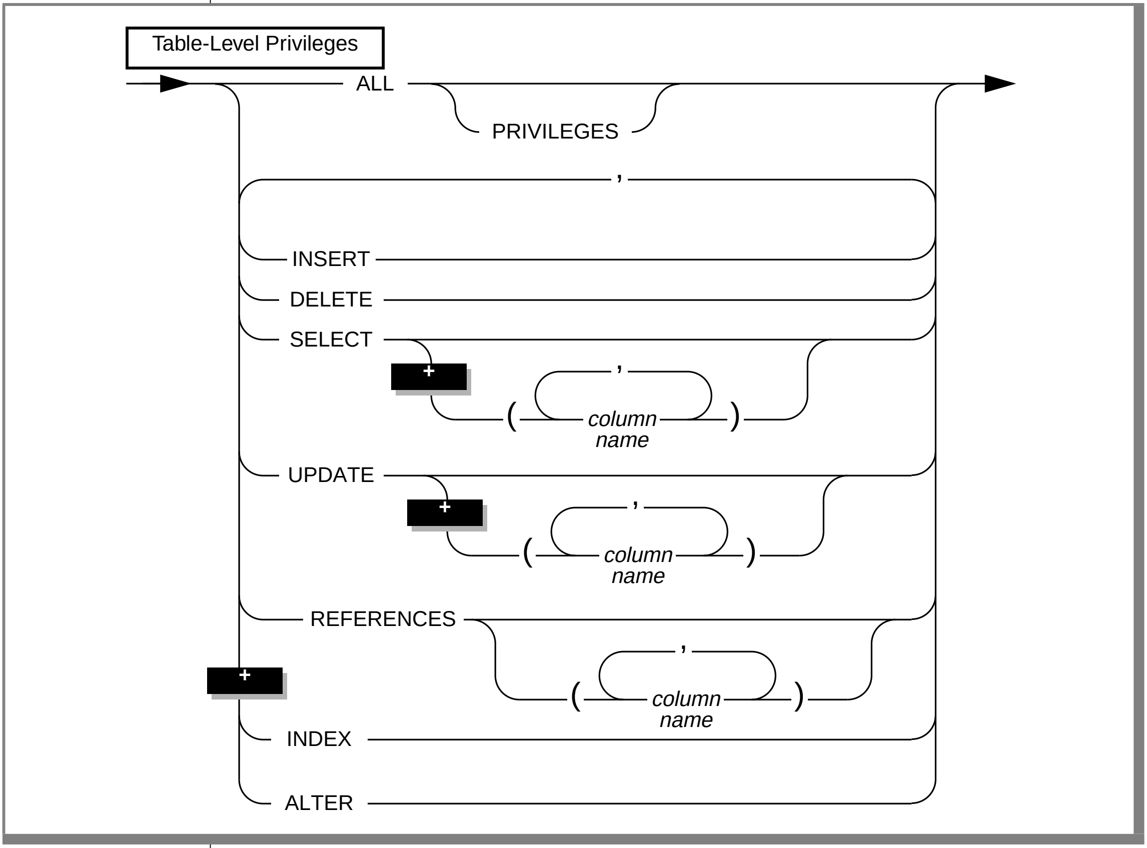
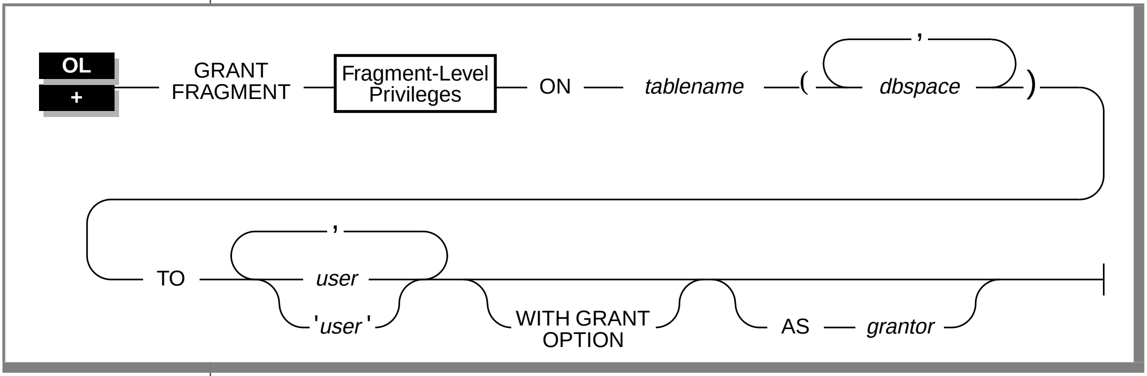


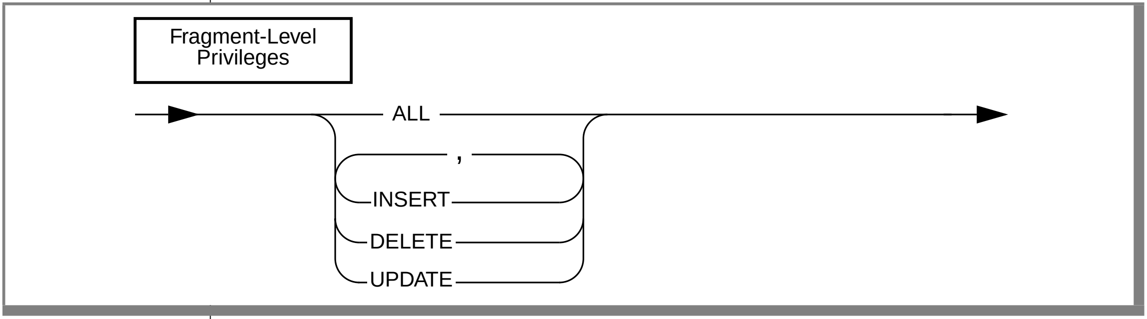
Table-Level Privileges



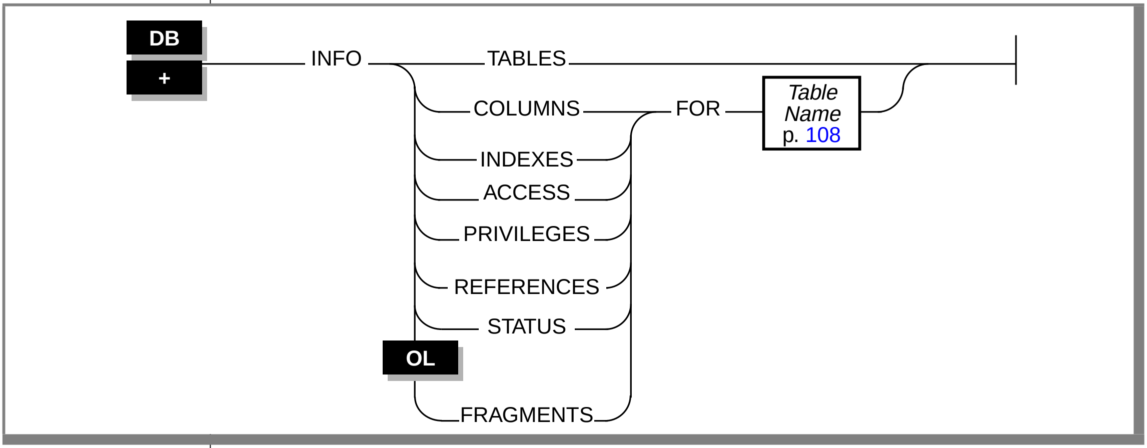
GRANT FRAGMENT



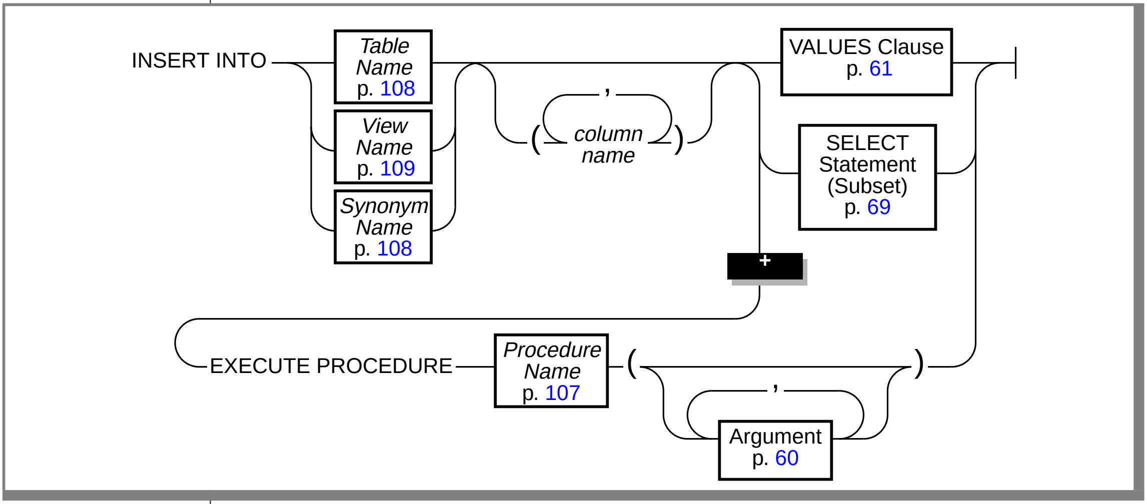
Fragment-Level Privileges



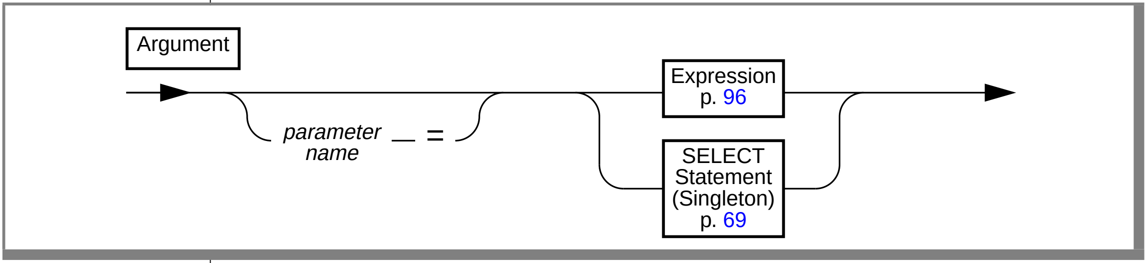
INFO


DB
+
INFO
TABLES
COLUMNS
FOR
Table Name p. 108
INDEXES
ACCESS
PRIVILEGES
REFERENCES
STATUS
OL
FRAGMENTS

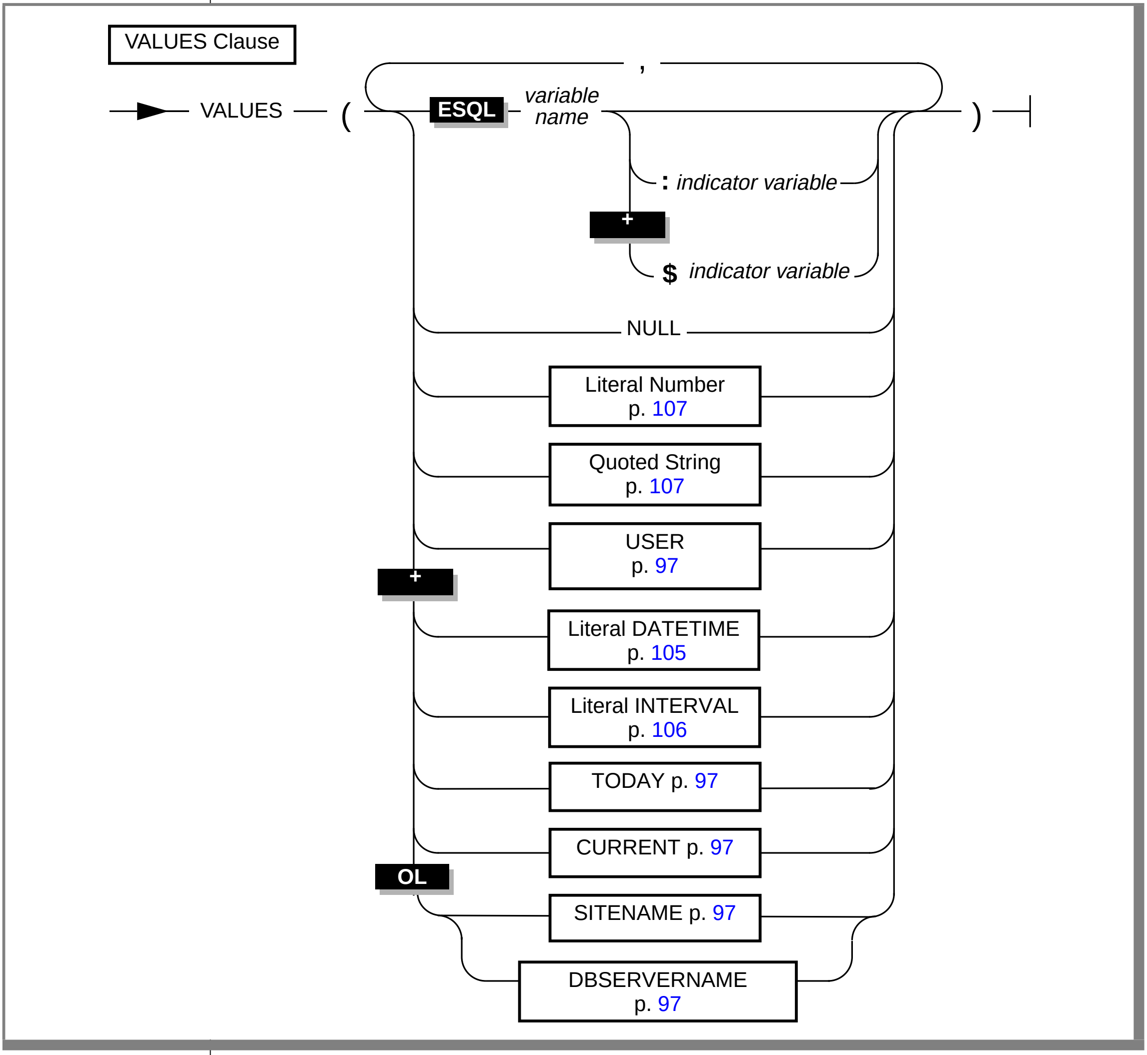
INSERT


INSERT INTO
Table Name p. 108
View Name p. 109
Synonym Name p. 108
(
,
column name
)
VALUES Clause p. 61
SELECT Statement (Subset) p. 69
+
EXECUTE PROCEDURE
Procedure Name p. 107
(
,
Argument p. 60
)

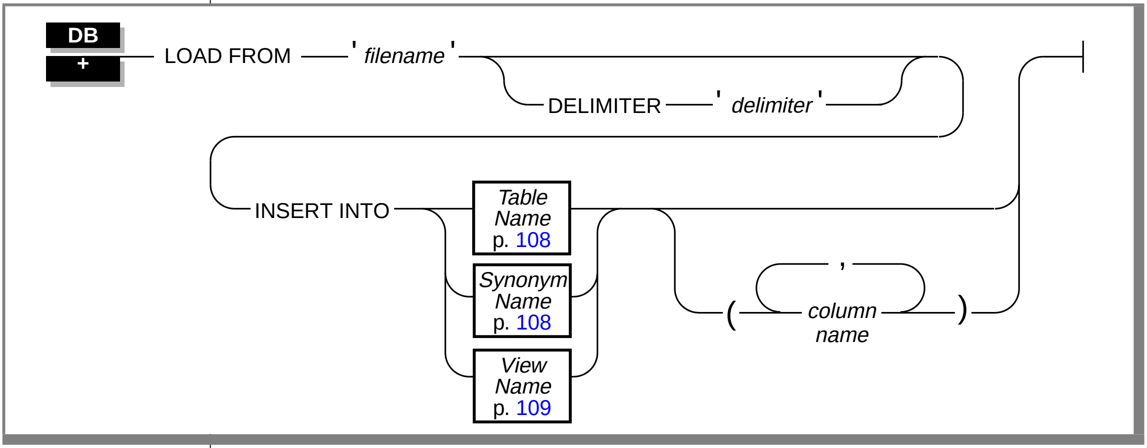
Argument


Argument
parameter name
=
Expression p. 96
SELECT Statement (Singleton) p. 69

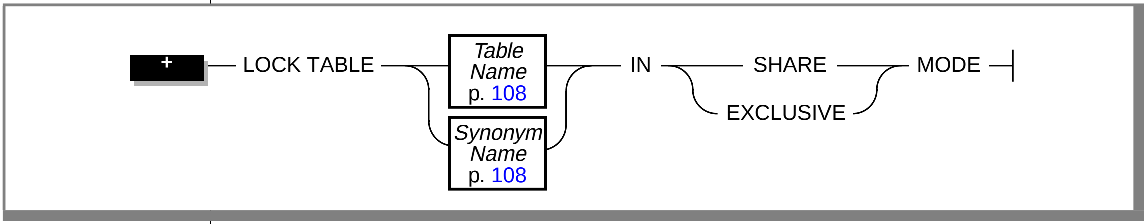
VALUES Clause



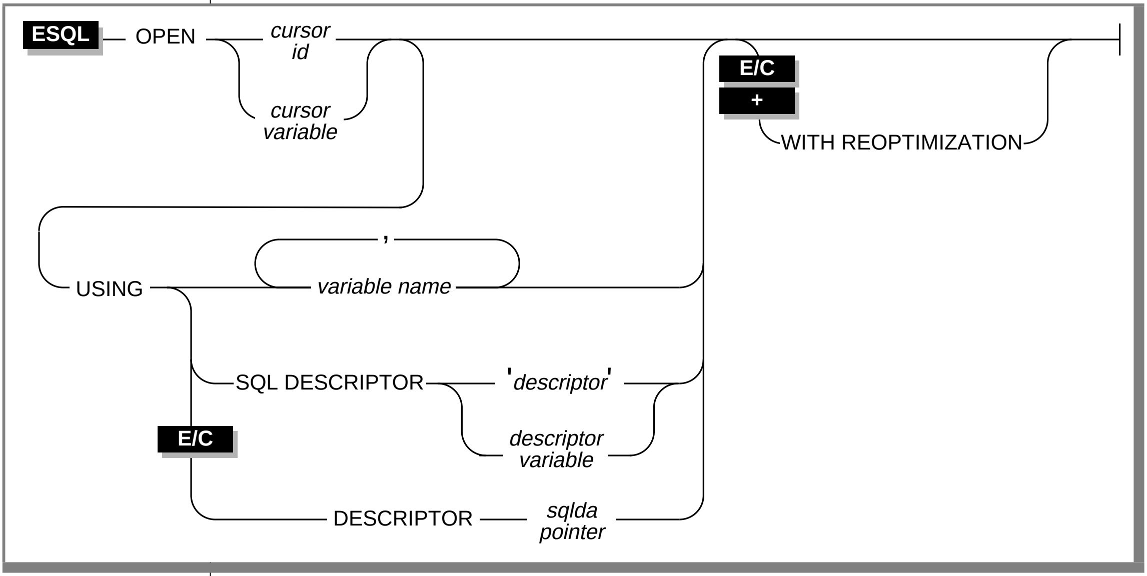
LOAD



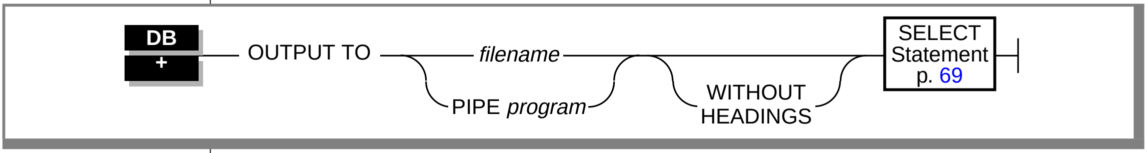
LOCK TABLE



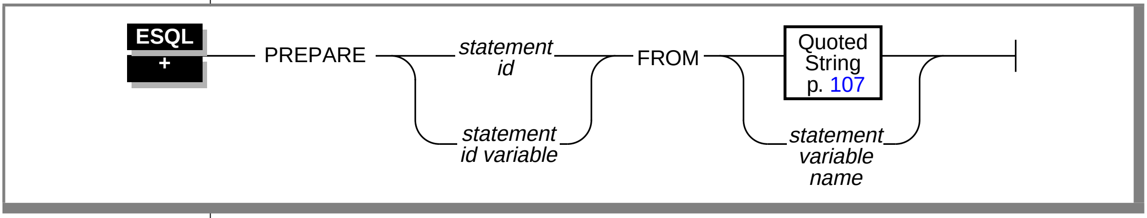
OPEN



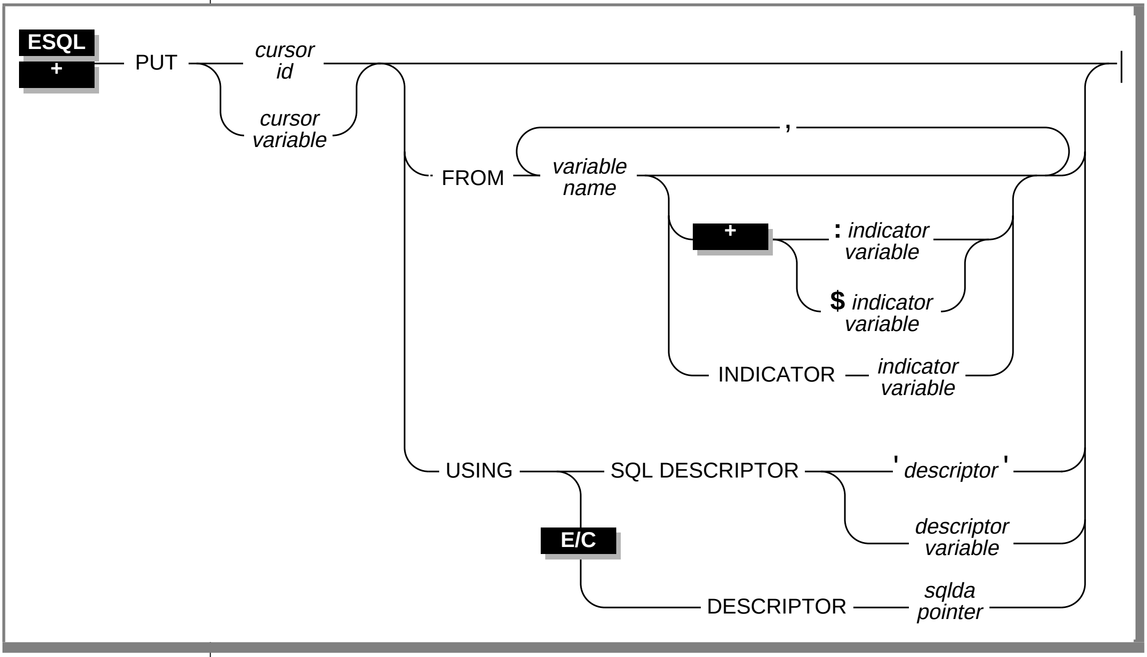
OUTPUT



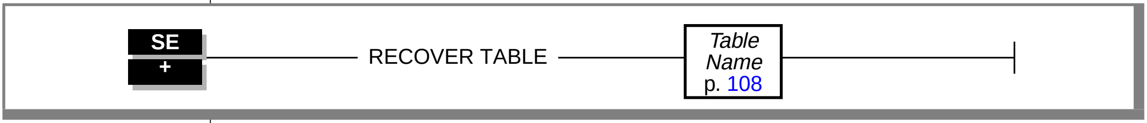
PREPARE



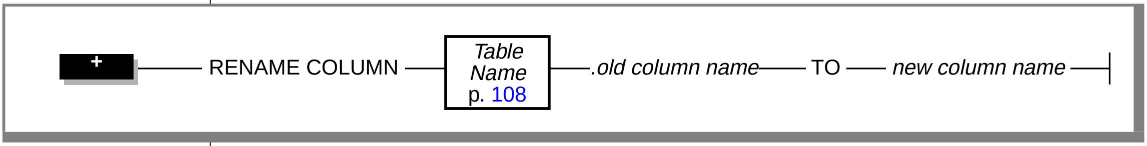
PUT



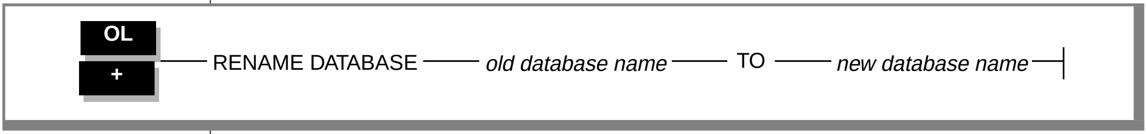
RECOVER TABLE



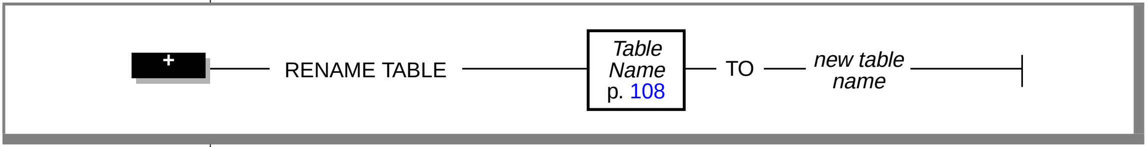
RENAME COLUMN



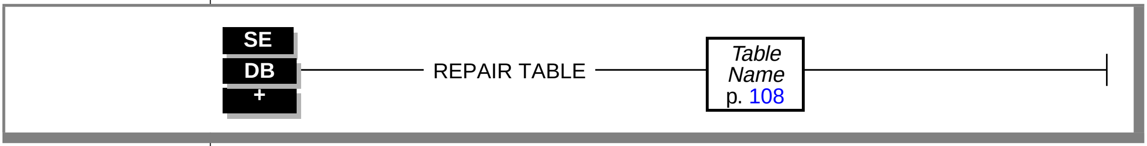
RENAME DATABASE



RENAME TABLE



REPAIR TABLE



REVOKE

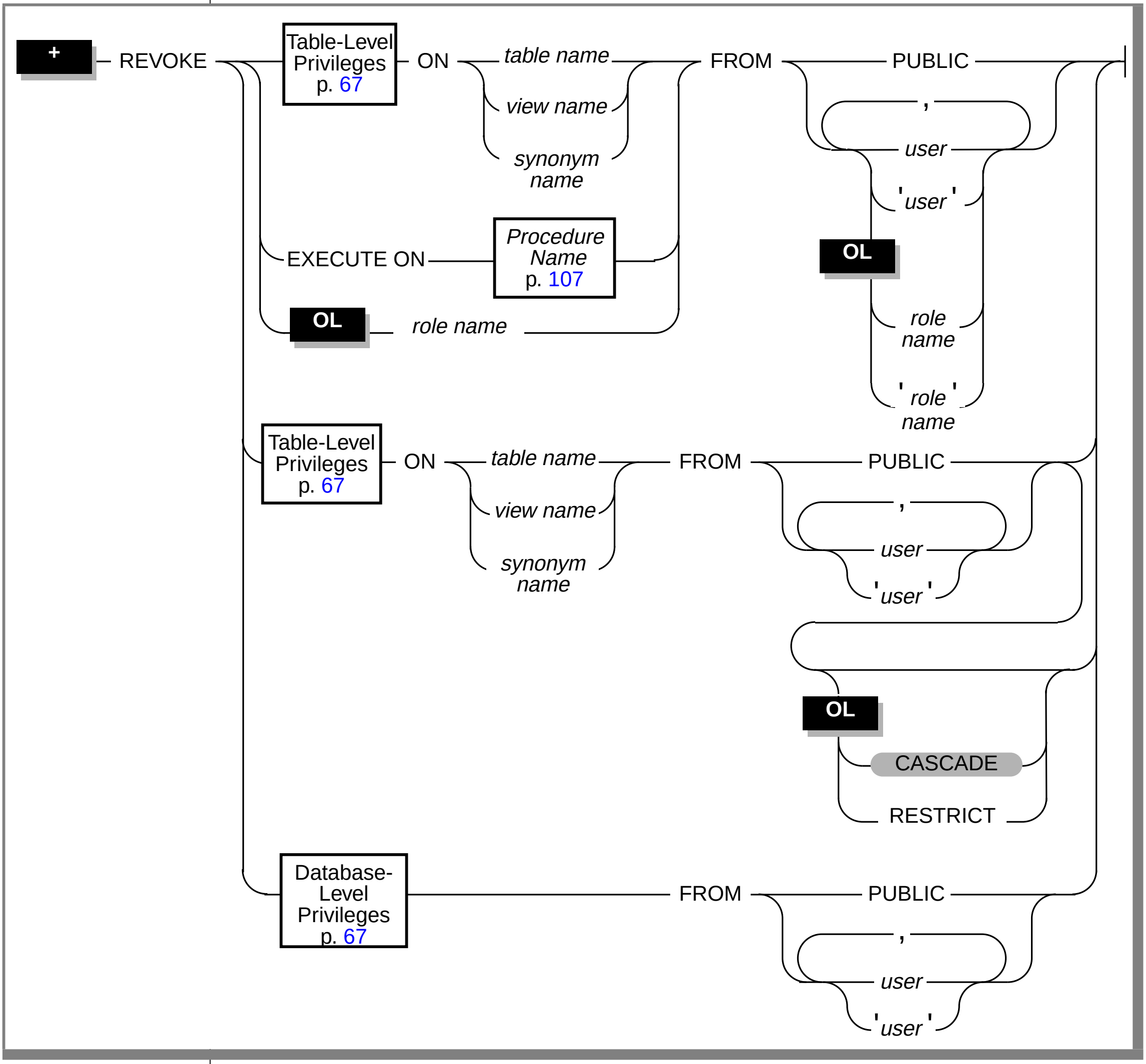
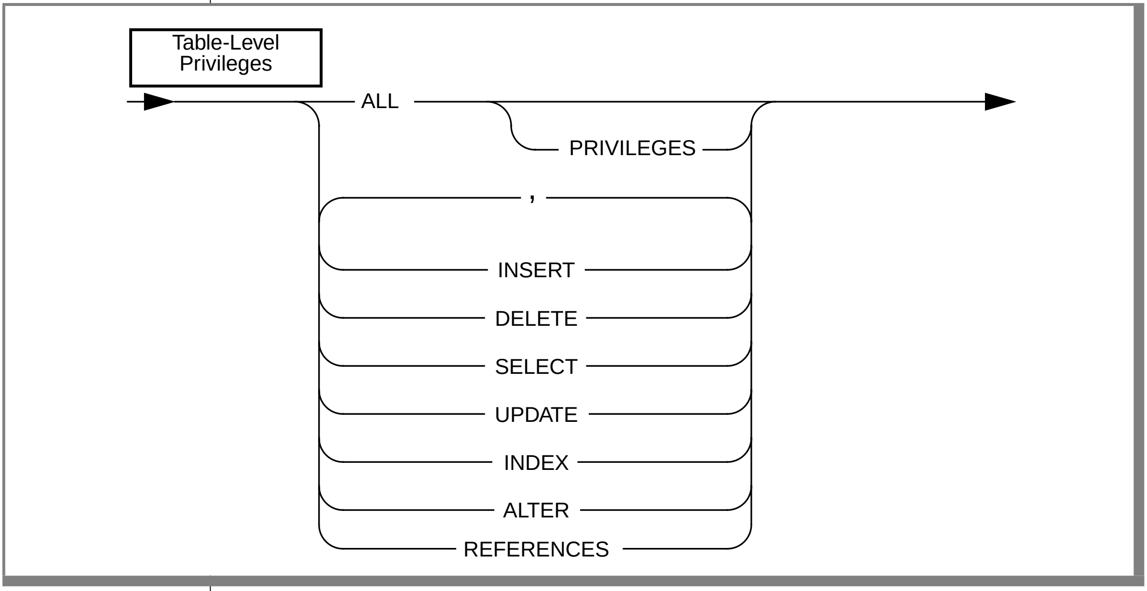
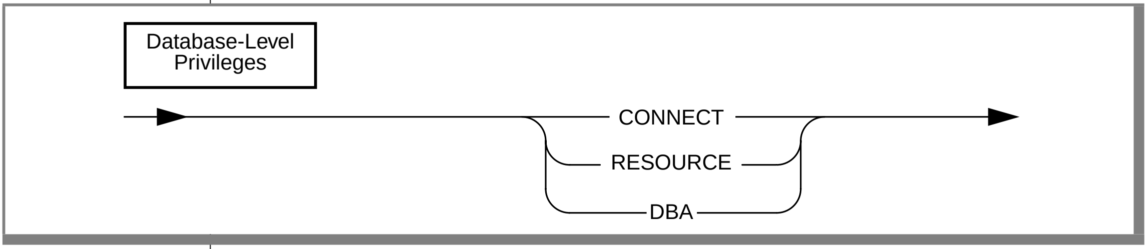

+
REVOKE
Table-Level Privileges p. 67
ON
table name
view name
synonym name
FROM
PUBLIC
,
user
'user'
OL
role name
'role' name
EXECUTE ON
Procedure Name p. 107
OL
role name
Table-Level Privileges p. 67
ON
table name
view name
synonym name
FROM
PUBLIC
,
user
'user'
OL
CASCADE
RESTRICT
Database-Level Privileges p. 67
FROM
PUBLIC
,
user
'user'

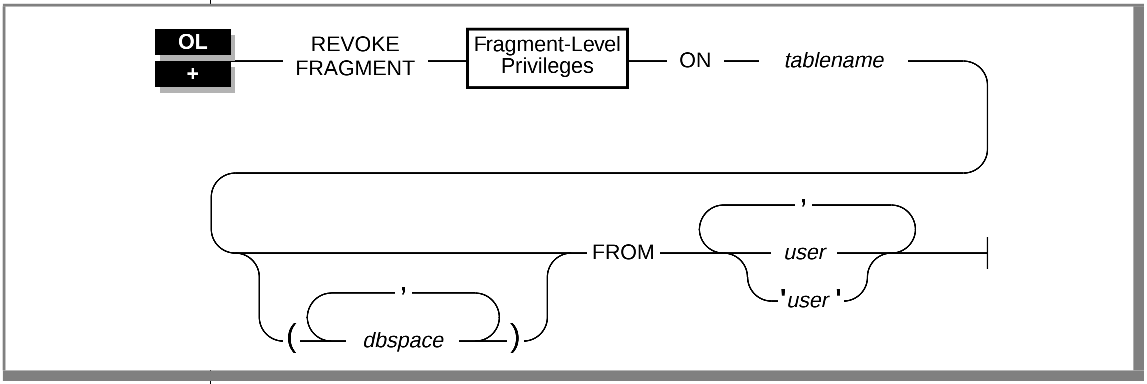
Table-Level Privileges



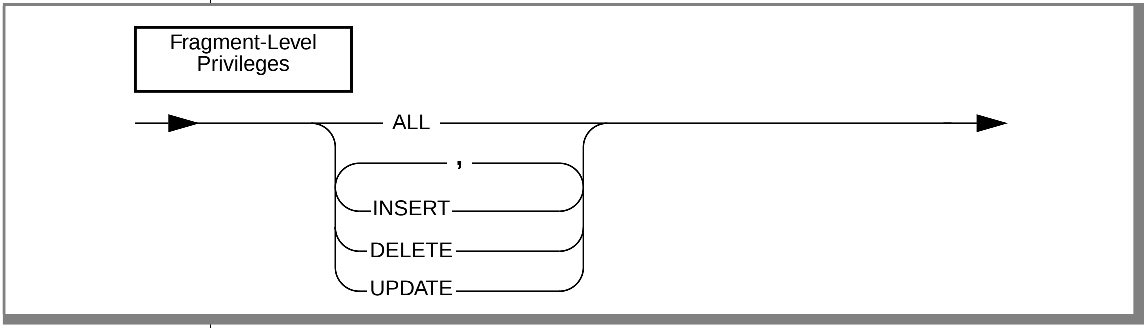
Database-Level Privileges



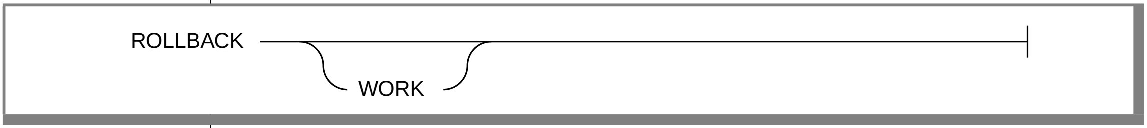
REVOKE FRAGMENT



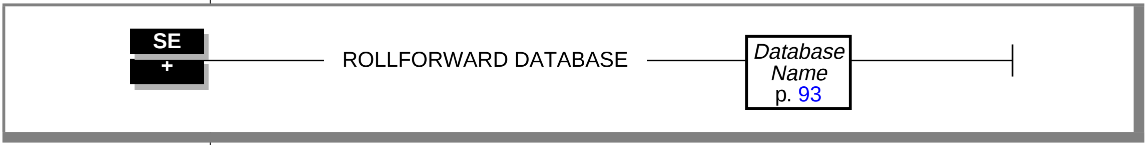
Fragment-Level Privileges



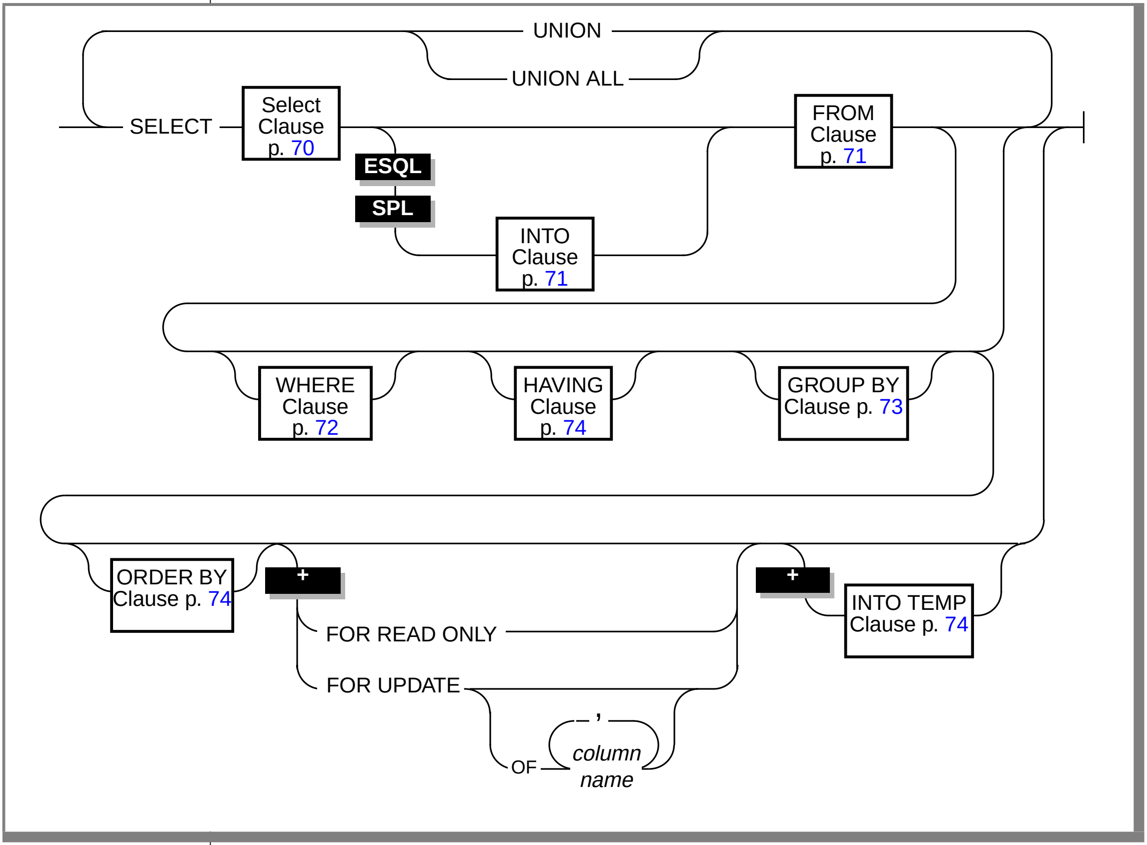
ROLLBACK WORK



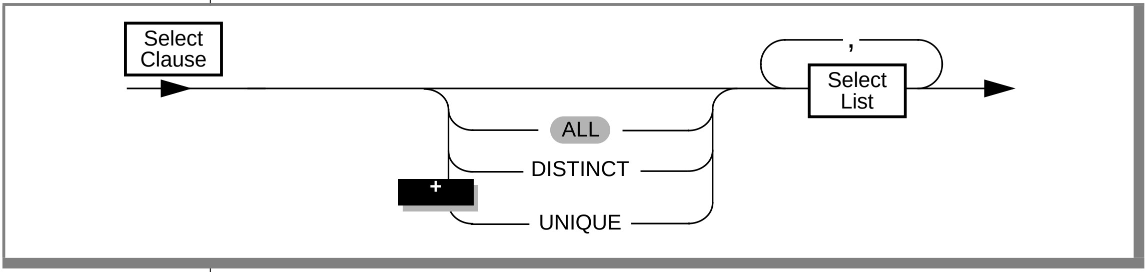
ROLLFORWARD DATABASE



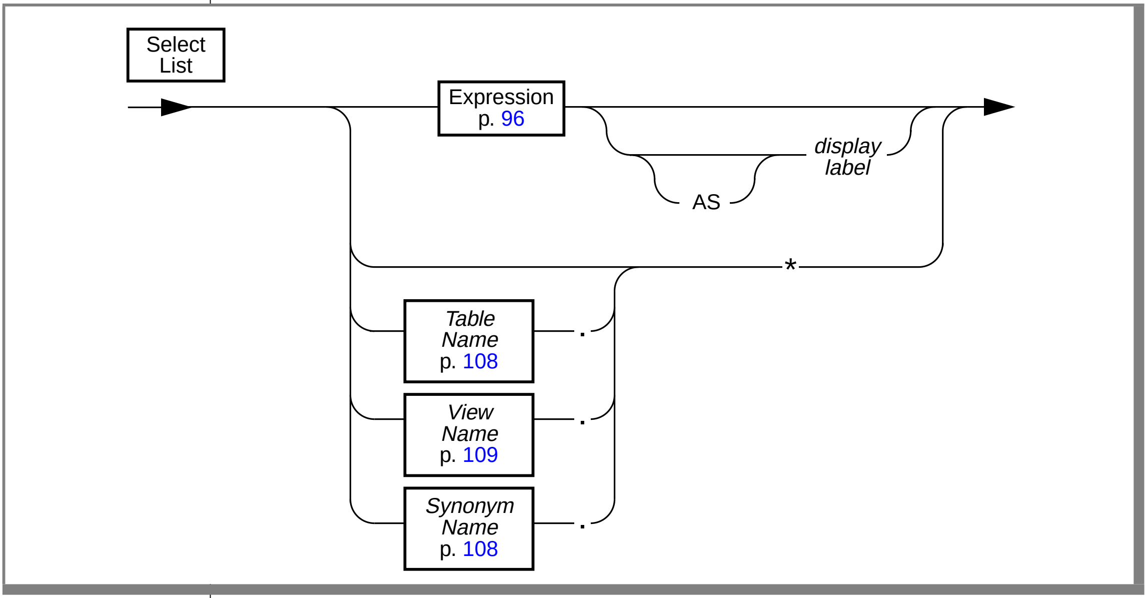
SELECT



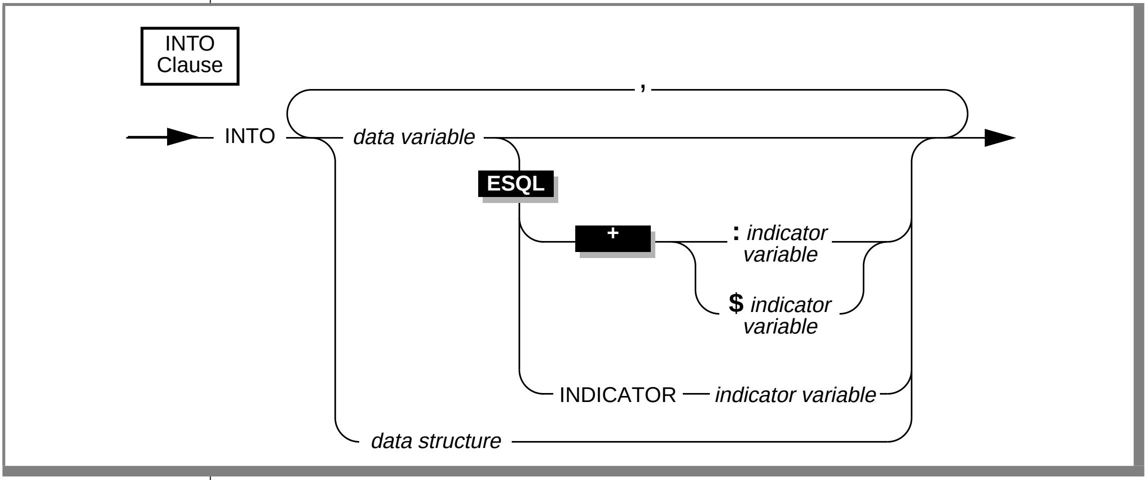
SELECT Clause



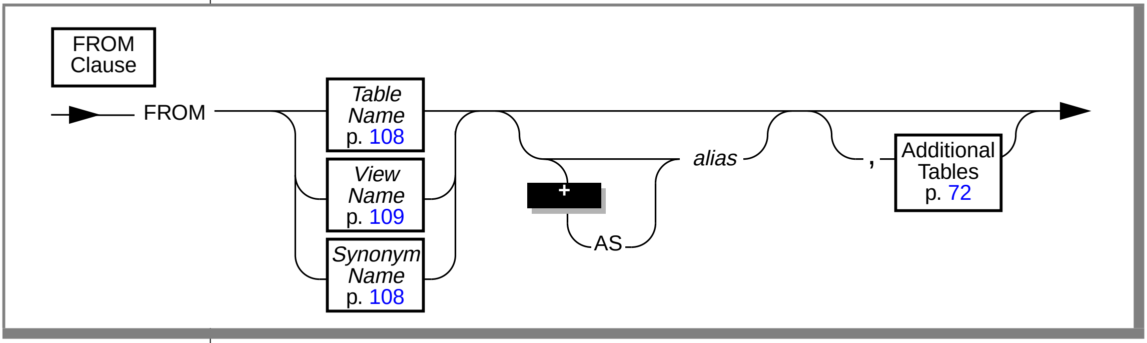
Select List



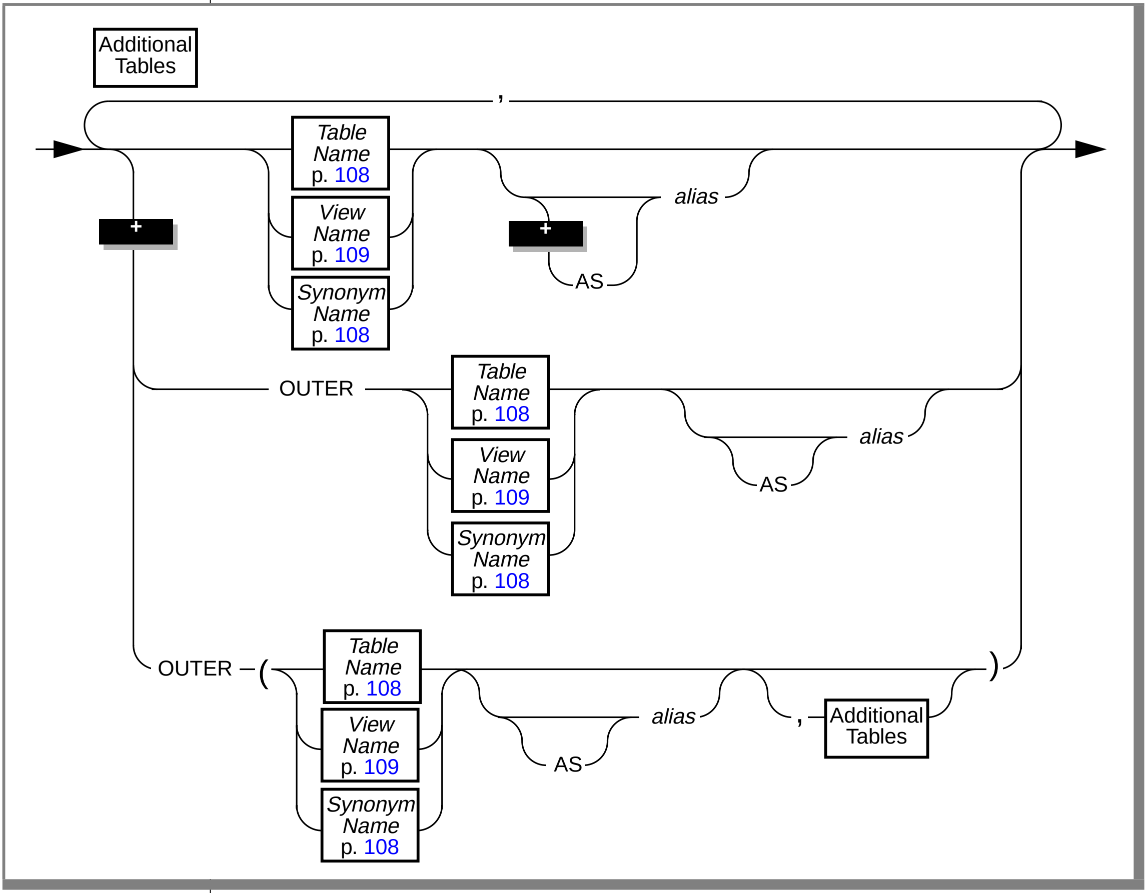
INTO Clause



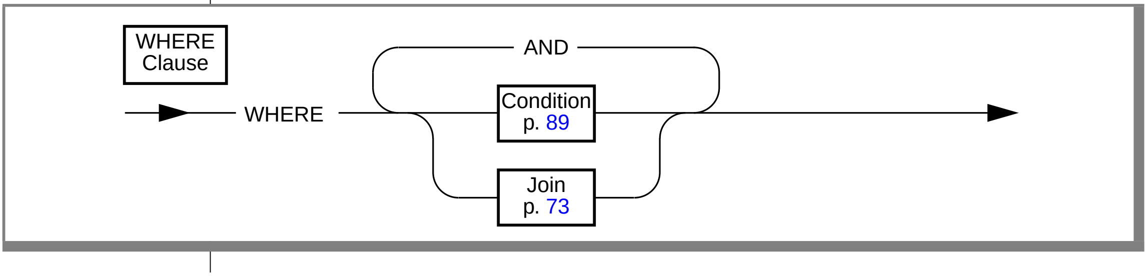
FROM Clause



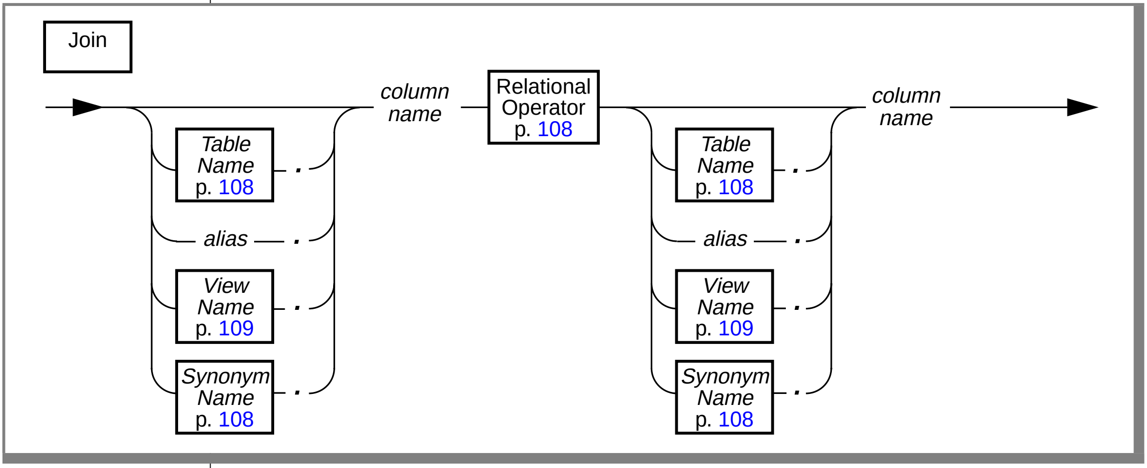
Additional Tables



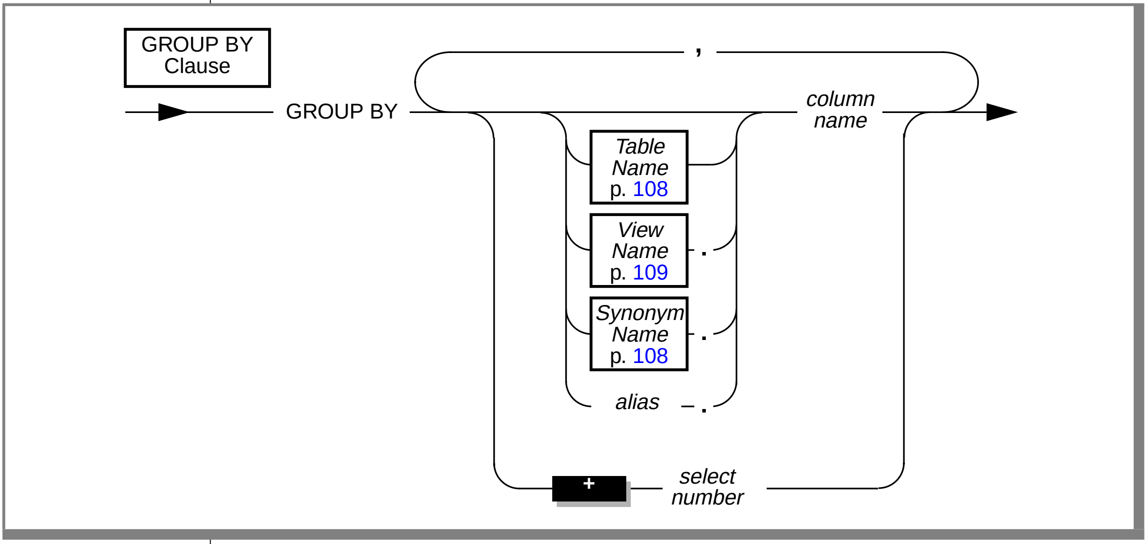
WHERE Clause



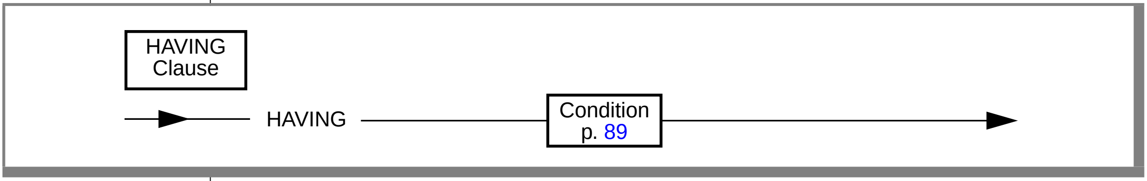
Join Option



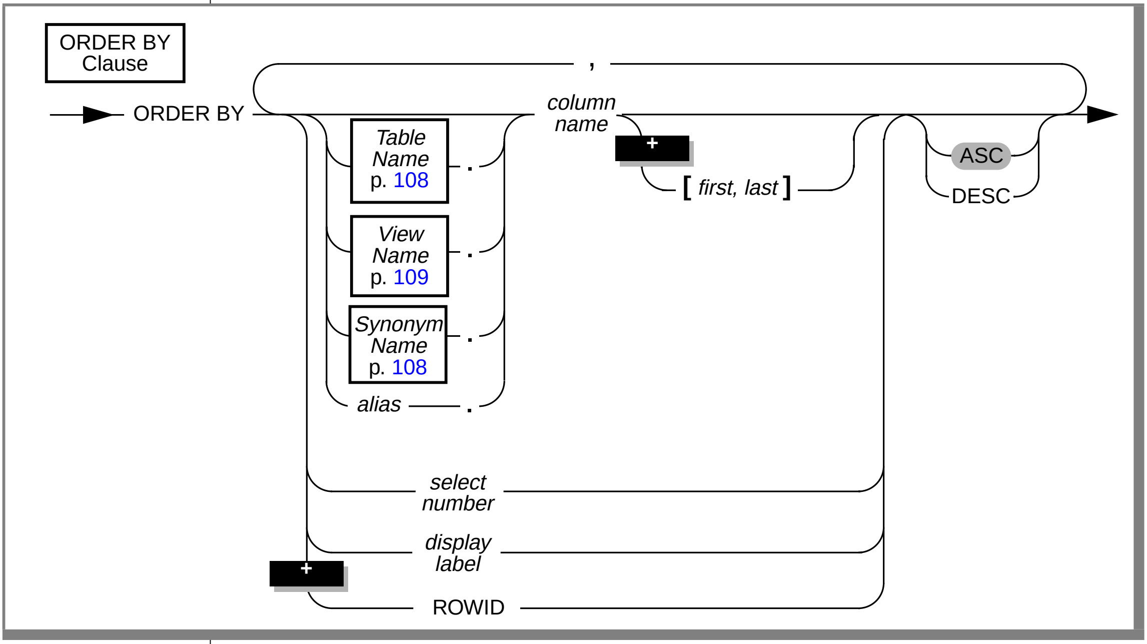
GROUP BY Clause



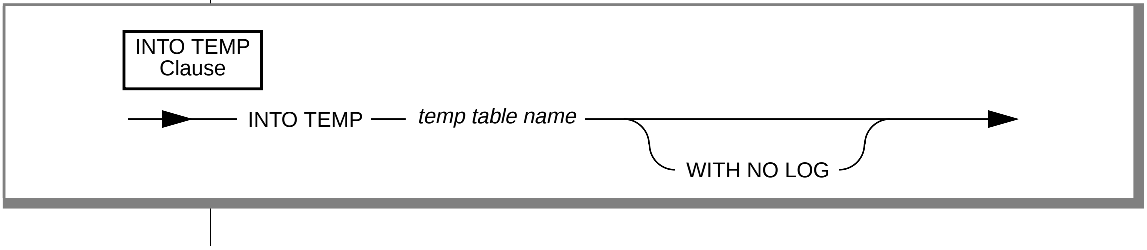
HAVING Clause



ORDER BY Clause



INTO TEMP Clause



SET

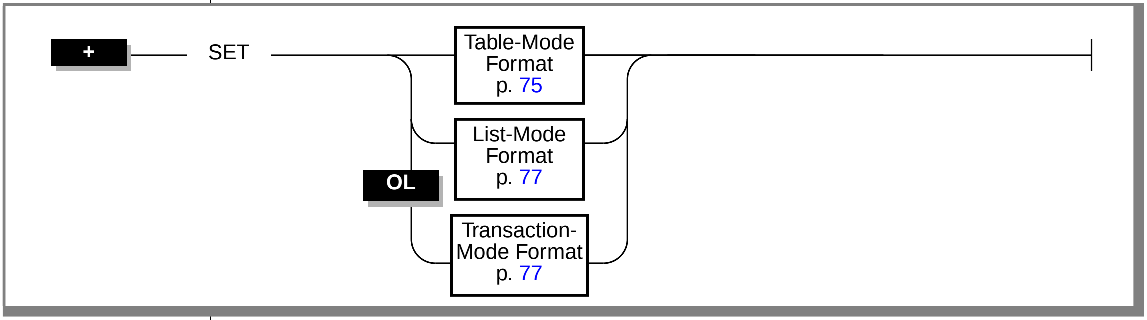


Table-Mode Format

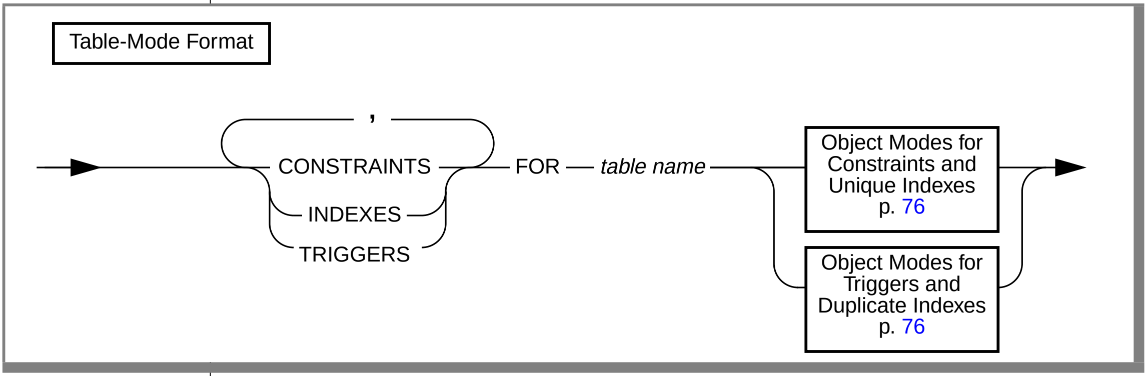
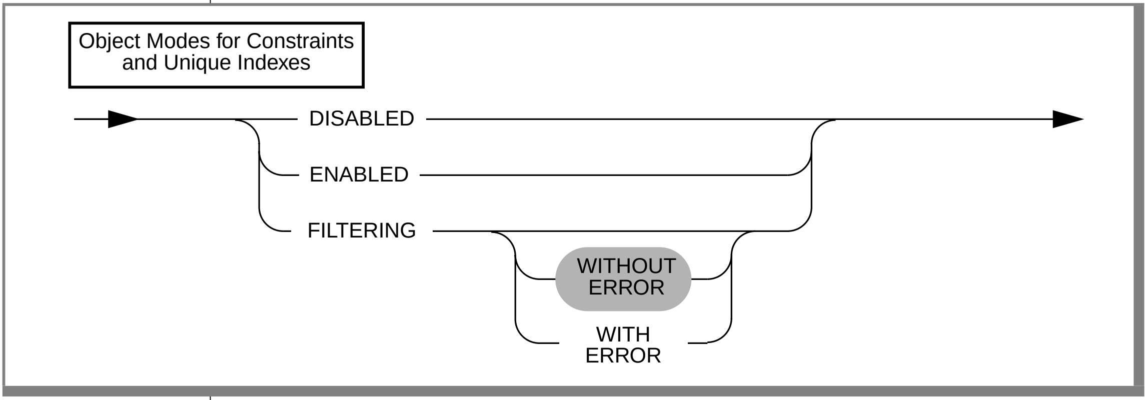
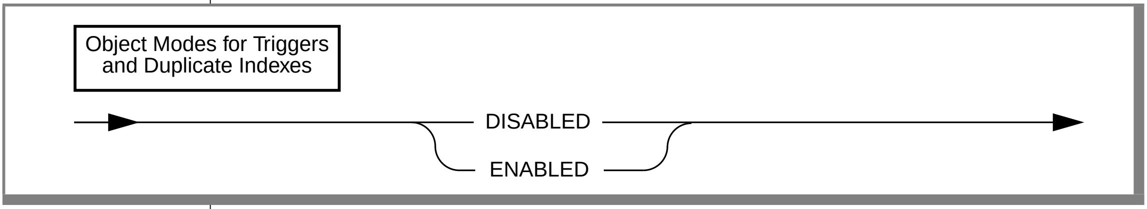


Table-Mode Format
,
CONSTRAINTS
INDEXES
TRIGGERS
FOR
table name
Object Modes for Constraints and Unique Indexes p. 76
Object Modes for Triggers and Duplicate Indexes p. 76

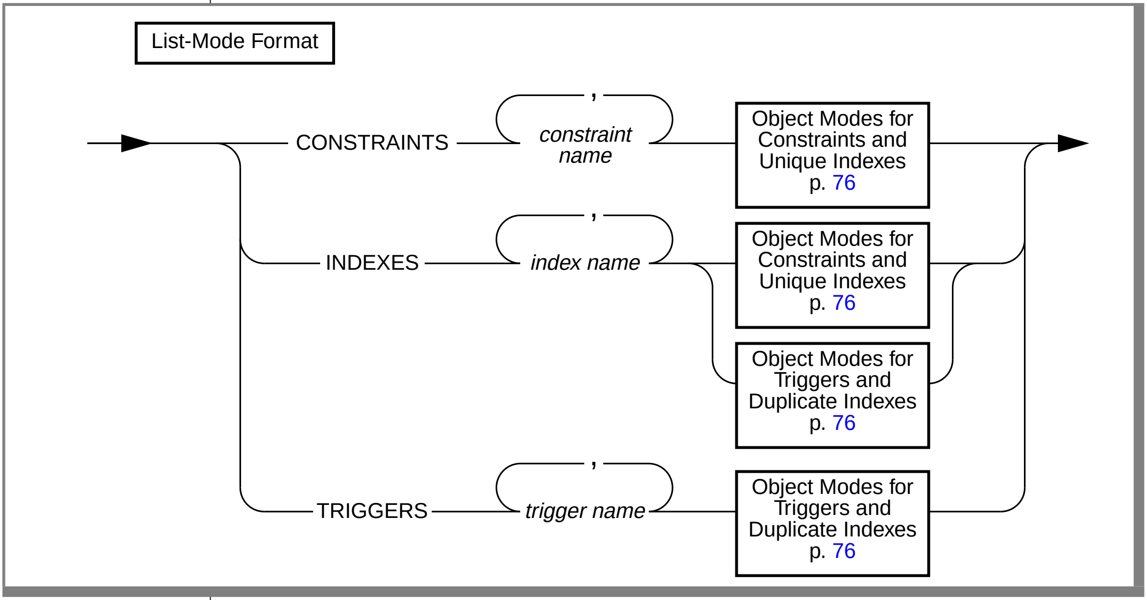
Object Modes for Constraints and Unique Indexes



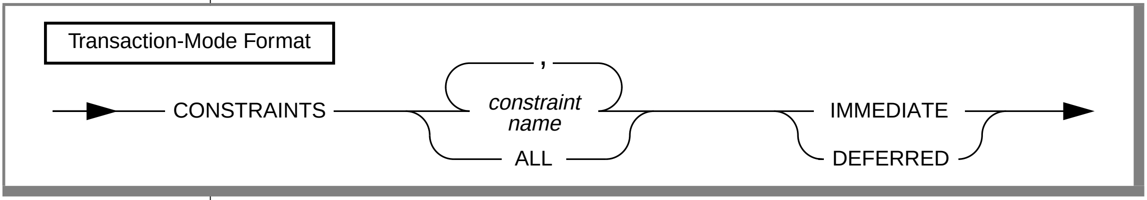
Object Modes for Triggers and Duplicate Indexes



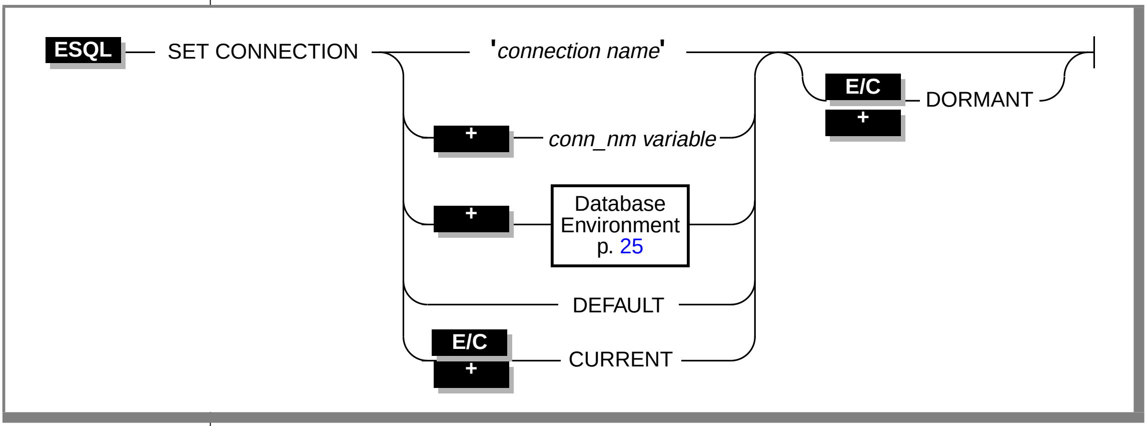
List-Mode Format


List-Mode Format
CONSTRAINTS
,
constraint name
INDEXES
,
index name
TRIGGERS
,
trigger name
Object Modes for Constraints and Unique Indexes p. 76
Object Modes for Constraints and Unique Indexes p. 76
Object Modes for Triggers and Duplicate Indexes p. 76
Object Modes for Triggers and Duplicate Indexes p. 76

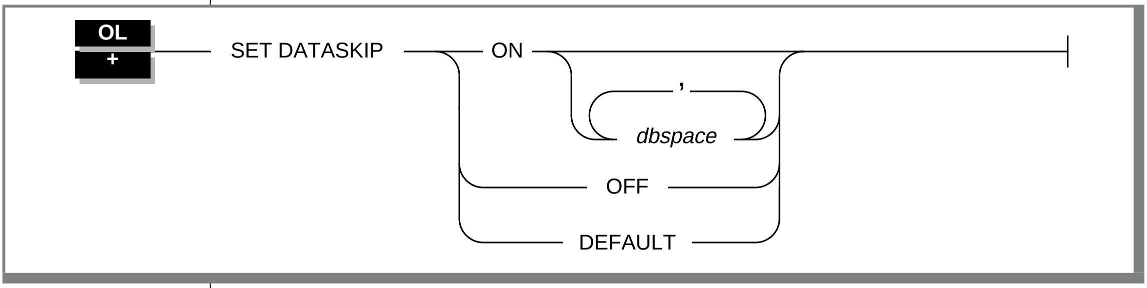
Transaction-Mode Format



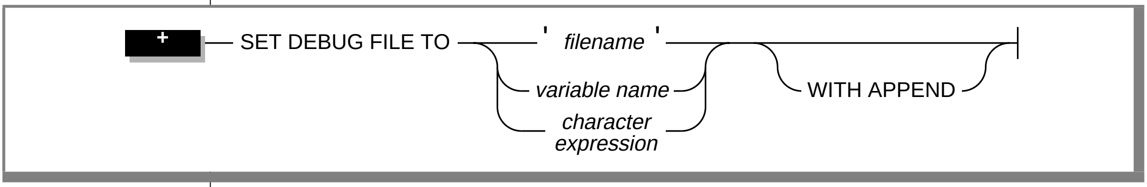
SET CONNECTION



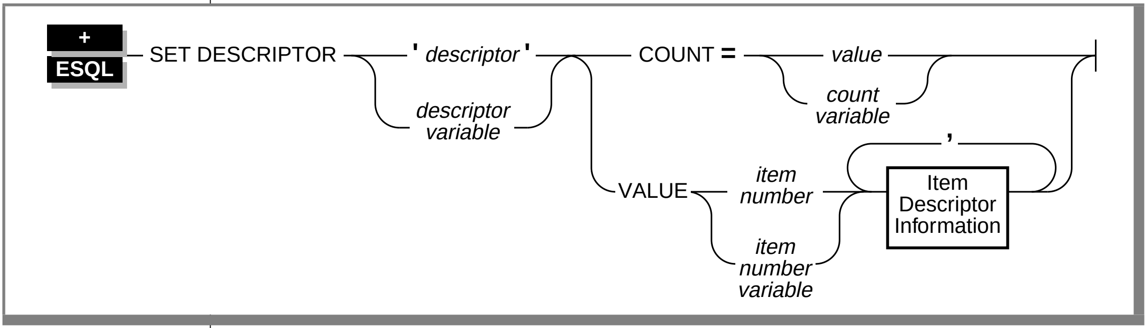
SET DATASKIP



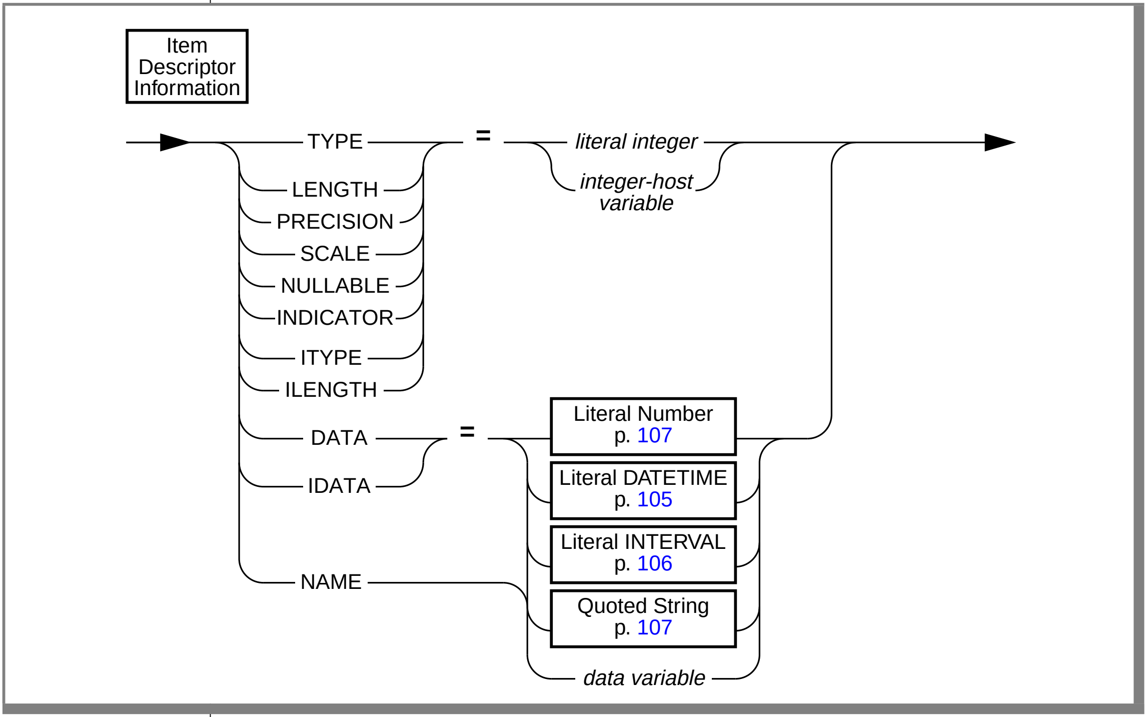
SET DEBUG FILE TO



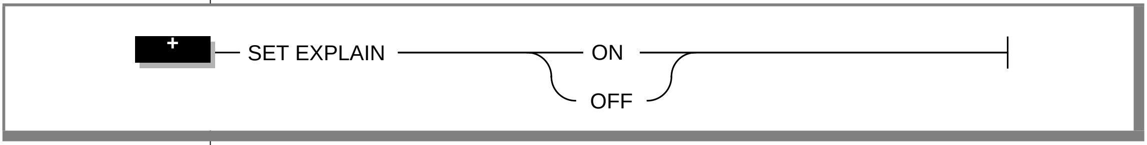
SET DESCRIPTOR



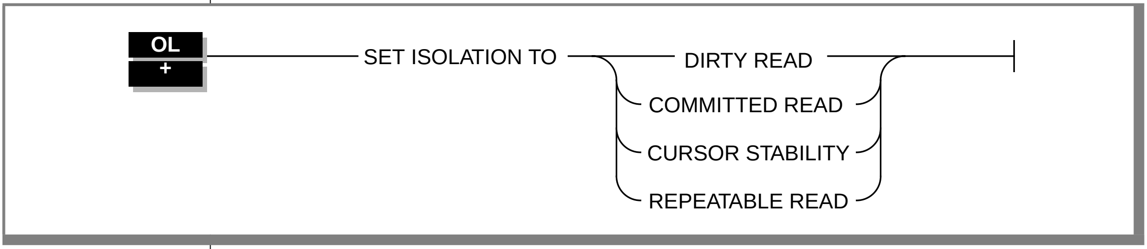
Item Descriptor Information



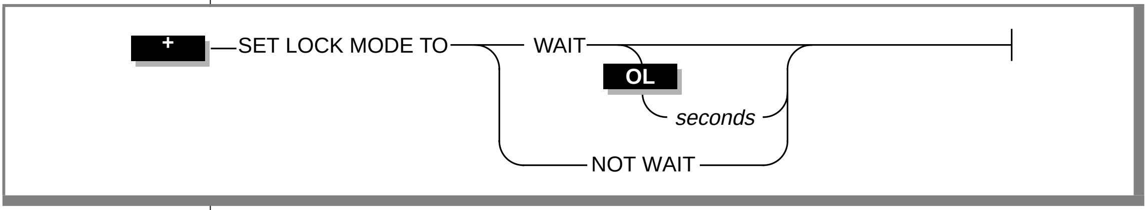
SET EXPLAIN



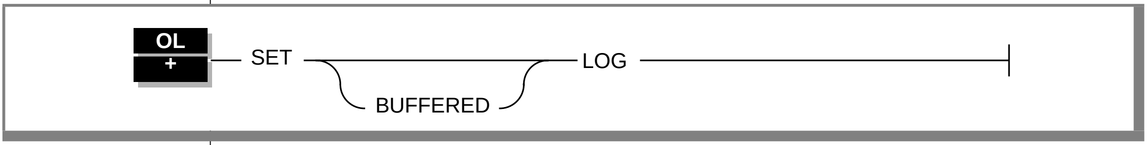
SET ISOLATION



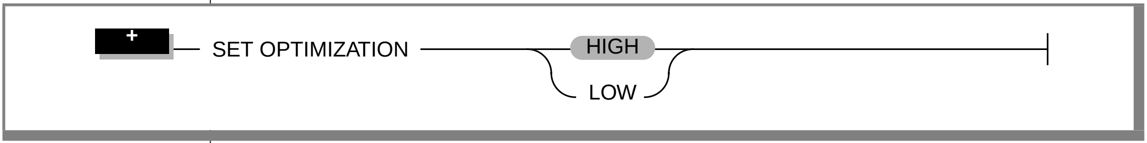
SET LOCK MODE TO



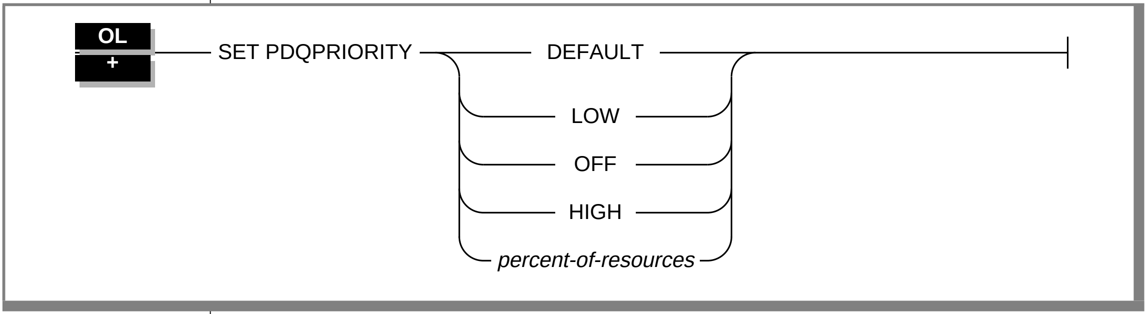
SET LOG



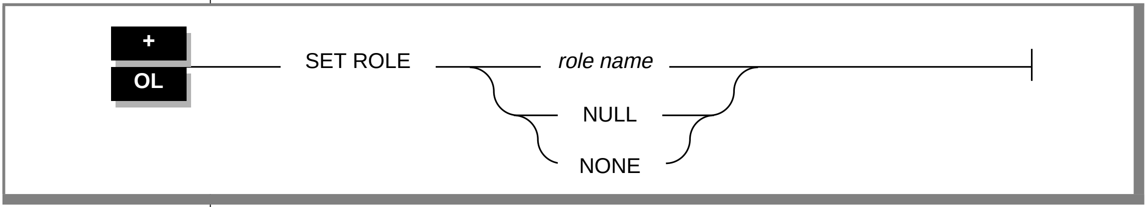
SET OPTIMIZATION



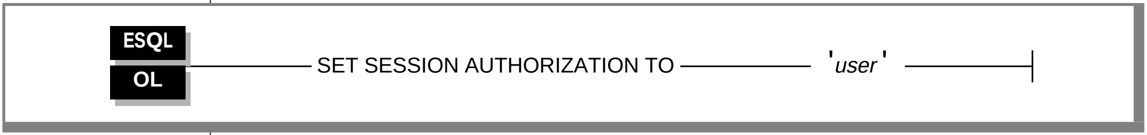
SET PDQPRIORITY



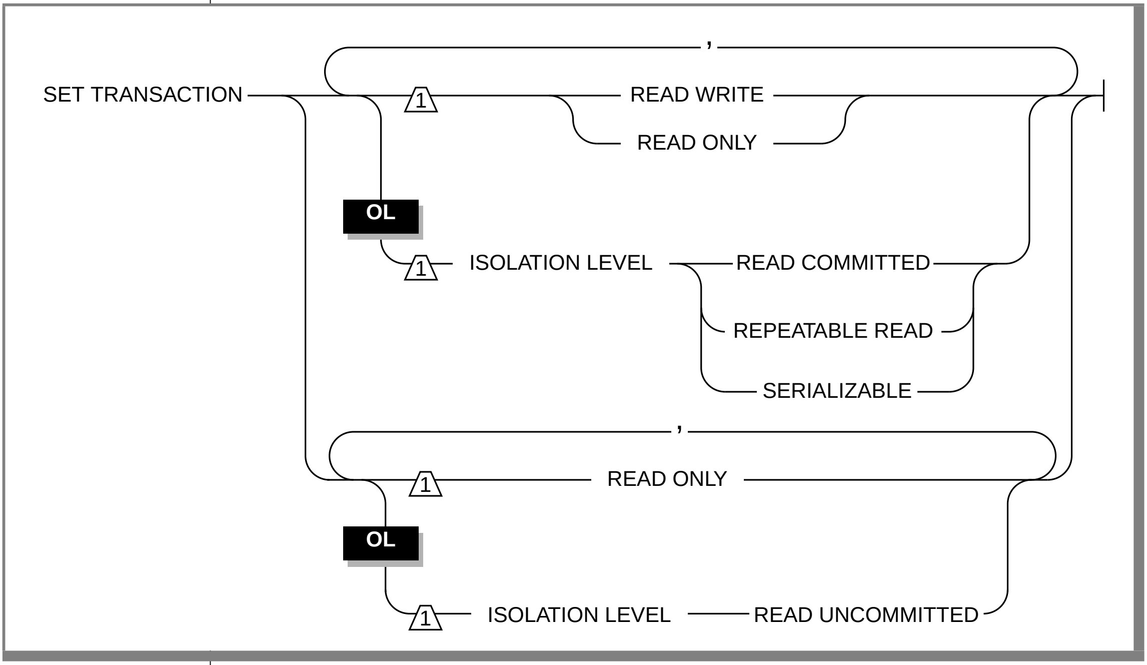
SET ROLE



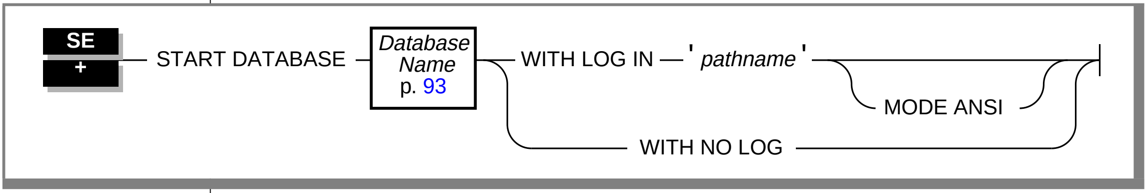
SET SESSION AUTHORIZATION



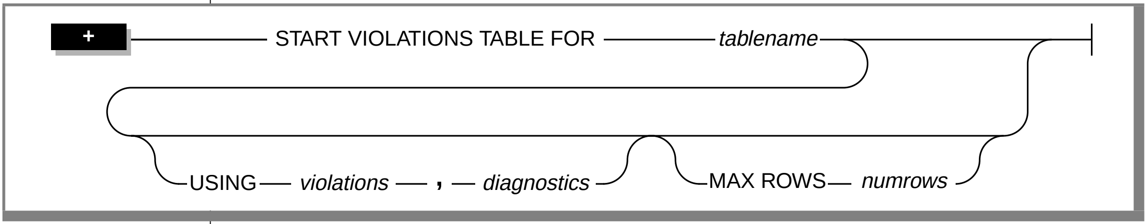
SET TRANSACTION



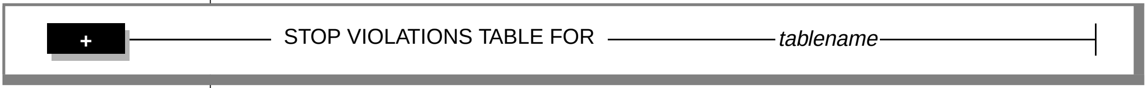
START DATABASE



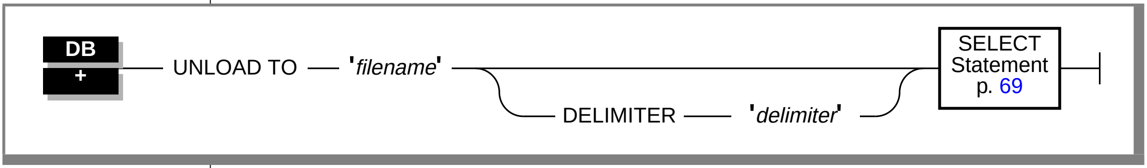
START VIOLATIONS TABLE



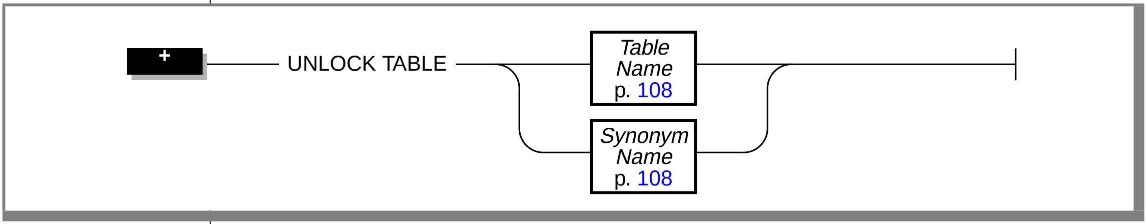
STOP VIOLATIONS TABLE



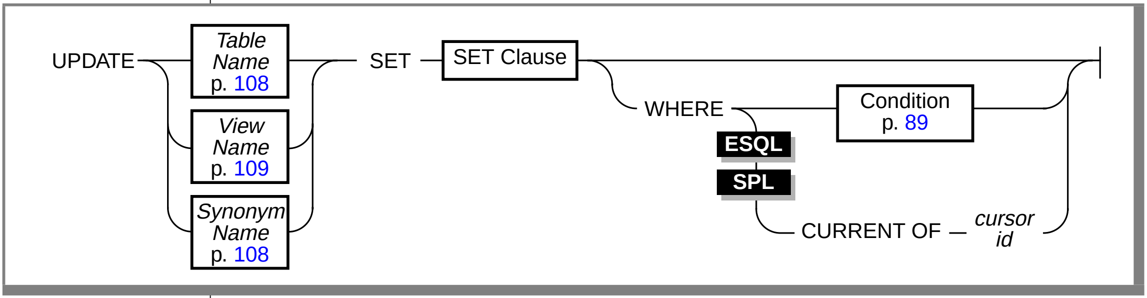
UNLOAD



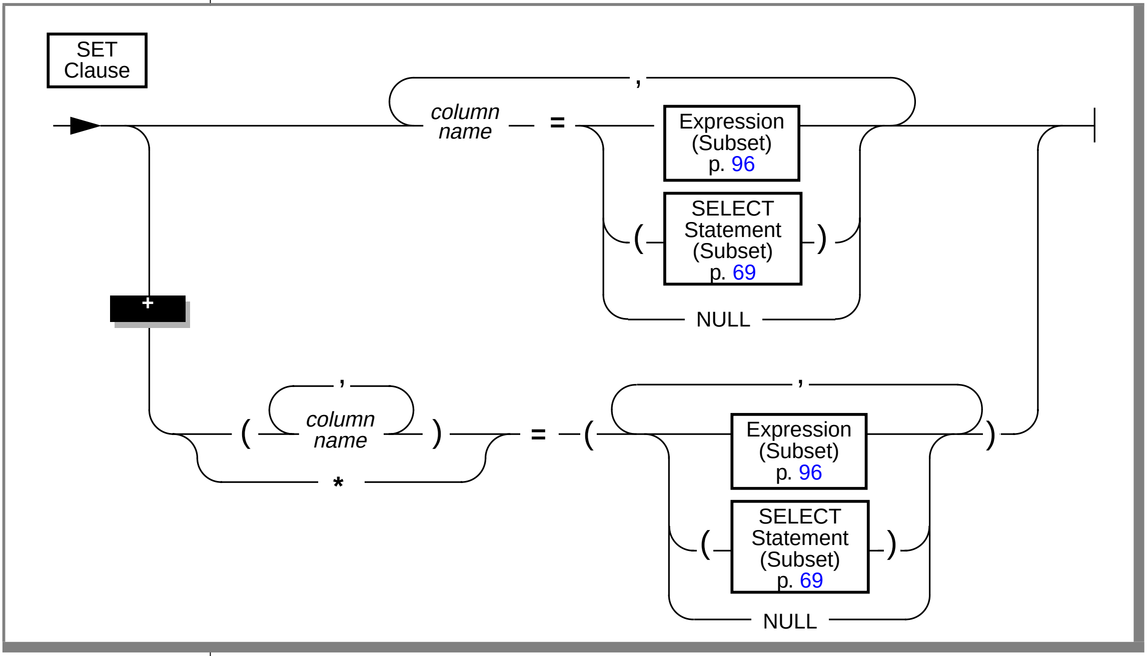
UNLOCK TABLE



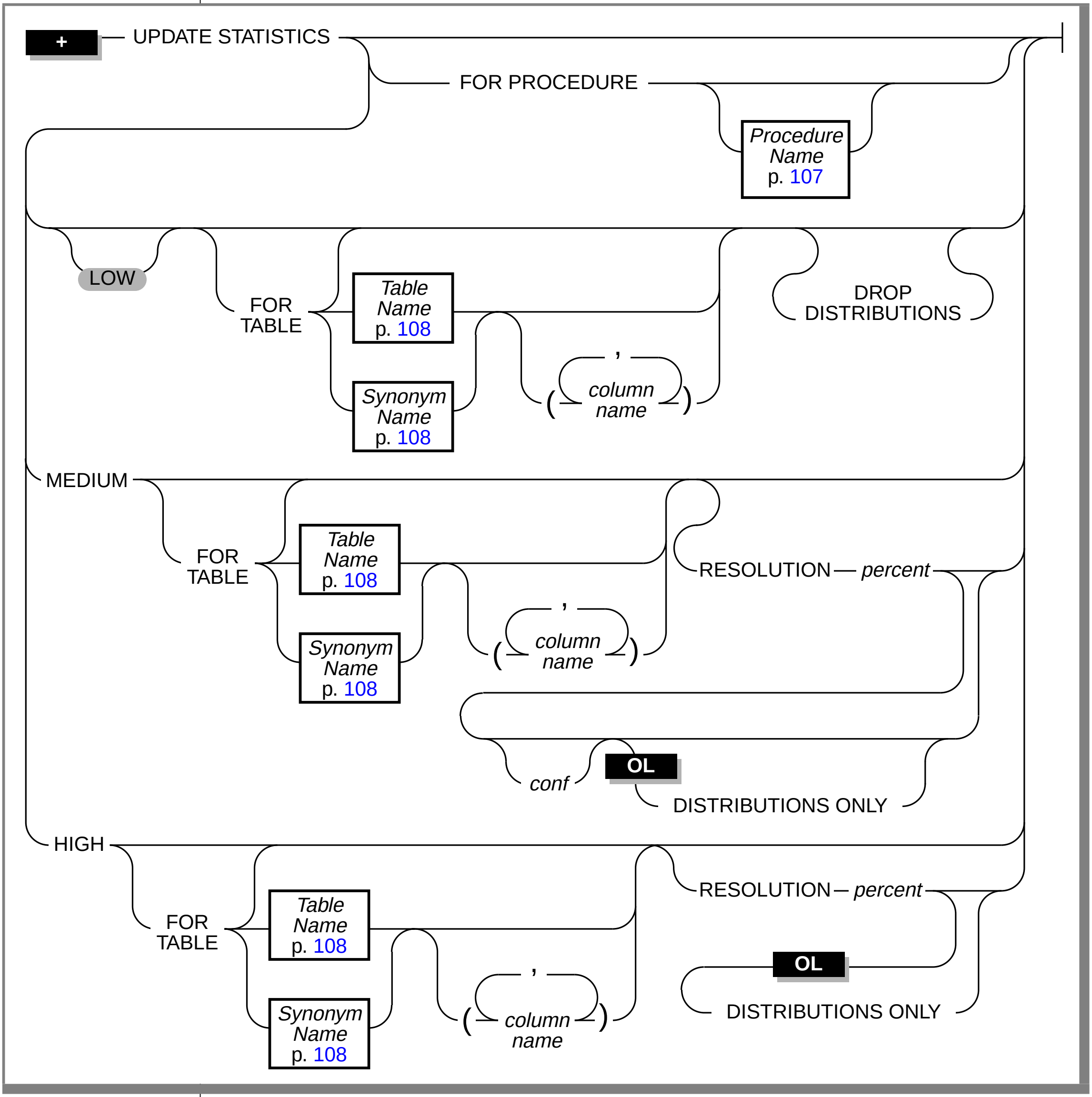
UPDATE



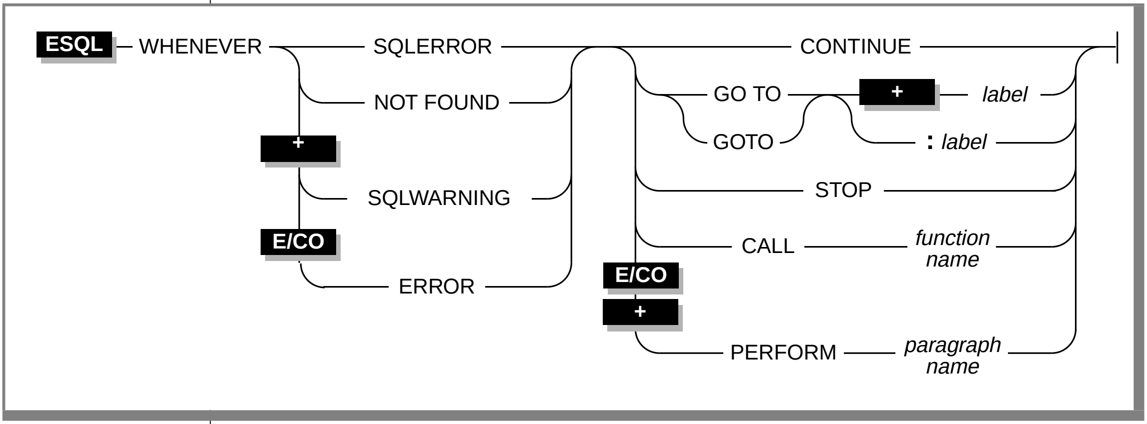
SET Clause


SET Clause
column name
=
,
Expression (Subset) p. 96
(
SELECT Statement (Subset) p. 69
)
NULL
+
(
,
column name
)
*
=
(
,
Expression (Subset) p. 96
(
SELECT Statement (Subset) p. 69
)
NULL
)

UPDATE STATISTICS

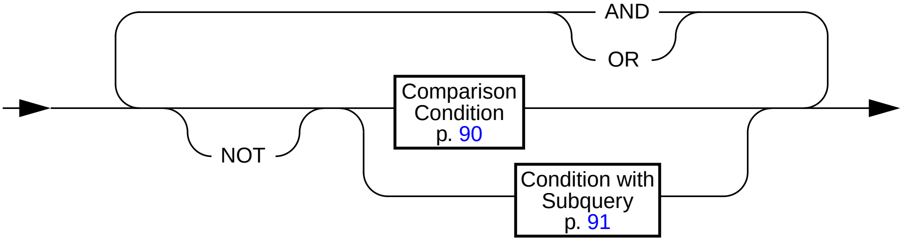


WHENEVER

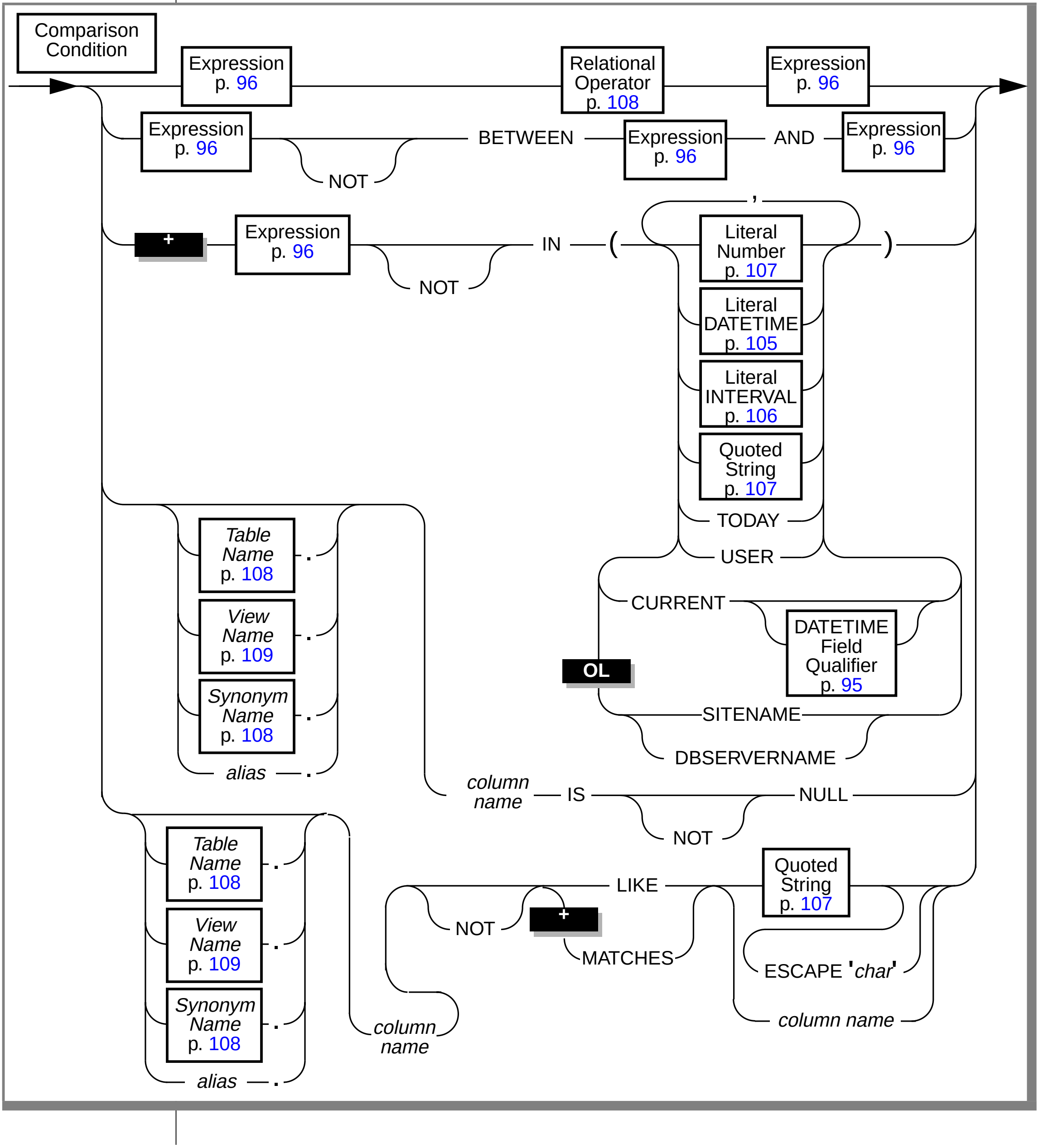

ESQL
WHENEVER
SQLERROR
NOT FOUND
+
SQLWARNING
E/CO
ERROR
CONTINUE
GO TO
GOTO
+
label
: label
STOP
CALL
function name
E/CO
+
PERFORM
paragraph name

SQL Segments

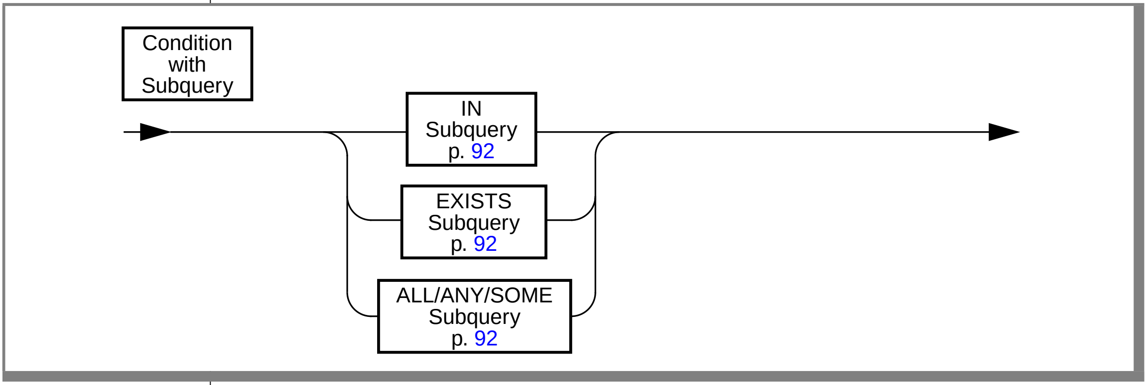
Condition


AND
OR
NOT
Comparison Condition p. 90
Condition with Subquery p. 91

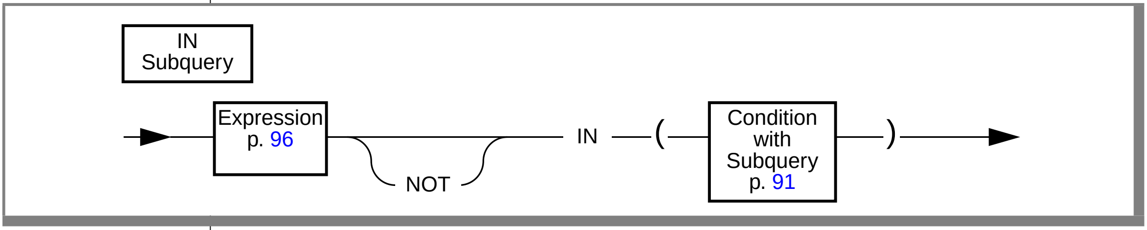
Comparison Conditions (Boolean Expressions)



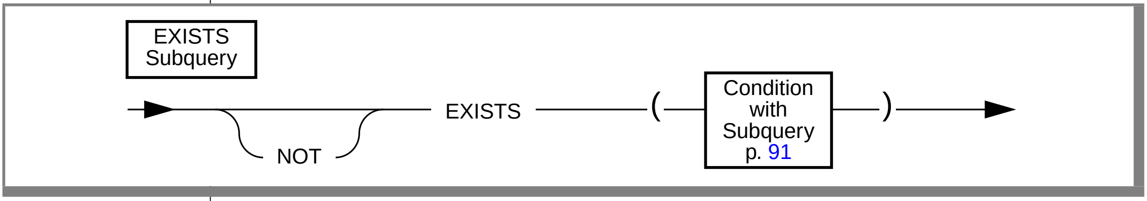
Condition with a Subquery



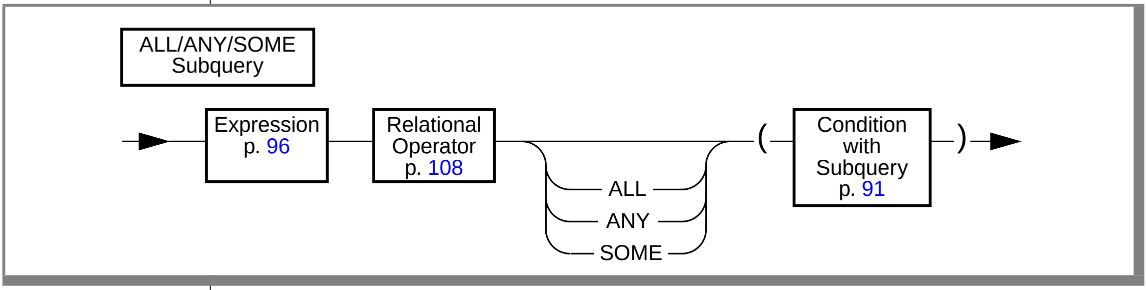
IN Subquery



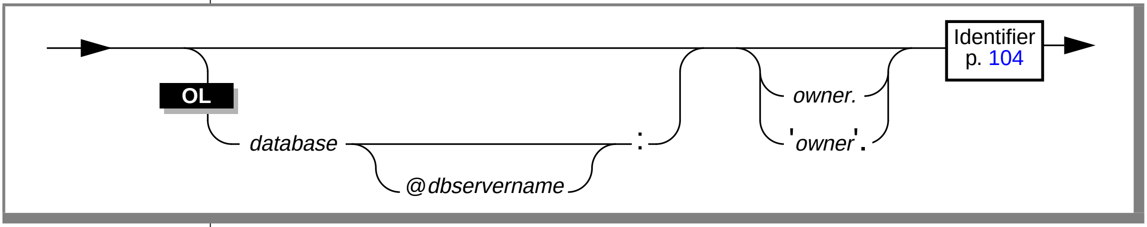
EXISTS Subquery



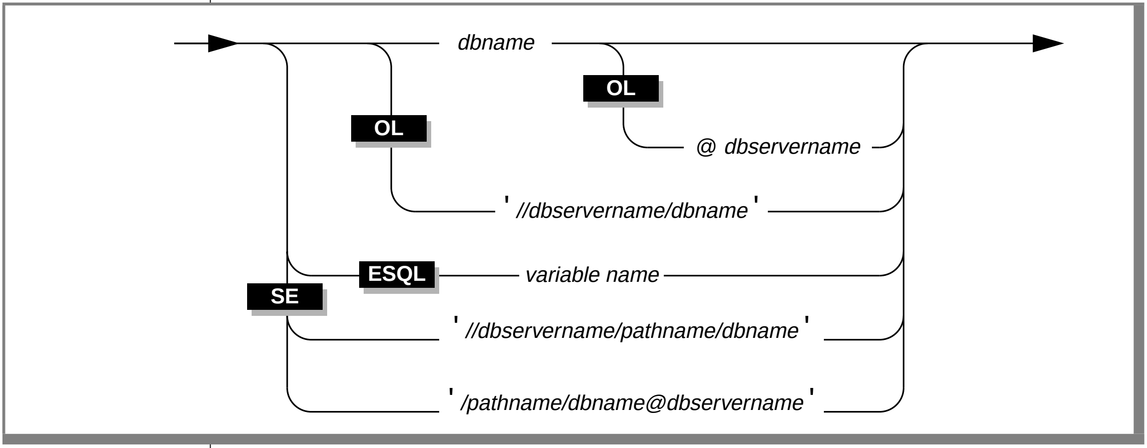
ALL/ANY/SOME Subquery



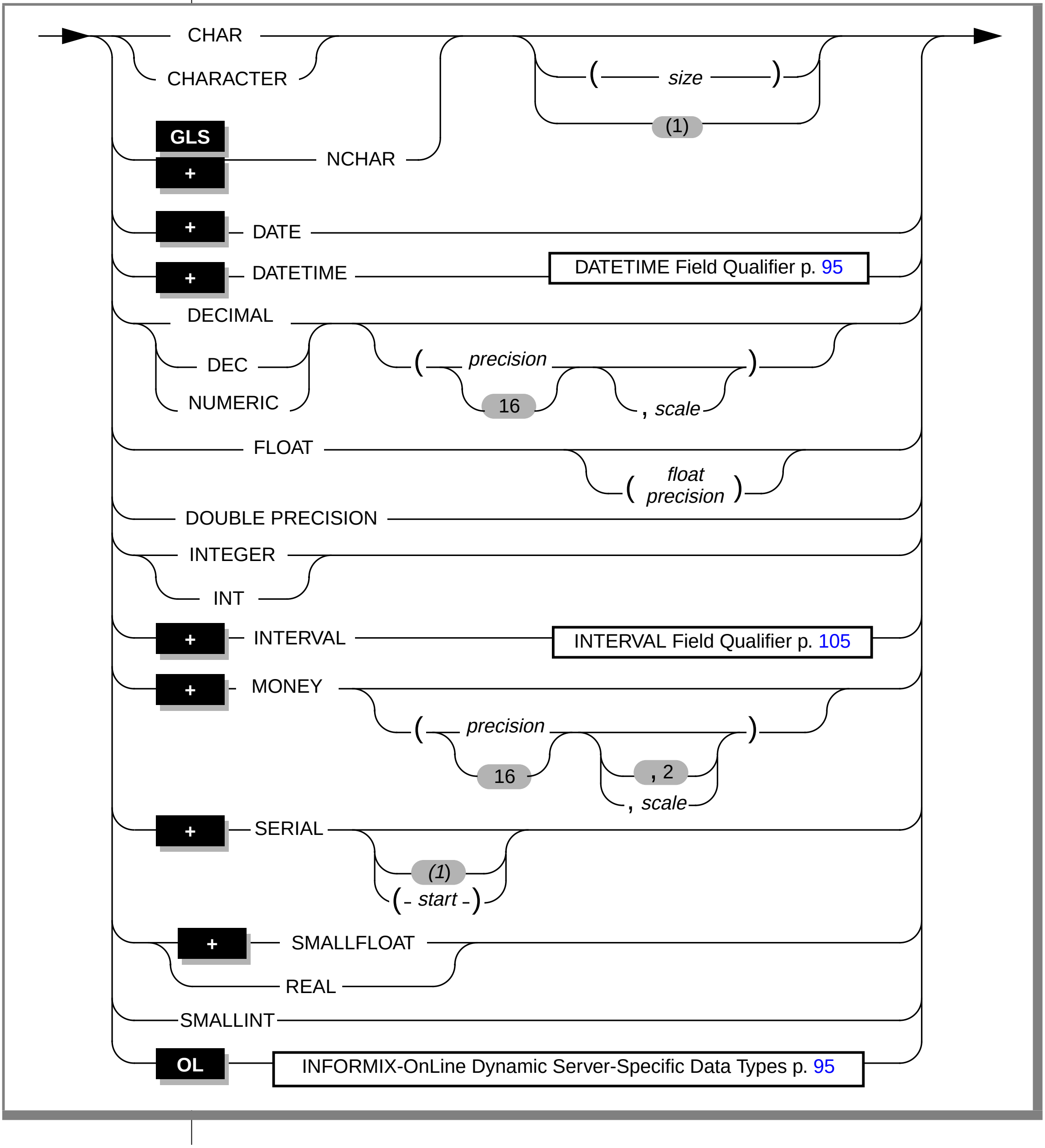
Constraint Name



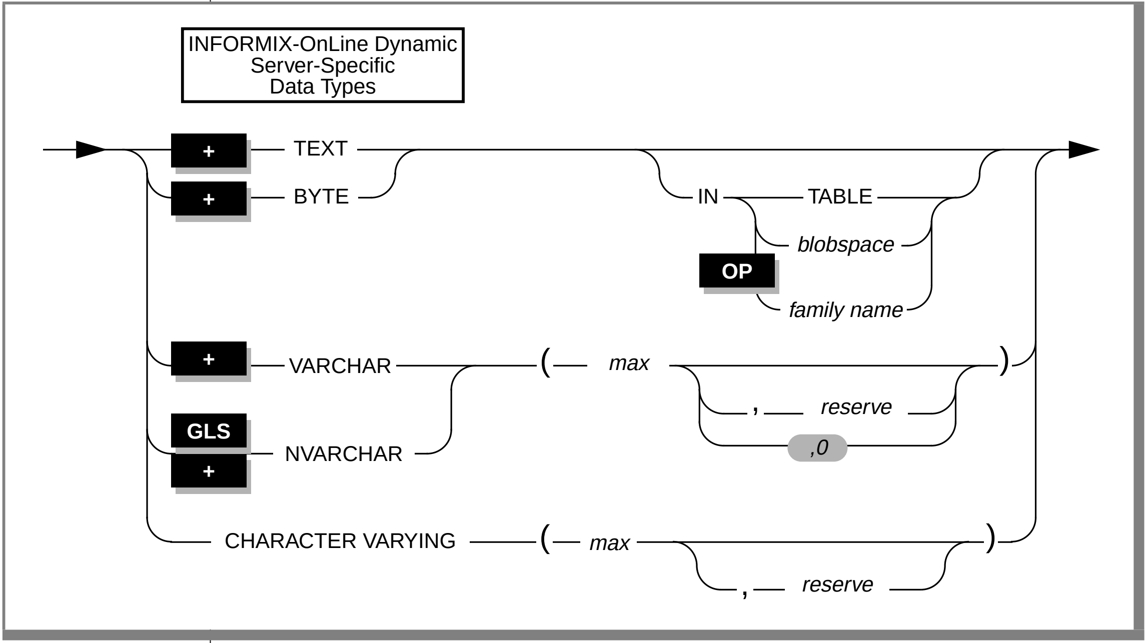
Database Name


dbname
OL
@ dbservername
OL
' //dbservername/dbname '
ESQL
variable name
SE
' //dbservername/pathname/dbname '
' /pathname/dbname@dbservername '

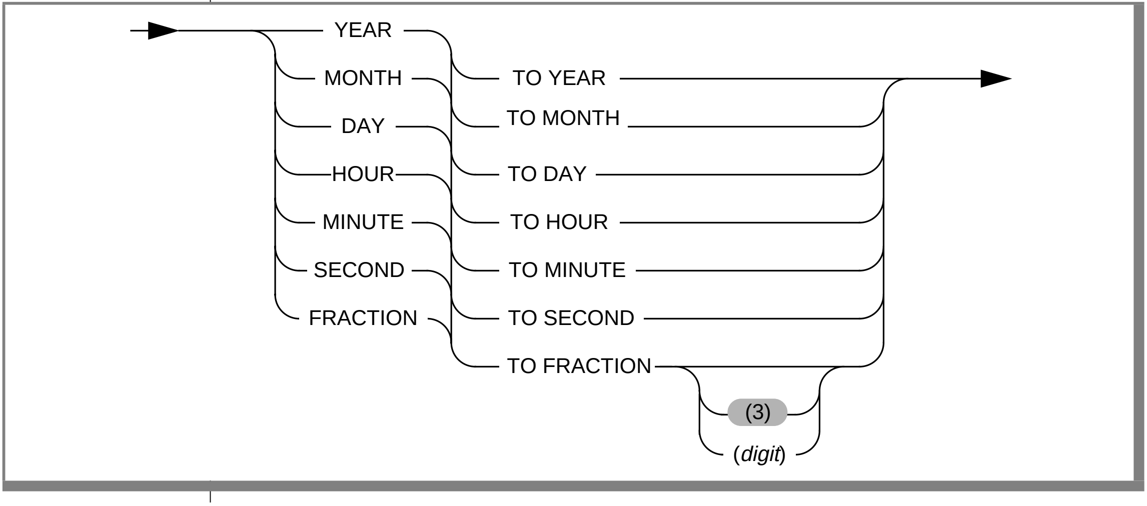
Data Type


CHAR
CHARACTER
(size)
(1)
GLS
+
NCHAR
+
DATE
+
DATETIME
DATETIME Field Qualifier p. 95
DECIMAL
DEC
NUMERIC
(precision)
16
, scale
FLOAT
(float precision)
DOUBLE PRECISION
INTEGER
INT
+
INTERVAL
INTERVAL Field Qualifier p. 105
+
MONEY
(precision)
16
, 2
, scale
+
SERIAL
(1)
(start)
+
SMALLFLOAT
REAL
SMALLINT
OL
INFORMIX-OnLine Dynamic Server-Specific Data Types p. 95

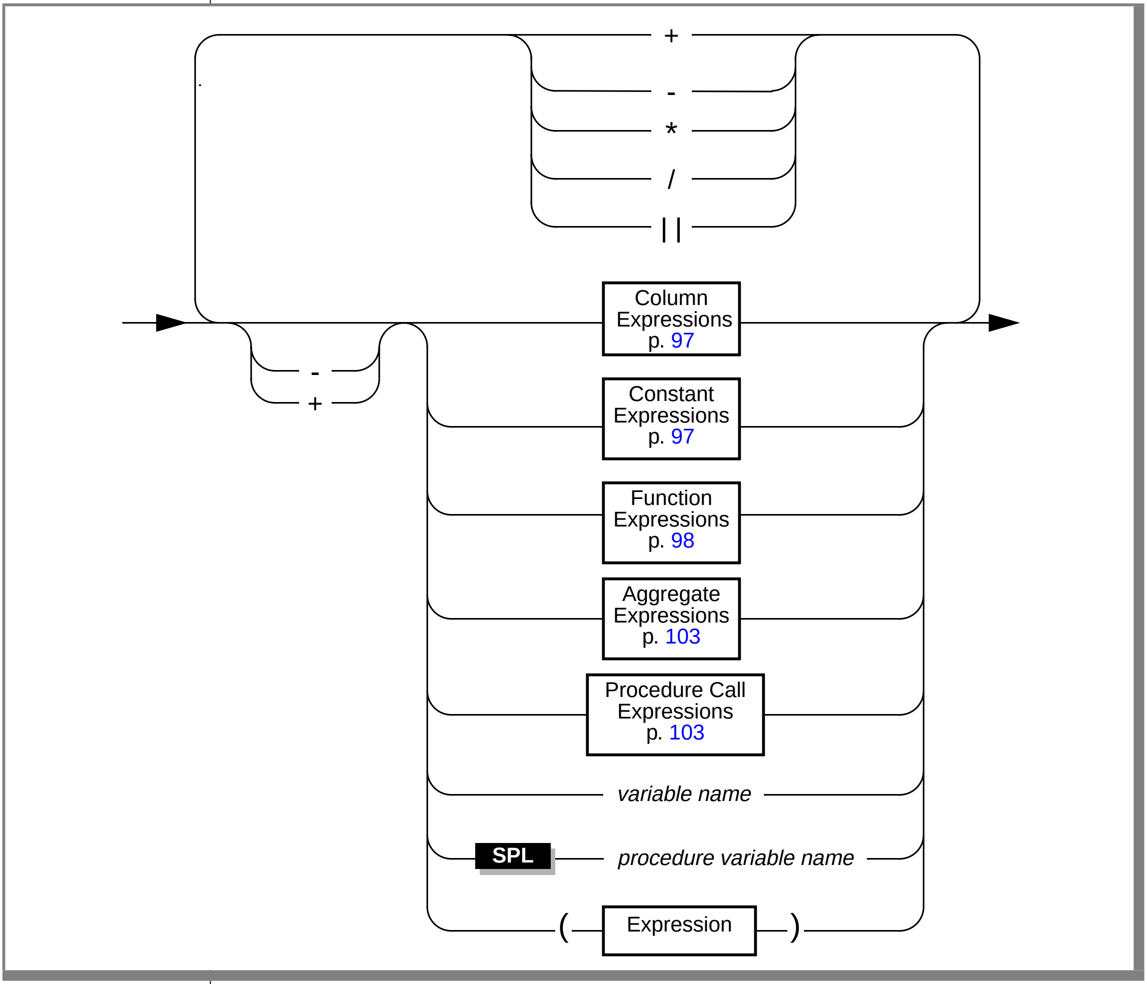
INFORMIX-OnLine Dynamic Server-Specific Data Types



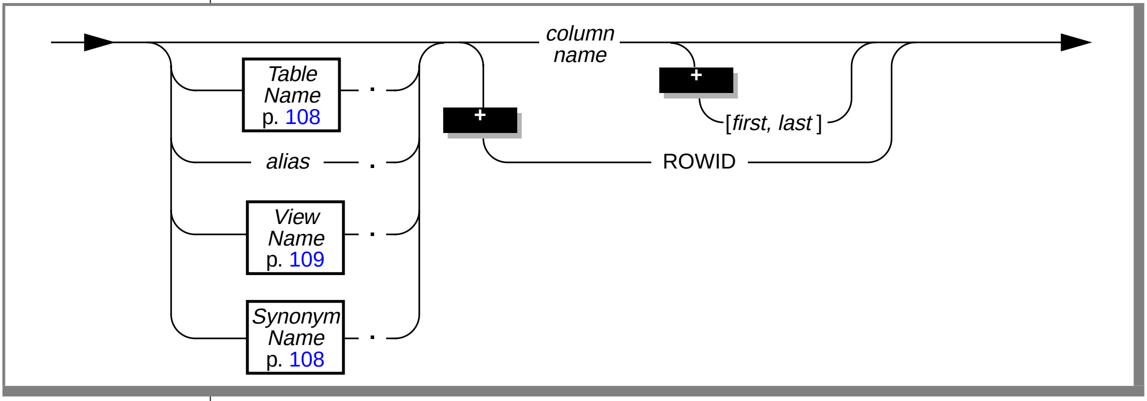
DATETIME Field Qualifier



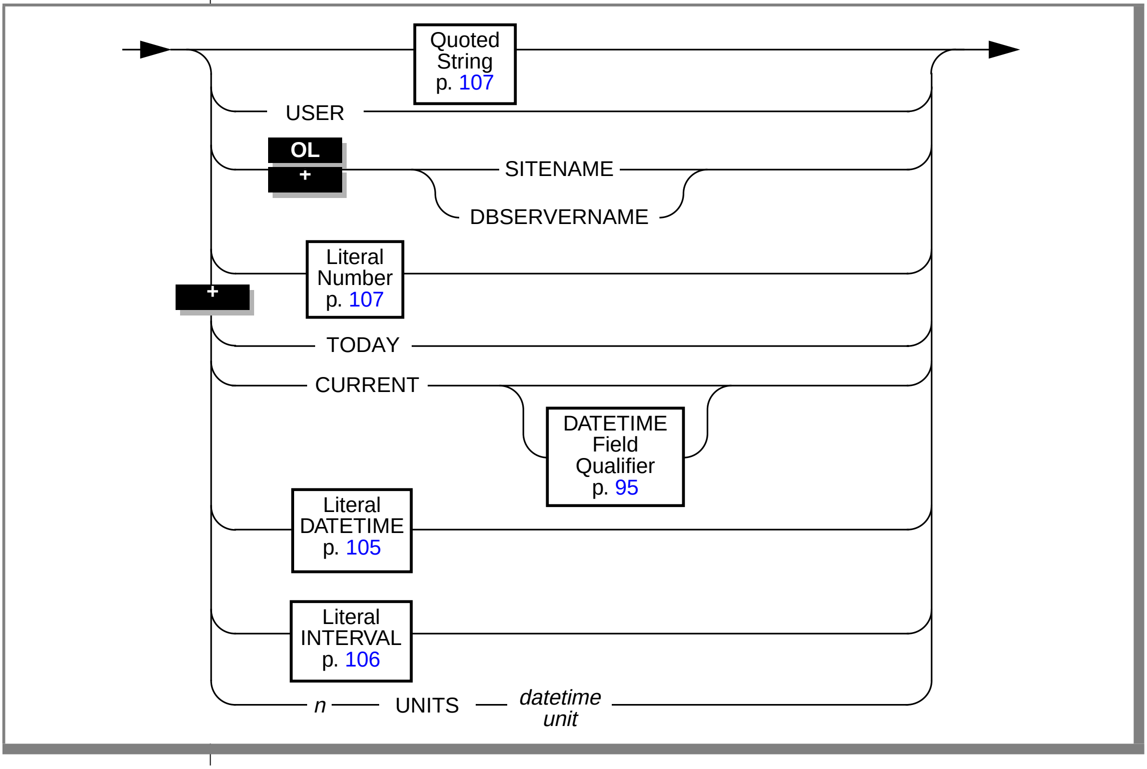
Expression


+
-
*
/
||
-
+
Column Expressions p. 97
Constant Expressions p. 97
Function Expressions p. 98
Aggregate Expressions p. 103
Procedure Call Expressions p. 103
variable name
SPL
procedure variable name
(
Expression
)

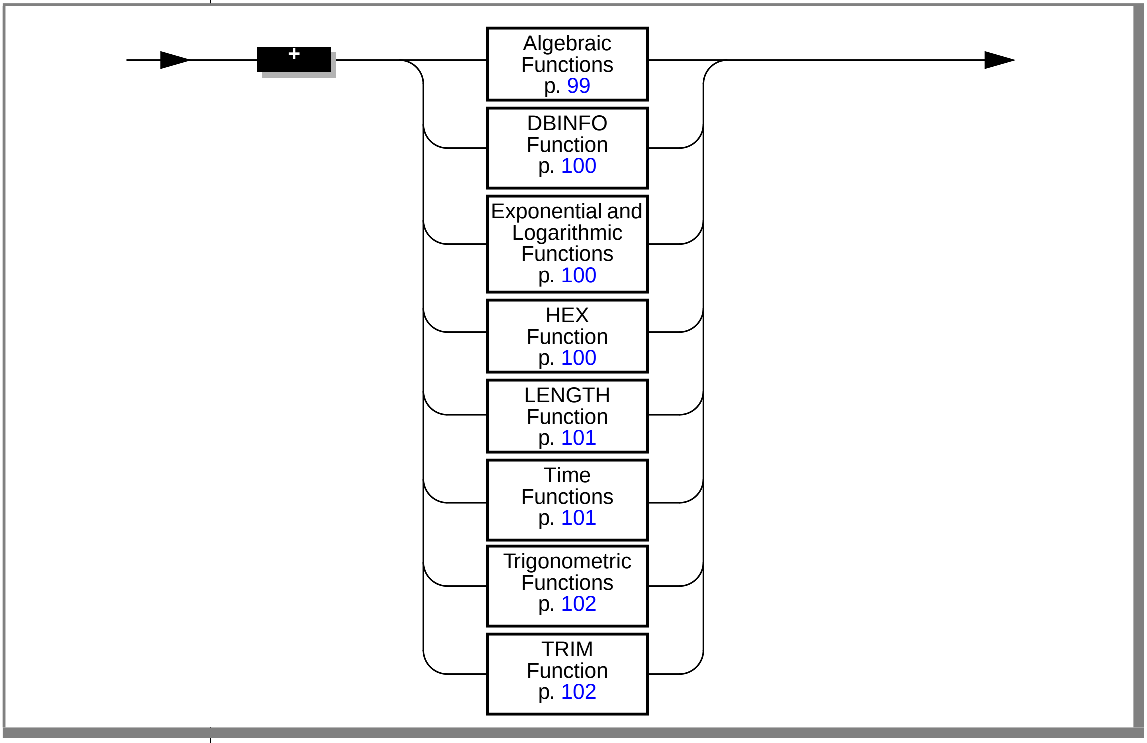
Column Expressions



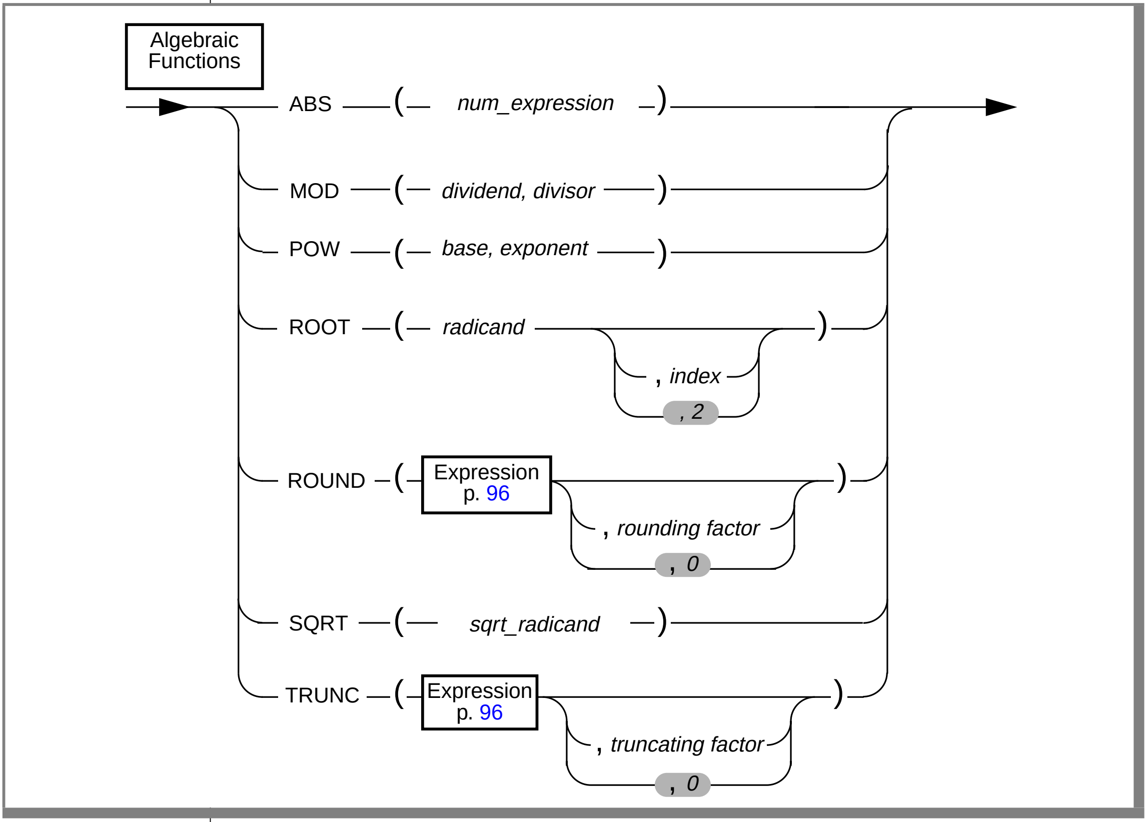
Constant Expressions



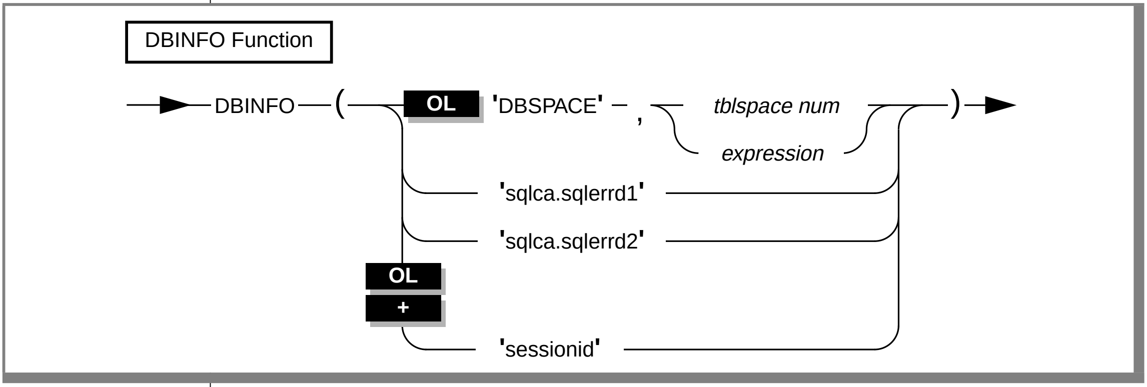
Function Expressions



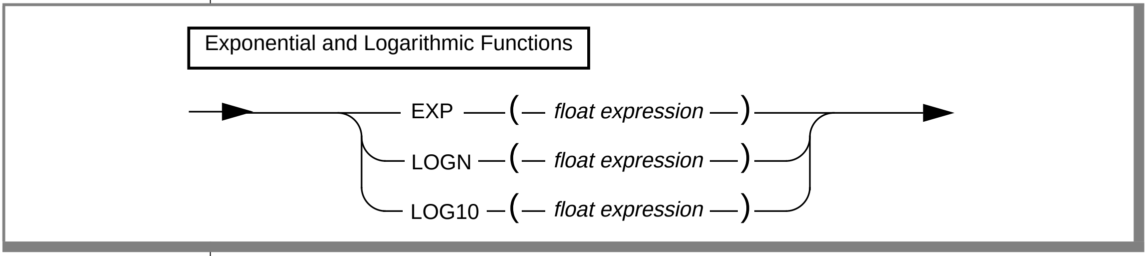
Algebraic Functions



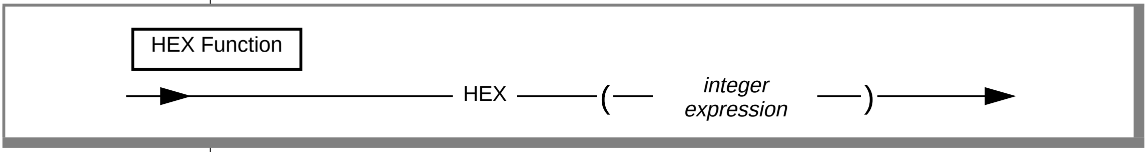
DBINFO Function



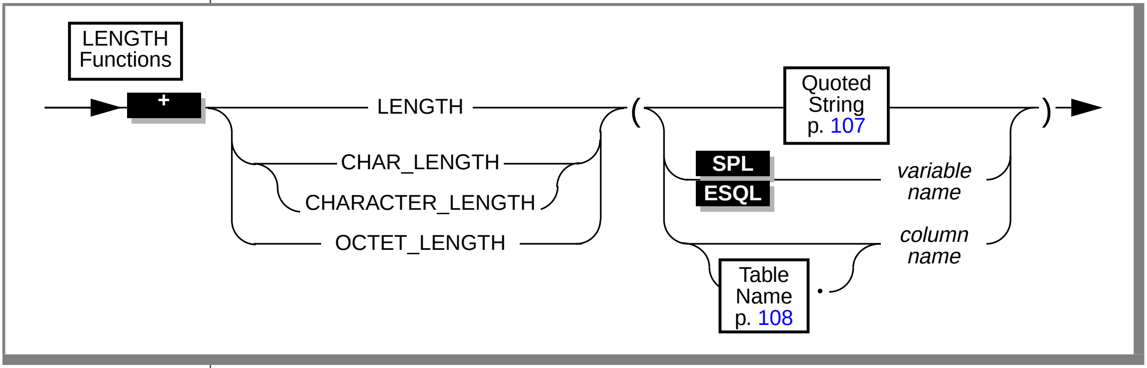
Exponential and Logarithmic Functions



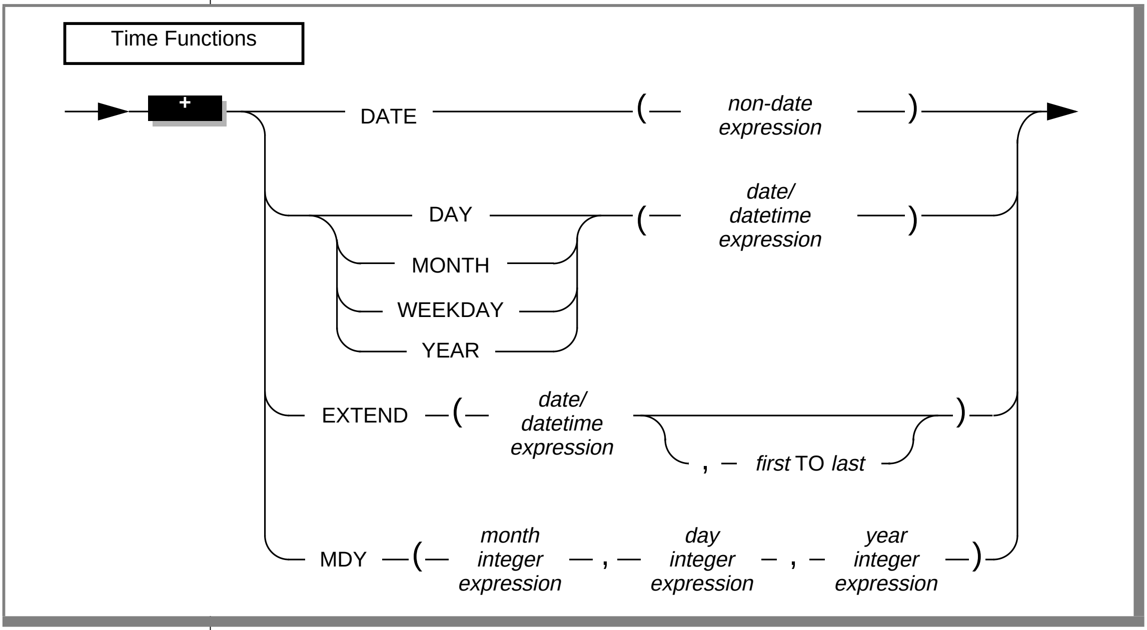
HEX Function



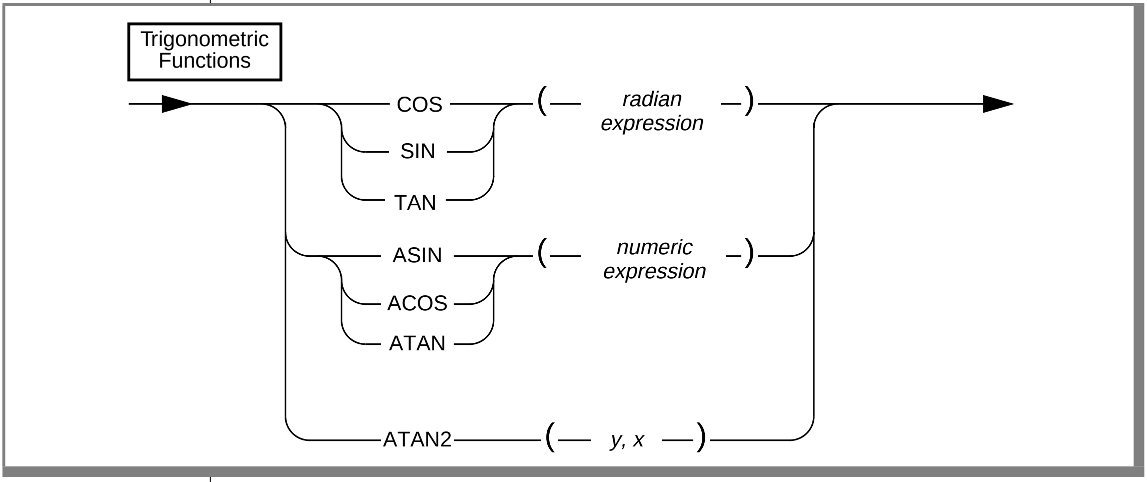
LENGTH Function



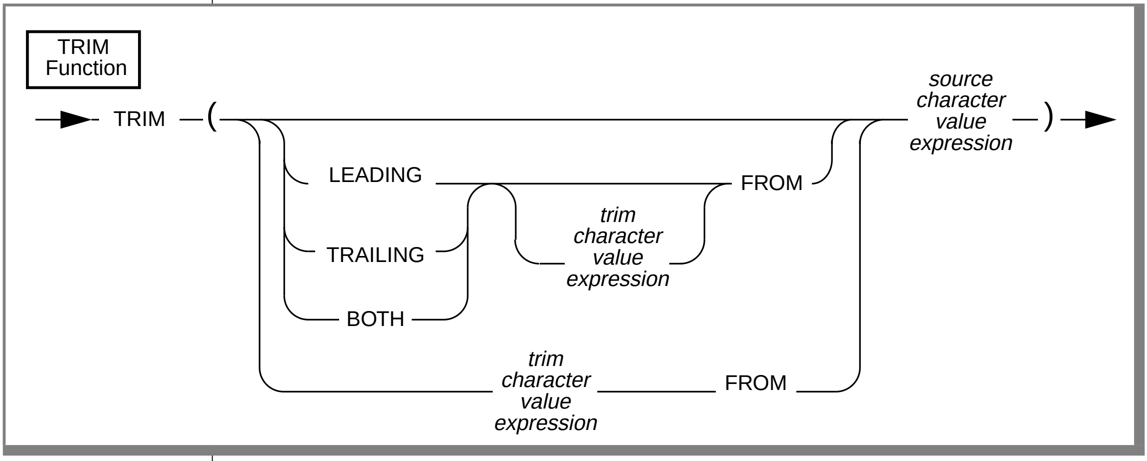
Time Functions



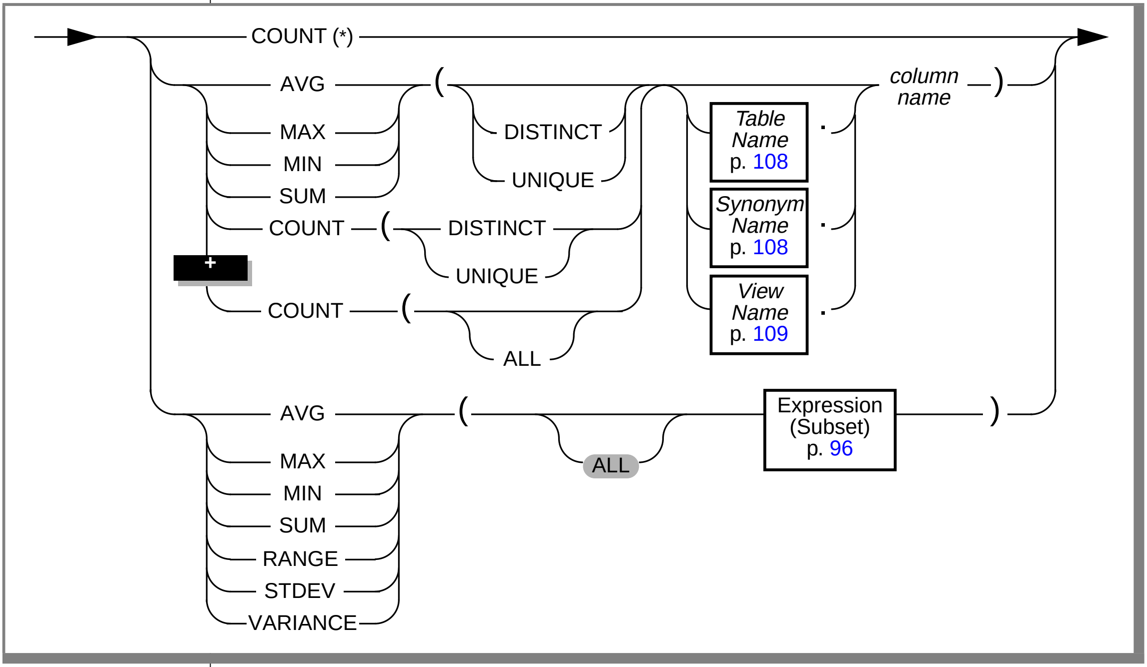
Trigonometric Functions



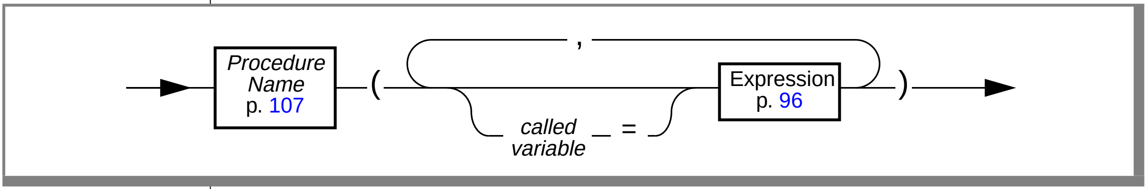
TRIM Function


TRIM Function
TRIM
(
LEADING
TRAILING
BOTH
trim character value expression
FROM
trim character value expression
FROM
source character value expression
)

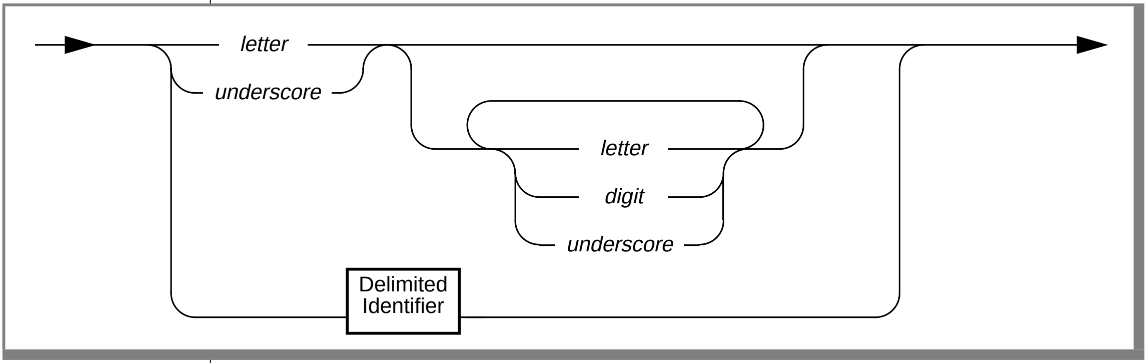
Aggregate Expressions



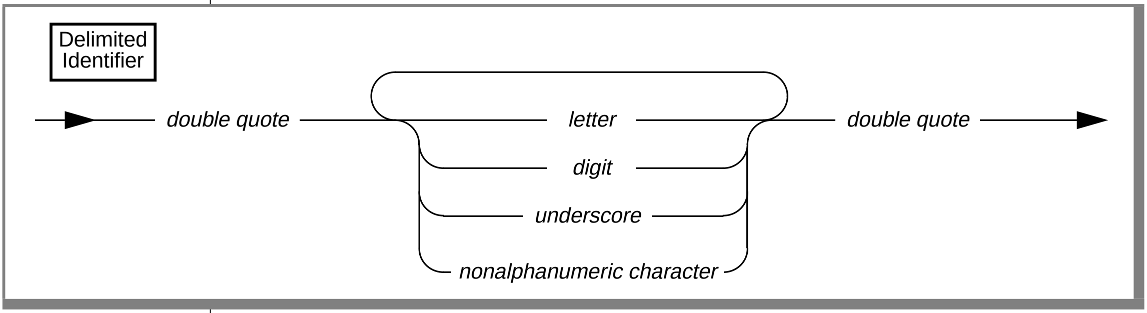
Procedure Call Expressions



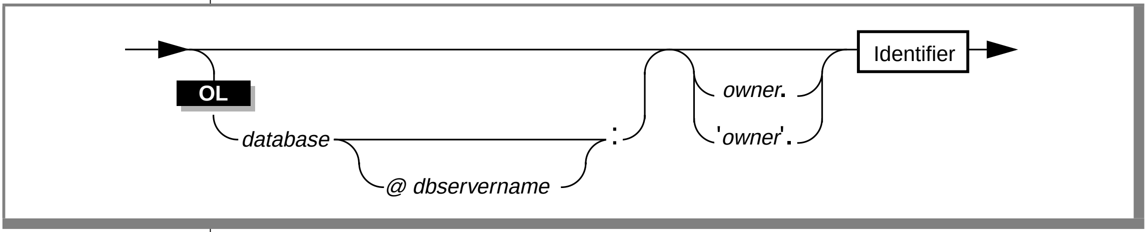
Identifier



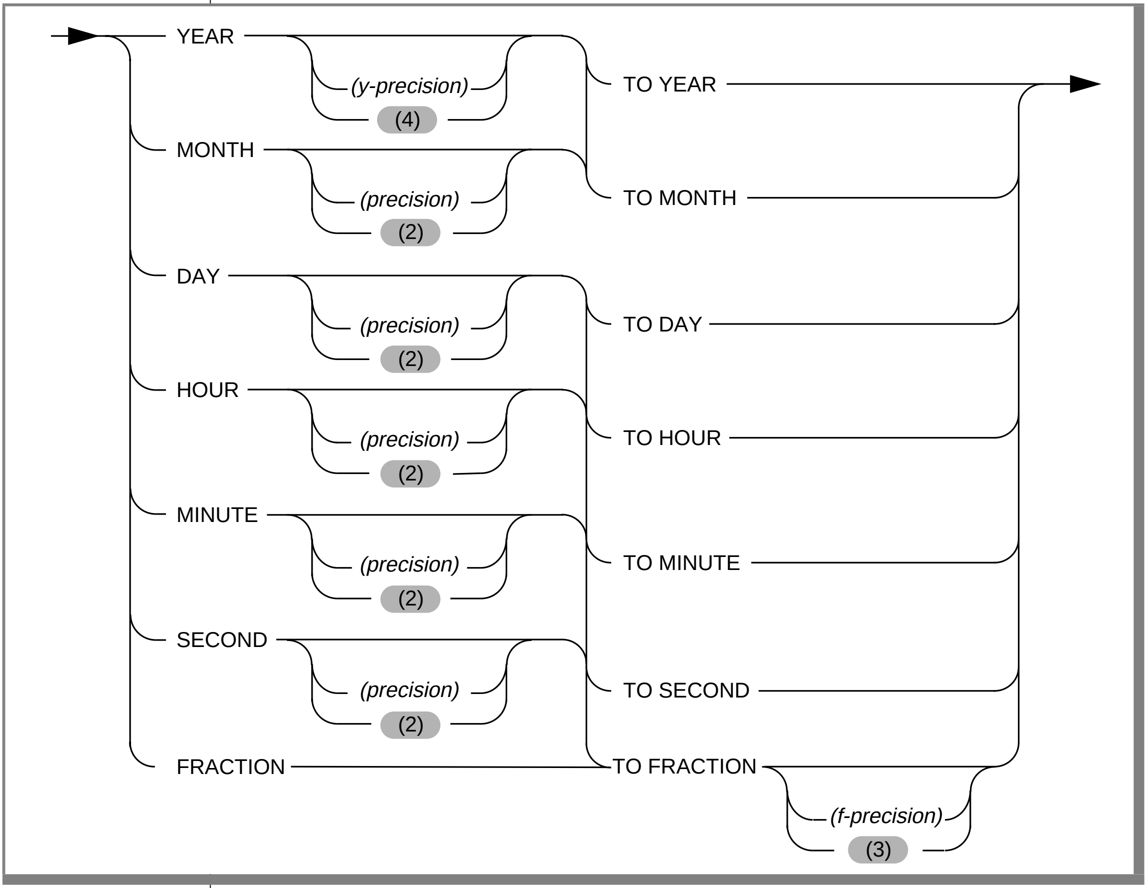
Delimited Identifiers



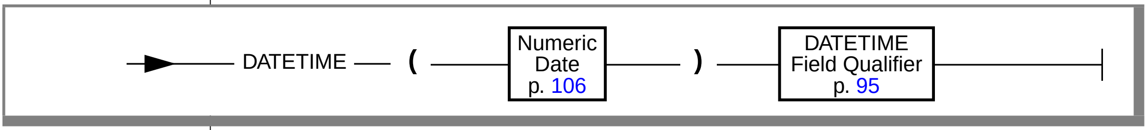
Index Name



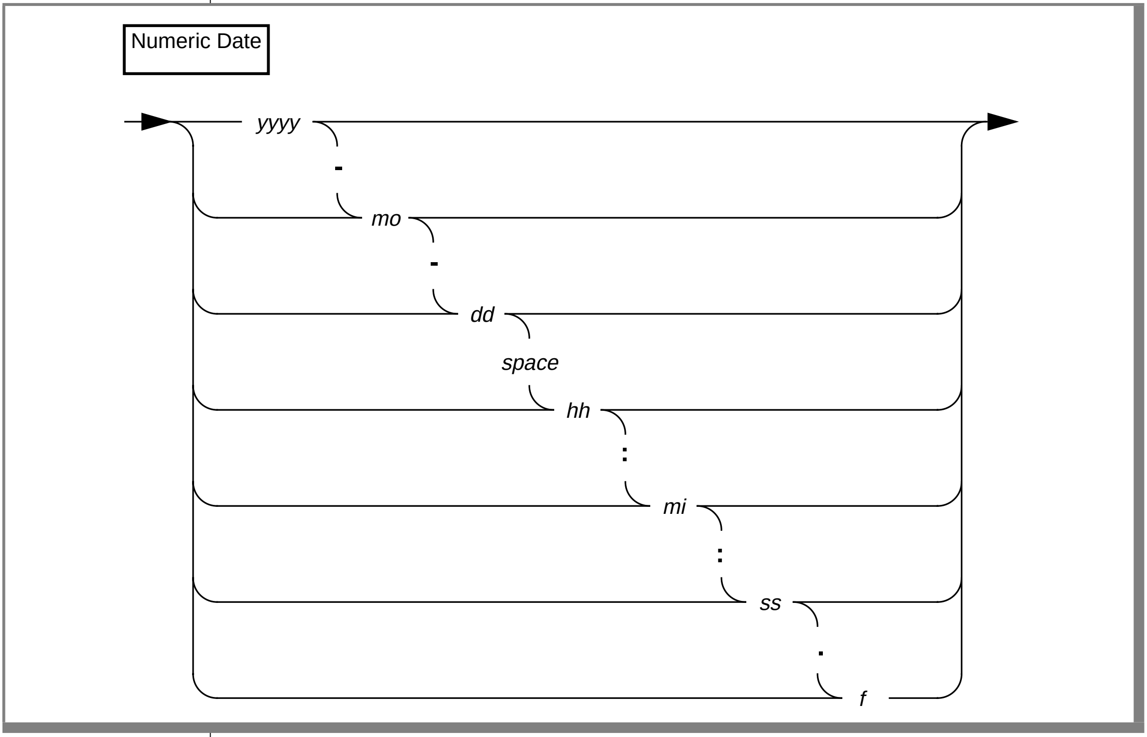
INTERVAL Field Qualifier



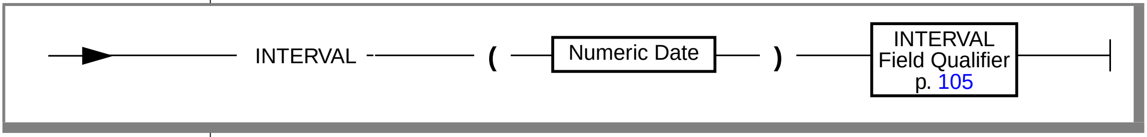
Literal DATETIME



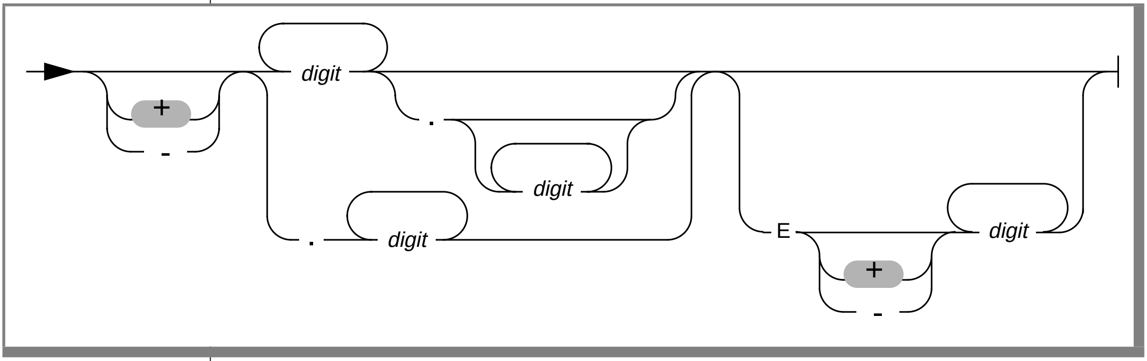
Numeric Date



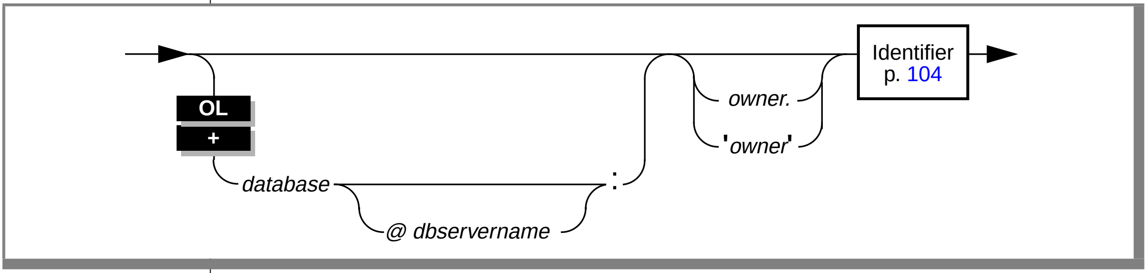
Literal Interval



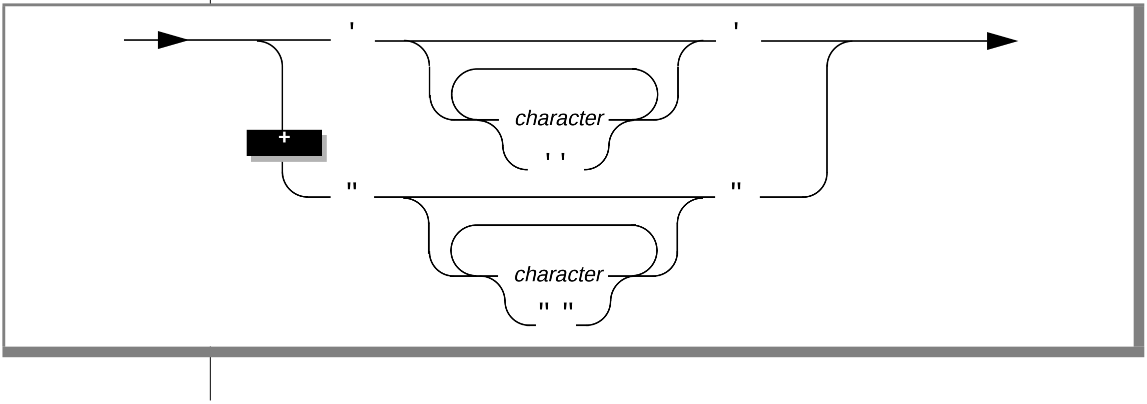
Literal Number



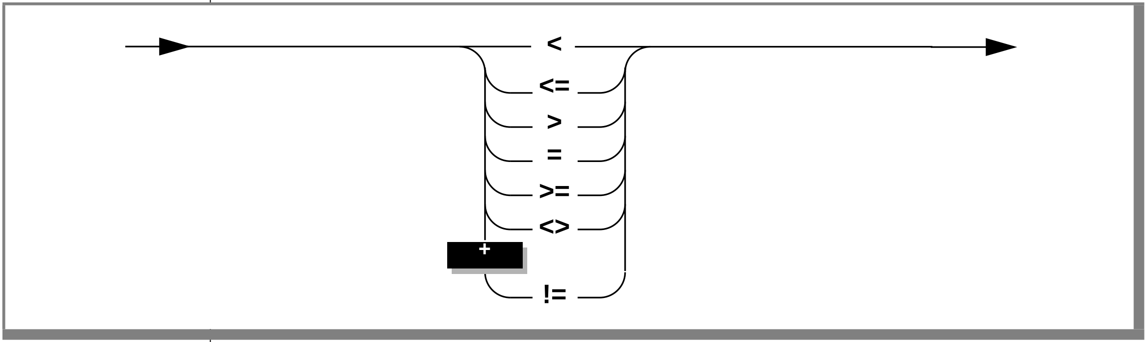
Procedure Name



Quoted String



Relational Operator



Synonym Name

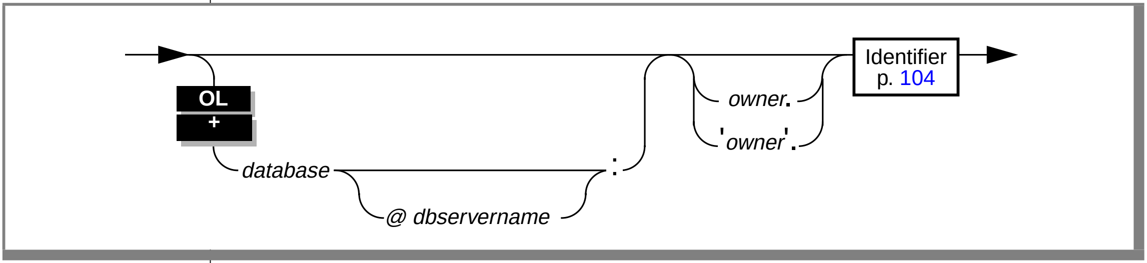
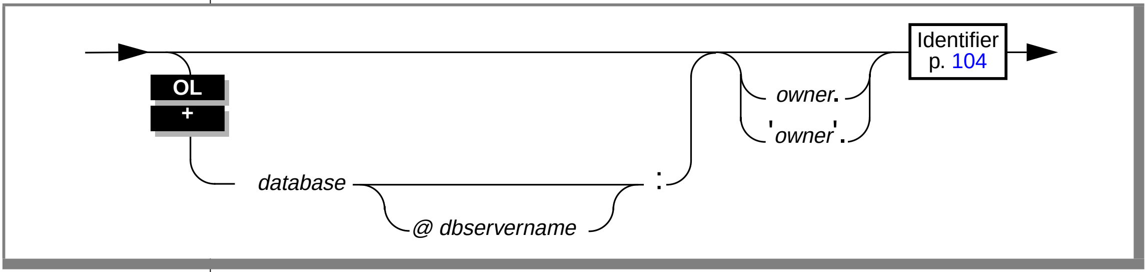
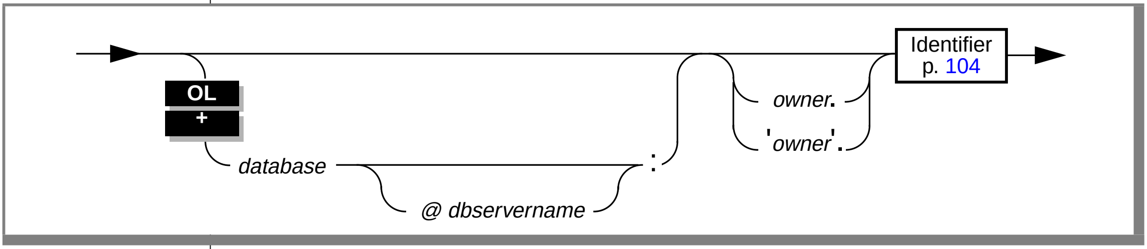


Table Name



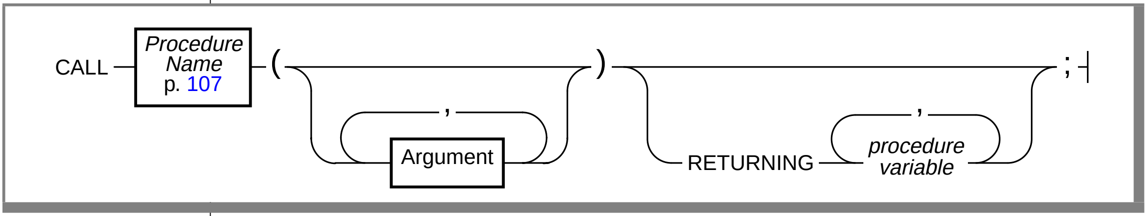
View Name



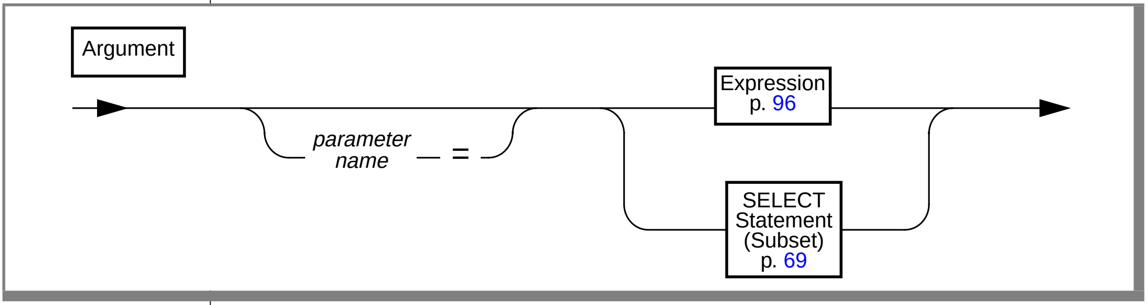
Stored Procedure Language Statements

Section III

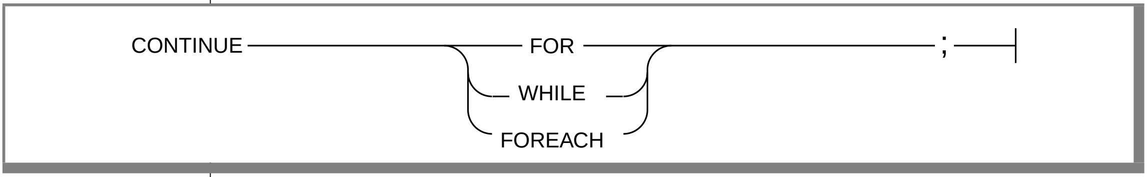
CALL



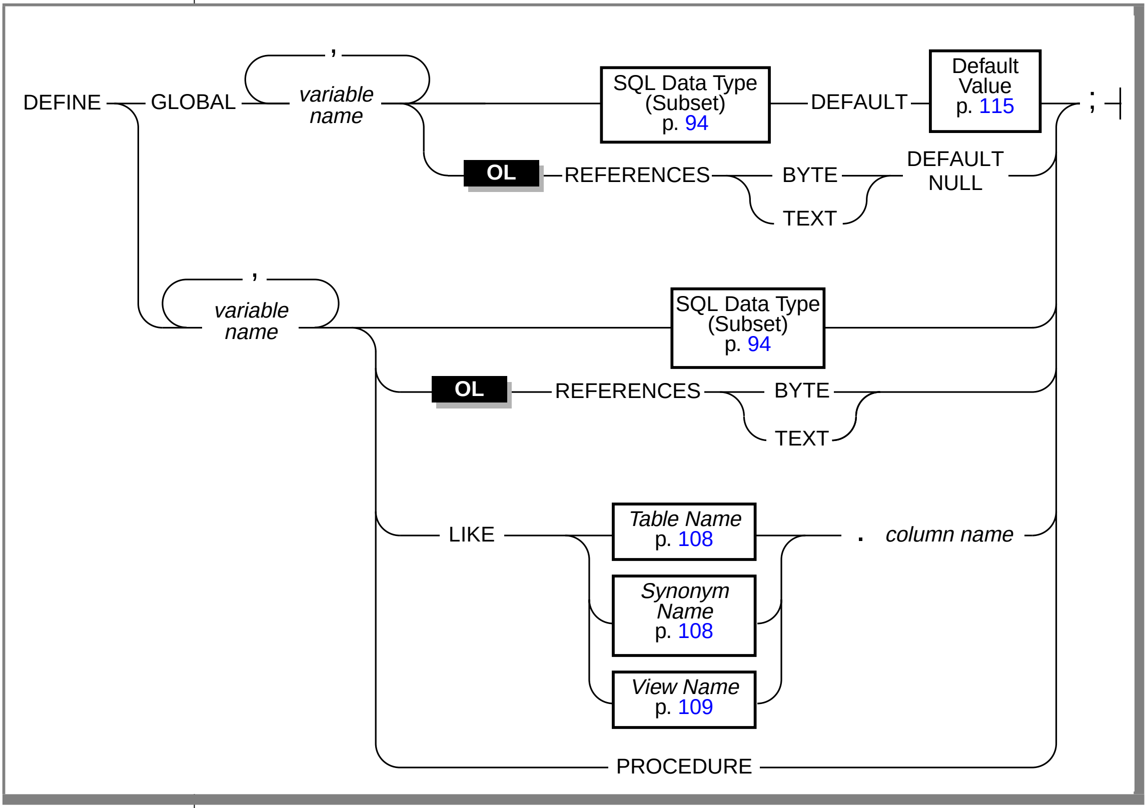
Argument



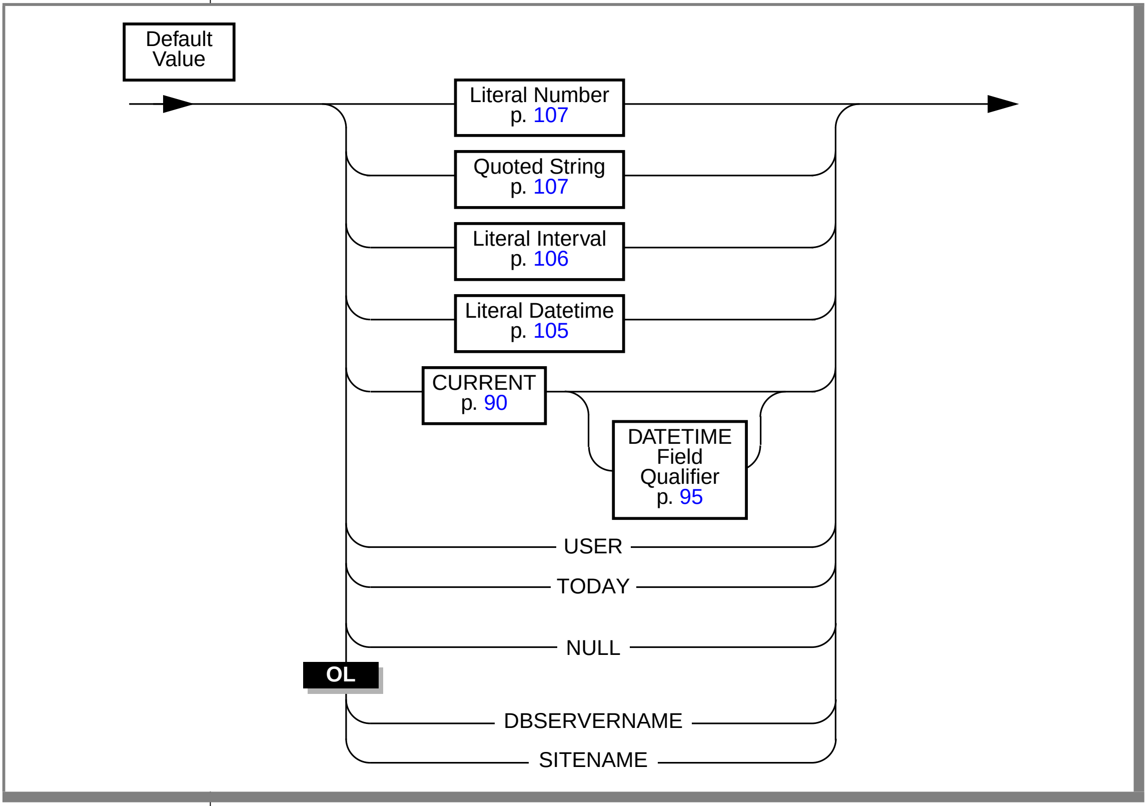
CONTINUE



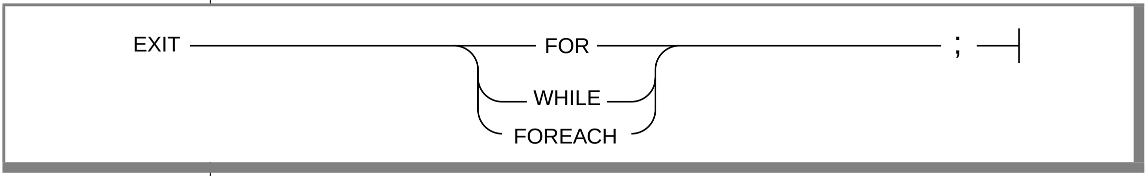
DEFINE


DEFINE
GLOBAL
,
variable name
SQL Data Type (Subset) p. 94
DEFAULT
Default Value p. 115
;
OL
REFERENCES
BYTE
TEXT
DEFAULT NULL
,
variable name
SQL Data Type (Subset) p. 94
OL
REFERENCES
BYTE
TEXT
LIKE
Table Name p. 108
Synonym Name p. 108
View Name p. 109
.
column name
PROCEDURE

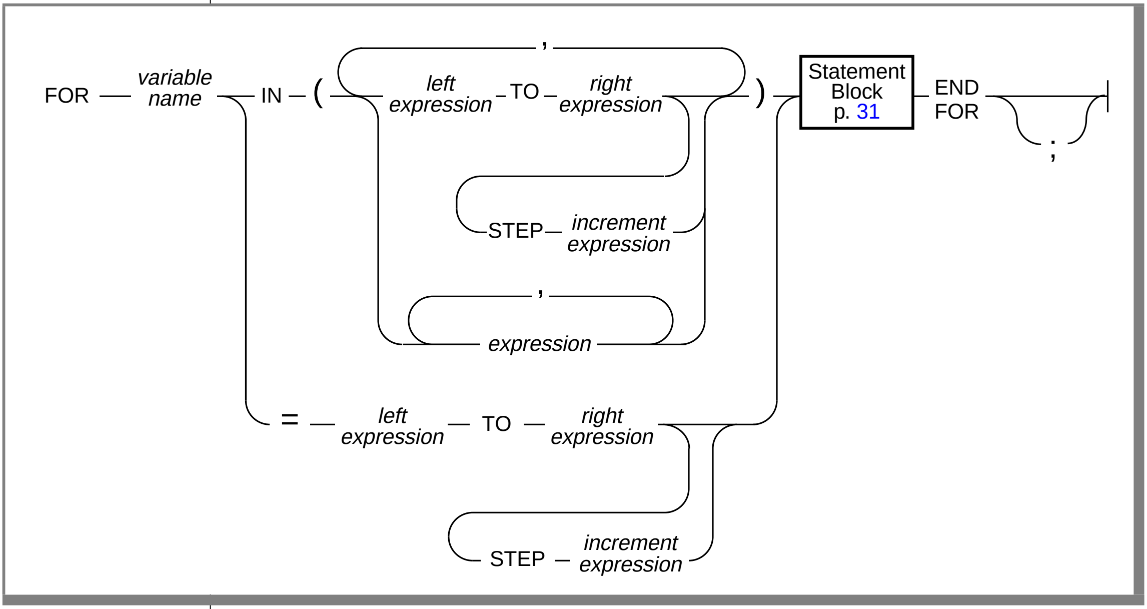
Default Value Clause



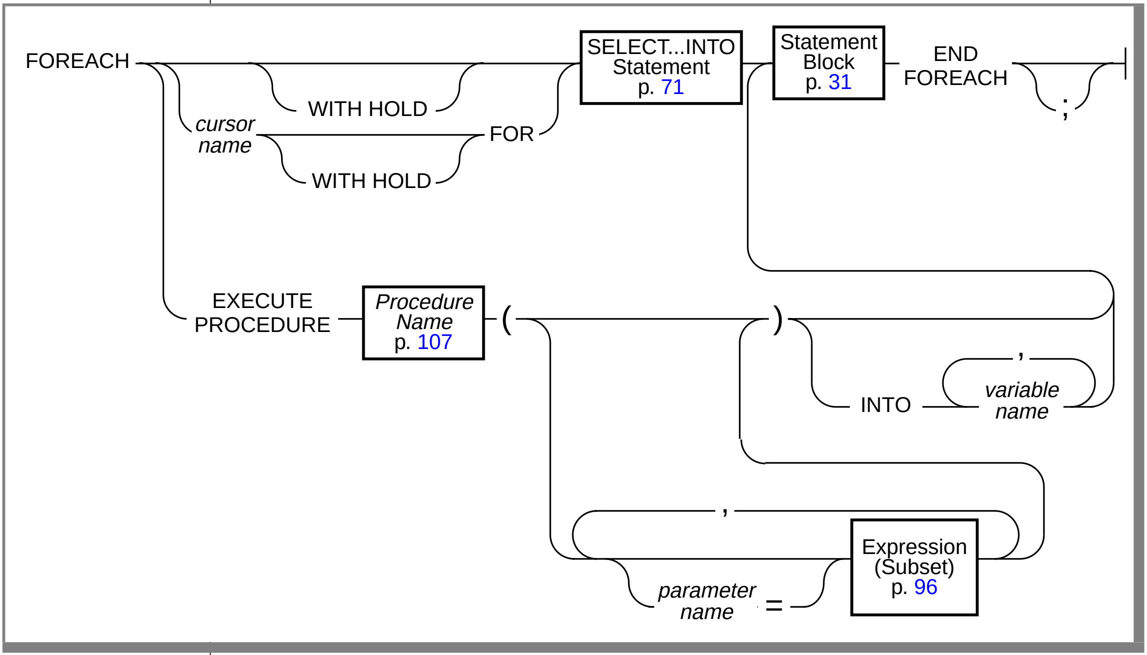
EXIT



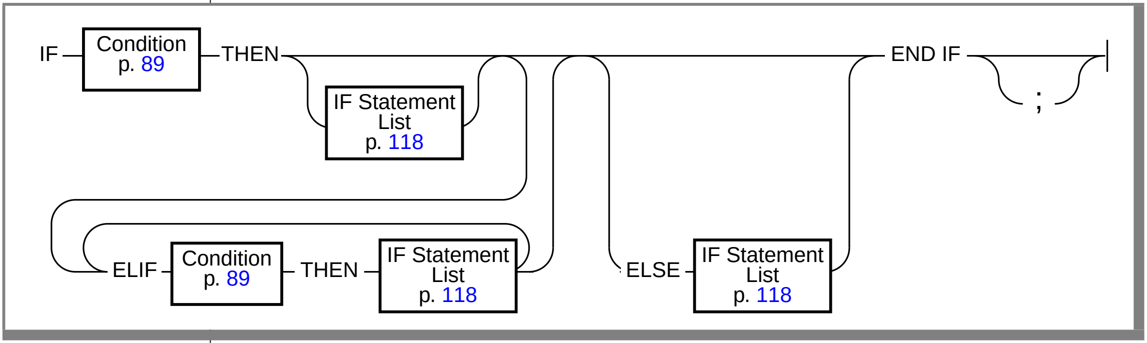
FOR


FOR
variable name
IN
(
,
left expression
TO
right expression
STEP
increment expression
,
expression
)
=
left expression
TO
right expression
STEP
increment expression
Statement Block p. 31
END FOR
;

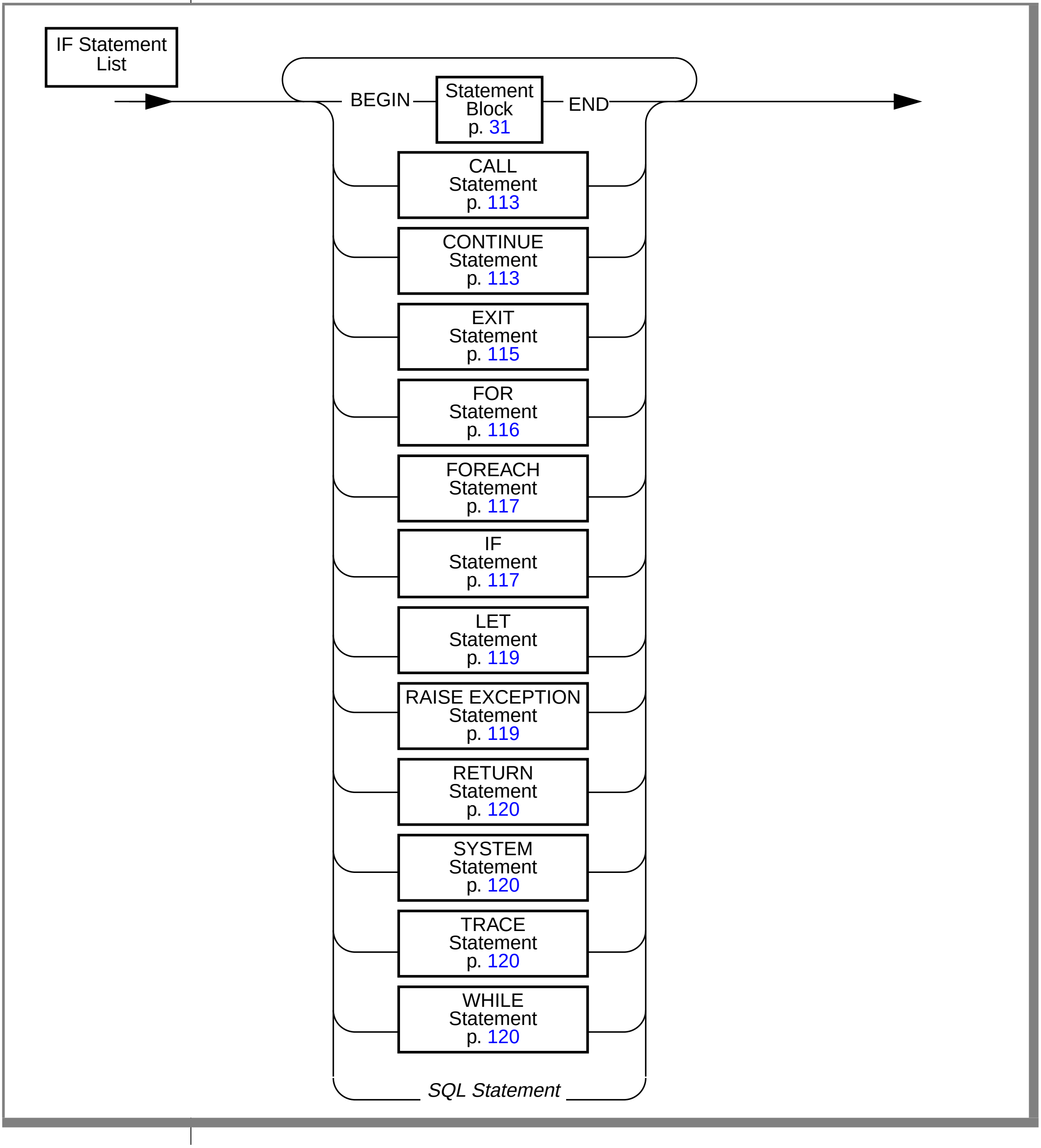
FOREACH


FOREACH
cursor name
WITH HOLD
WITH HOLD
FOR
SELECT...INTO Statement p. 71
Statement Block p. 31
END FOREACH
;
EXECUTE PROCEDURE
Procedure Name p. 107
(
)
INTO
,
variable name
,
parameter name
=
Expression (Subset) p. 96

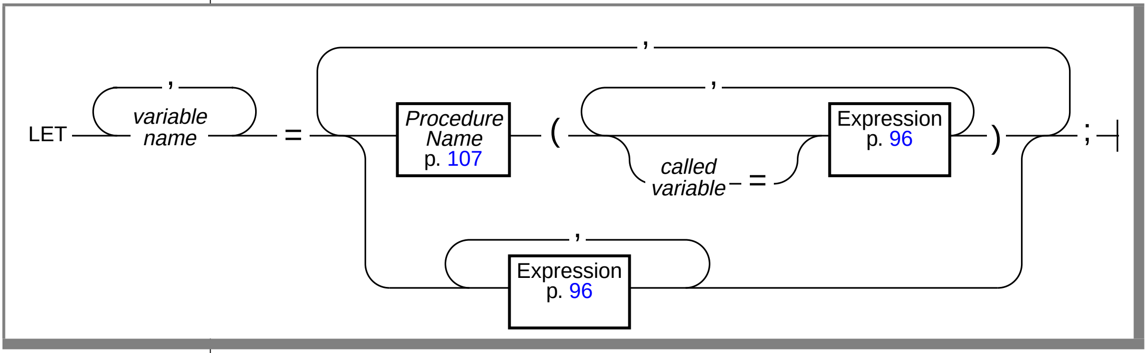
IF


IF
Condition p. 89
THEN
IF Statement List p. 118
END IF
;
ELIF
Condition p. 89
THEN
IF Statement List p. 118
ELSE
IF Statement List p. 118

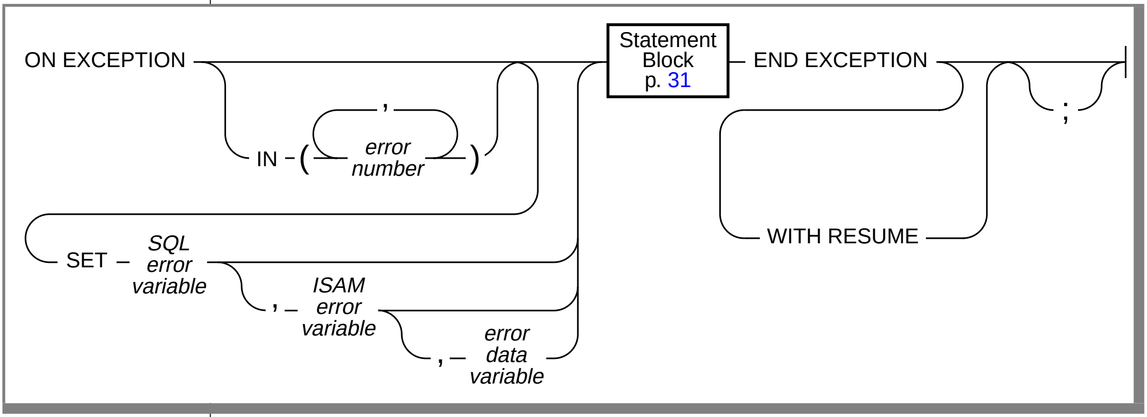
IF Statement List



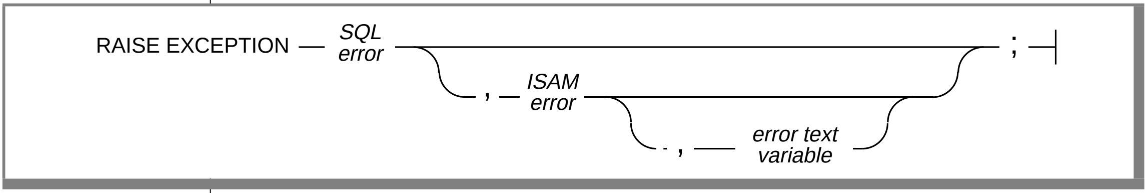
LET



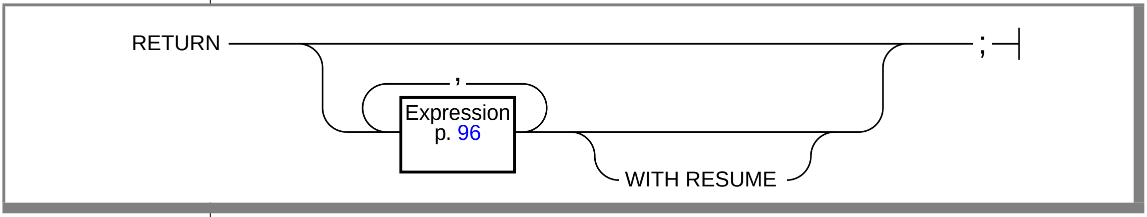
ON EXCEPTION



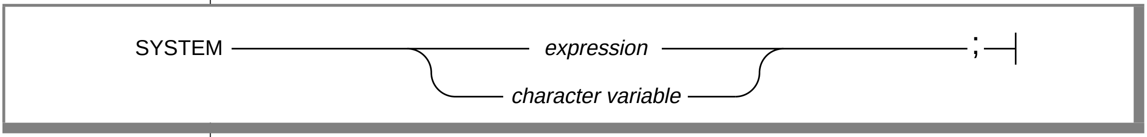
RAISE EXCEPTION



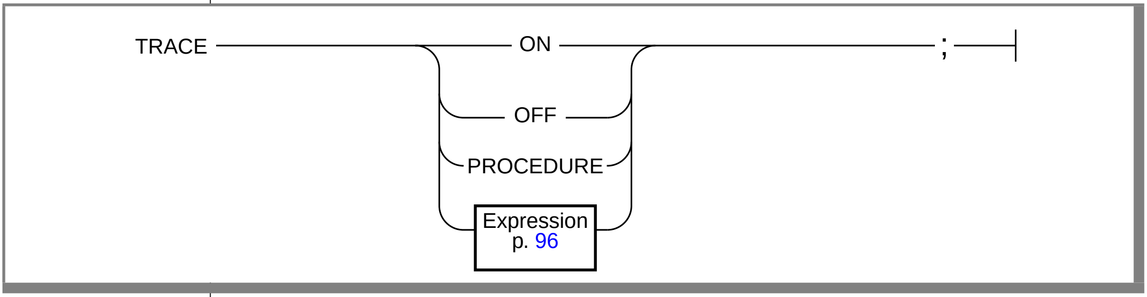
RETURN



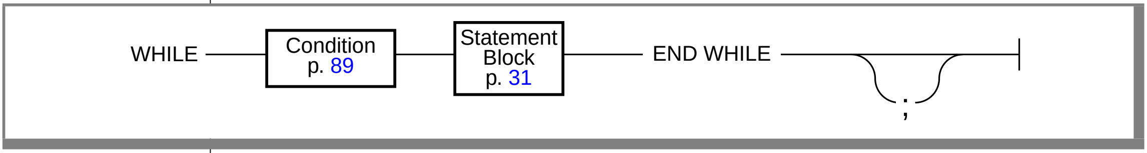
SYSTEM



TRACE

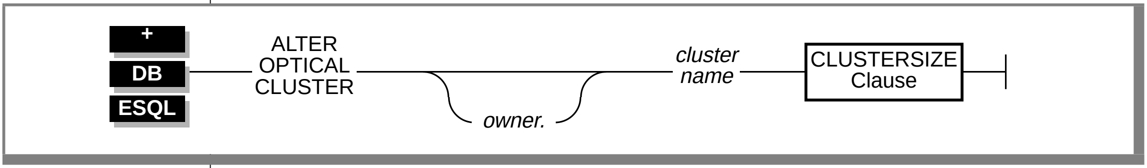


WHILE

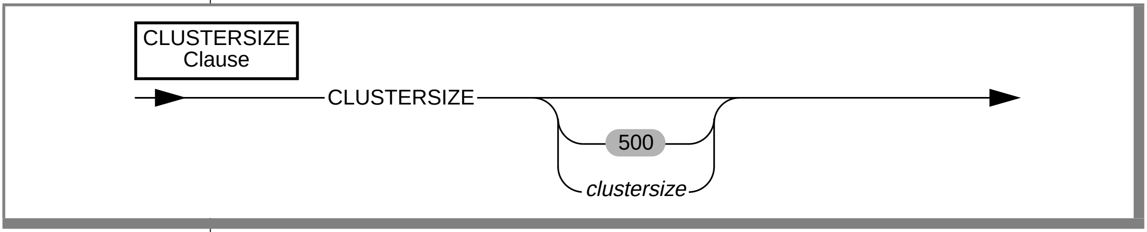


INFORMIX-OnLine Optical Statements

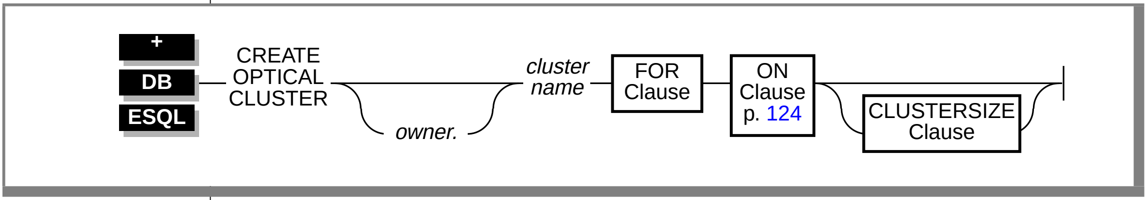
ALTER OPTICAL CLUSTER



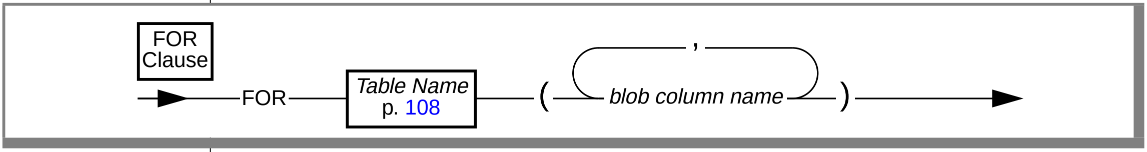
CLUSTERSIZE Clause



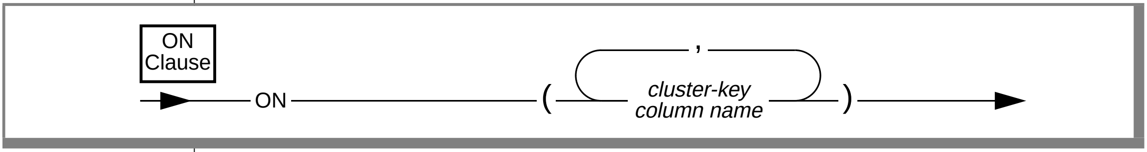
CREATE OPTICAL CLUSTER



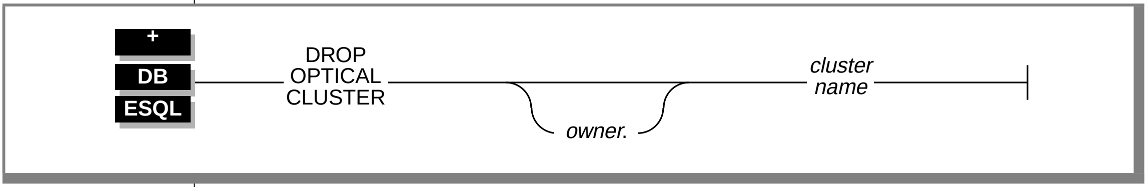
FOR Clause



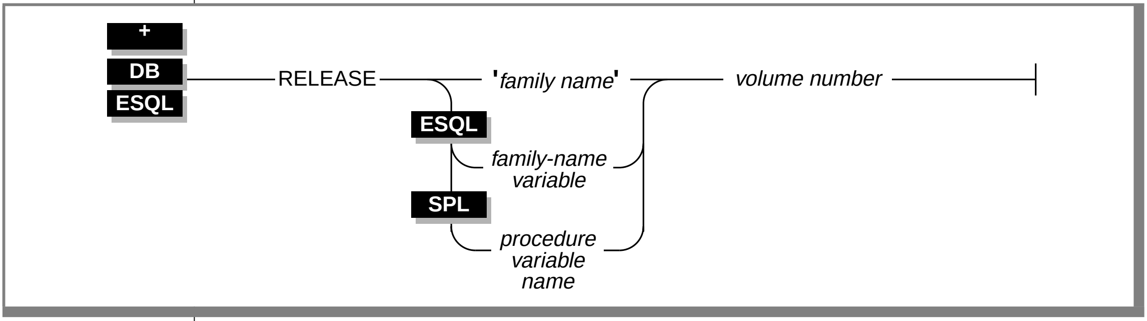
ON Clause



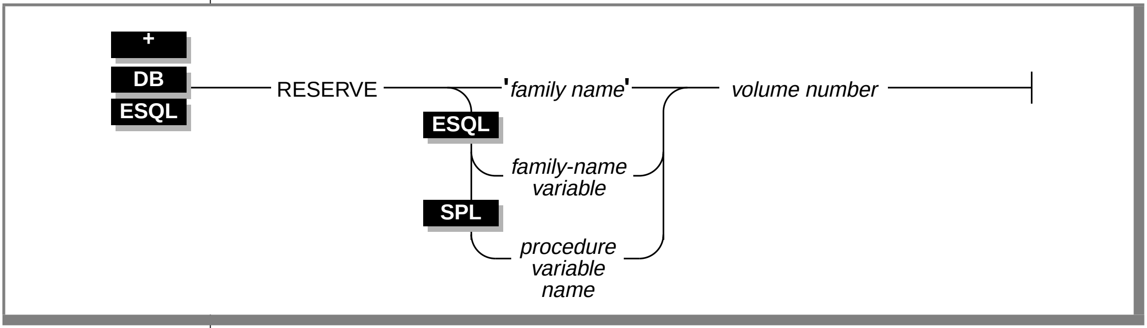
DROP OPTICAL CLUSTER



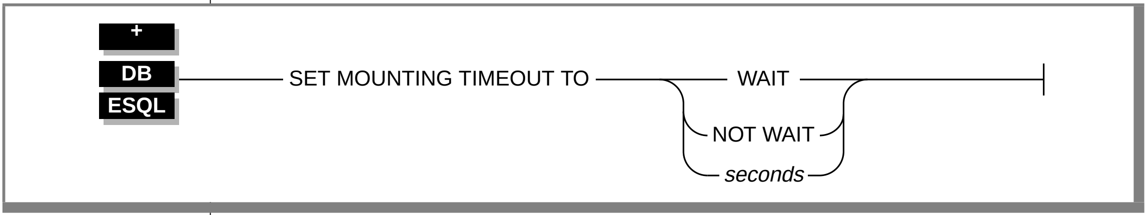
RELEASE



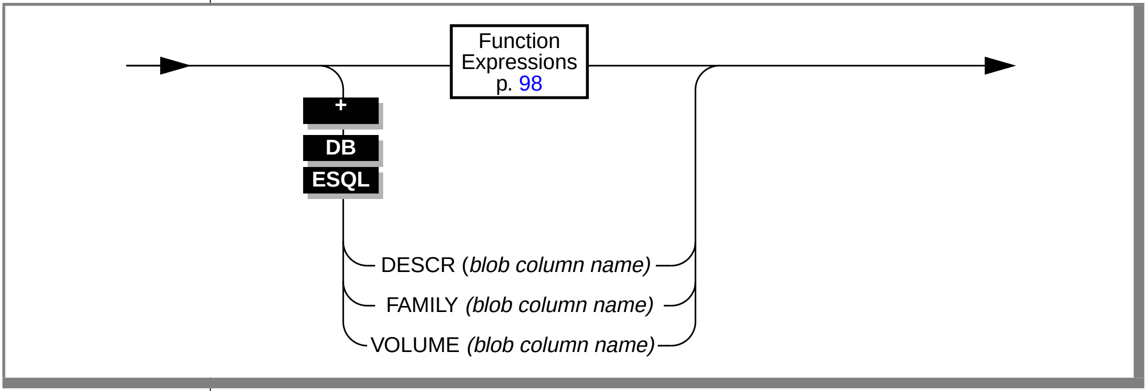
RESERVE



SET MOUNTING TIMEOUT



Function Expressions


Function Expressions p. 98
+
DB
ESQL
DESCR (blob column name)
FAMILY (blob column name)
VOLUME (blob column name)