DB2 Server for VSE & VM

Quick Reference

Version 7 Release 1

SC09-2988-00

DB2 Server for VSE & VM

Quick Reference

Version 7 Release 1

SC09-2988-00

Note!

Before using this information and the product it supports, be sure to read the general information under "Notices" on page 125.

First Edition (September 2000)

This edition applies to Version 7 Release 1 Modification 0 of the IBM® DATABASE 2™ Server for VSE & VM Program, (product number 5697-F42) and to all subsequent releases and modifications until otherwise indicated in new editions.

This edition replaces SC09-2670-00.

Contents

About This Manual

This reference pictorially summarizes Structured Query Language statements used by:
- DB2 Server for VM and DB2 Server for VSE
- Interactive SQL Facility (ISQL) commands
- Database Services Utility (DBS Utility) commands.

It also contains information about the following:
- SQL language elements
- Functions
- Queries
- Preprocessing application programs
- ISQL program function keys
- Operator commands
- Catalog tables
- Application server support for remote applications
- Multiple application server support for DB2 Server for VSE
- SQL communication area (SQLCA) and SQL descriptor area (SQLDA)
- SQL reserved words
- Database Services Utility reserved words.

Who Should Use This Manual

This manual is intended as a quick reference for application developers, system programmers, and database administrators who write application programs using SQL, or use ISQL, or the Database Services Utility in a DB2 Server for VM or DB2 Server for VSE environment. It contains syntax diagrams for SQL statements, ISQL commands, operator commands, and DBS Utility commands.

It is assumed that the VM user has some knowledge of VM (CMS, CP), a programming language, and structured query language (SQL). It is assumed that the VSE user has some knowledge of a VSE system, a CICS/VSE® system or batch as applicable, a programming language, and structured query language (SQL).

Both the VSE and VM user should be familiar with the information in the *DB2 Server for VSE & VM Overivew*. For further information on the required environment, refer to either the *DB2 Server for VM Program Directory* or the *DB2 Server for VSE Program Directory* for your database manager.

Related Publications

For more information about the DB2 Server for VM and DB2 Server for VSE database managers, ISQL, and the DBS Utility, refer to the following IBM publications for DB2 Server for VM or DB2 Server for VSE as appropriate:

- *DB2 Server for VSE & VM Overivew*
- *DB2 Server for VSE & VM SQL Reference*
- *DB2 Server for VSE & VM Interactive SQL Guide and Reference*
- *DB2 Server for VSE & VM Database Services Utility*
- *DB2 Server for VSE & VM Operation*.

Syntax Notation Conventions

Throughout this manual, syntax is described using the structure defined below.

- Read the syntax diagrams from left to right and from top to bottom, following the path of the line.

  The ►►── symbol indicates the beginning of a statement or command.

  The ──► symbol indicates that the statement syntax is continued on the next line.

  The ►── symbol indicates that a statement is continued from the previous line.

  The ──►◄ symbol indicates the end of a statement.

  Diagrams of syntactical units that are not complete statements start with the ►── symbol and end with the ──► symbol.

- Some SQL statements, Interactive SQL (ISQL) commands, or database services utility (DBS Utility) commands can stand alone. For example:

```
►►──SAVE──────────────────────────────────────────────────────────────────►◄
```

  Others must be followed by one or more keywords or variables. For example:

```
►►──SET AUTOCOMMIT OFF────────────────────────────────────────────────────►◄
```

- Keywords may have parameters associated with them which represent user-supplied names or values. These names or values can be specified as either constants or as user-defined variables called *host_variables* (*host_variables* can only be used in programs).

```
►►──DROP SYNONYM──synonym──────────────────────────────────────────────►◄
```

- Keywords appear in either uppercase (for example, SAVE) or mixed case (for example, CHARacter). All uppercase characters in keywords must be present; you can omit those in lowercase.
- Parameters appear in lowercase and in italics (for example, *synonym*).
- If such symbols as punctuation marks, parentheses, or arithmetic operators are shown, you must use them as indicated by the syntax diagram.
- All items (parameters and keywords) must be separated by one or more blanks.
- Required items appear on the same horizontal line (the main path). For example, the parameter *integer* is a required item in the following command:

```
►►──SHOW DBSPACE──integer──────────────────────────────────────────────►◄
```

This command might appear as:

```
SHOW DBSPACE 1
```

- Optional items appear below the main path. For example:

```
►►──CREATE──┬──────────┬──INDEX────────────────────────────────────────►◄
            └─UNIQUE───┘
```

This statement could appear as either:

```
CREATE INDEX
```

or

```
CREATE UNIQUE INDEX
```

- If you can choose from two or more items, they appear vertically in a stack.

  If you must choose one of the items, one item appears on the main path. For example:

```
►►──SHOW LOCK DBSPACE──┬─ALL─────┬─────────────────────────────────────►◄
                       └─integer─┘
```

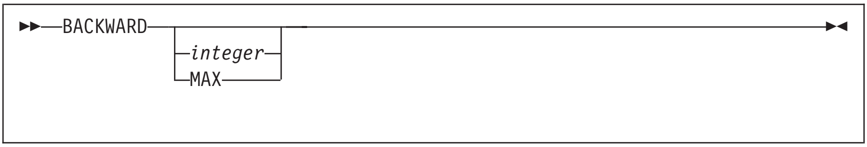
Here, the command could be either:

```
SHOW LOCK DBSPACE ALL
```

or

```
SHOW LOCK DBSPACE 1
```

If choosing one of the items is optional, the entire stack appears below the main path. For example:



Here, the command could be:

```
BACKWARD
```

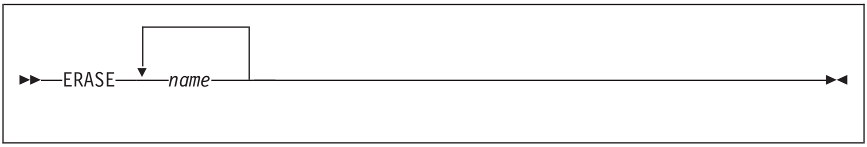
or

```
BACKWARD 2
```

or

```
BACKWARD MAX
```

- The repeat symbol indicates that an item can be repeated. For example:



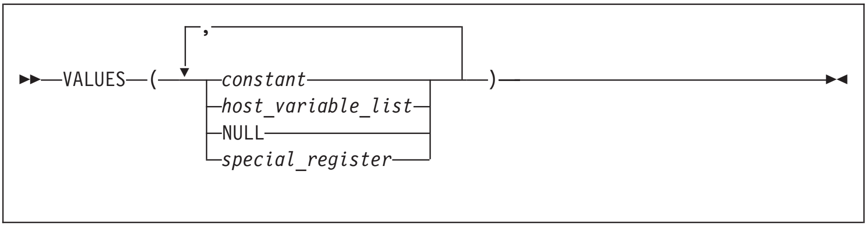
This statement could appear as:

```
ERASE NAME1
```

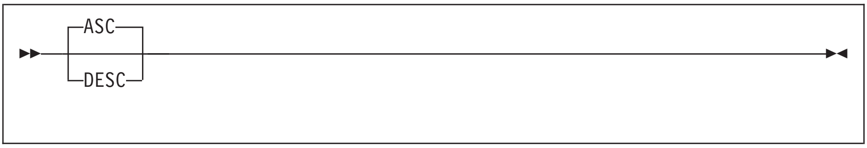
or

```
ERASE NAME1 NAME2
```

A repeat symbol above a stack indicates that you can make more than one choice from the stacked items, or repeat a choice. For example:

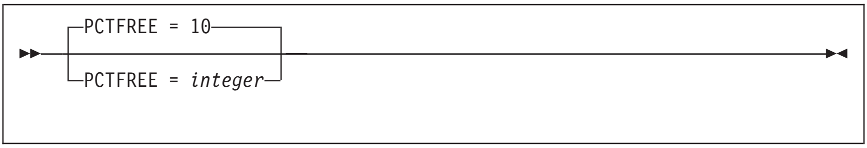


- If an item is above the main line, it represents a default, which means that it will be used if no other item is specified. In the following example, the ASC keyword appears above the line in a stack with DESC. If neither of these values is specified, the command would be processed with option ASC.

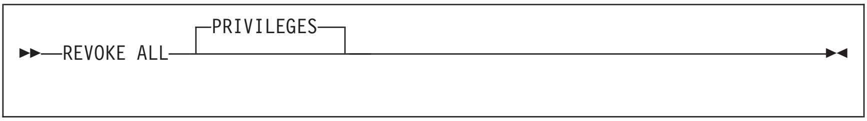


- When an optional keyword is followed on the same path by an optional default parameter, the default parameter is assumed if the keyword is not entered. However, if this keyword is entered, one of its associated optional parameters must also be specified.

  In the following example, if you enter the optional keyword PCTFREE =, you also have to specify one of its associated optional parameters. If you do not enter PCTFREE =, the database manager will set it to the default value of 10.

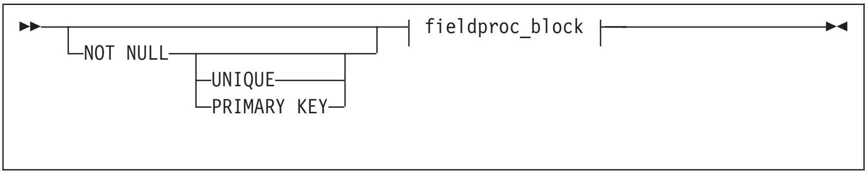


- Words that are only used for readability and have no effect on the execution of the statement are shown as a single uppercase default. For example:

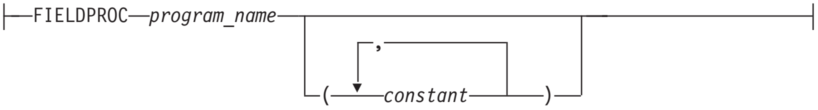


Here, specifying either REVOKE ALL or REVOKE ALL PRIVILEGES means the same thing.

- Sometimes a single parameter represents a fragment of syntax that is expanded below. In the following example, **fieldproc_block** is such a fragment and it is expanded following the syntax diagram containing it.



fieldproc_block:



How to Send Your Comments

Your feedback is important in helping to provide the most accurate and high-quality information. If you have any comments about this book or any other DB2 Server for VSE & VM documentation:

- Visit our home page at:

  `http://www-4.ibm.com/software/data/db2/vse-vm/`

- A form for readers' comments is provided at the back of this publication. If the form has been removed, address your comments to:

  IBM CANADA LTD.
  DB2 Server for VSE & VM
  2S/240/1150/TOR
  1150 Eglinton Ave. East
  North York, Ontario
  Canada M3C 1H7

- Send your comments by electronic mail to one of the following addresses:

Format	Address
Internet	torrcf@ca.ibm.com
Facsimile	(416) 448-6161 (Attention RCF Coordinator)

Be sure to include the name of the book, the form number (including the suffix), and the page, section title, or topic you are commenting on.

If you choose to respond through the Internet, please include either your entire Internet network address, or a postal address.

- Fill out the form at the back of this book and return it by mail, by fax, or by giving it to an IBM representative.

Conventions for Representing Mixed Data Values

When mixed data values are shown in examples, the following conventions are used:

Convention	Meaning
<	Represents the mixed shift-out character (X'0E').
>	Represents the mixed shift-in character (X'0F').
x	Represents an SBCS character (x can be any lowercase letter).
XX	Represents a DBCS character (XX can be any double-byte uppercase letter).

Short Forms Used in Syntax Diagrams

Some words have been shortened in the syntax diagrams in this book. The words are:

Full Word	Short Form
duration	dur
expressions	exp
string	str

Chapter 1. DB2 Language Elements

Primitive Elements

character	A letter, digit, space, or special-character
letter	The letters a to z, A to Z, or national language extender (# @ $), or as specified in SYSCHARSETS
digit	The digits 0 to 9
space	The space character
special-character	Any element in a character set other than a letter, digit, or space
hexadecimal-character	A pair of characters in the range 00 to FF
double-byte-character	A character that occupies 2 bytes.

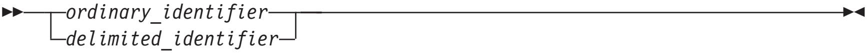
SQL Comments

An SQL comment is all text following two consecutive hyphens (--) on the same line of a static SQL statement in an application program or the command portion of a DBS Utility command.

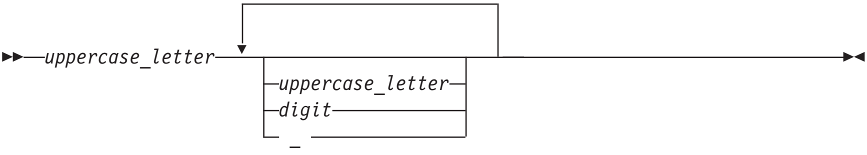
Comments are allowed wherever a separator (space character) is valid.

Identifiers

identifier



ordinary_identifier



Notes:

1 A reserved word cannot be used as an ordinary identifier. For a list of SQL reserved words, see “Chapter 12. SQL Reserved Words” on page 119.

delimited_identifier

```
►►─"─non_space_character──────────────────────────────────────────────►

►─┬──────────────────────────────────────────────┬─"─────────────────►◄
  │   ┌───────────────┐                          │
  │   ▼           (1) │                          │
  └───character───────┴───non_space_character────┘
```

Notes:

1 With the exception of ".

long_identifier An identifier with a maximum length of 18 characters (not including any quotation marks).

short_identifier An identifier with a maximum length of 8 characters (not including any quotation marks).

host_identifier As defined by the host language, has a maximum length imposed by the host language.

Names and Other Metavariables

A metavariable (or parameter) is a lowercase character or group of characters used in syntax diagrams to represent a group of variables.

authorization_name

With a VSE system, authorization names and passwords are limited to 8 characters and cannot have embedded blanks.

```
►►─short_identifier───────────────────────────────────────────────────►◄
```

collection_id

```
►►─short_identifier───────────────────────────────────────────────────►◄
```

column_name

```
►►─┬───────────────────┬─long_identifier─────────────────────────►◄
   ├─table_name.───────┤
   ├─view_name.────────┤
   ├─synonym.──────────┤
   └─correlation_name.─┘
```

constraint_name

```
►►─long_identifier───────────────────────────────────────────────►◄
```

correlation_name

```
►►─long_identifier───────────────────────────────────────────────►◄
```

cursor_name

```
►►─long_ordinary_identifier──────────────────────────────────────►◄
```

cursor_variable

```
►►─long_ordinary_identifier──────────────────────────────────────►◄
```

dbspace_name

```
►►─┬────────┬─short_ordinary_identifier──────────────────────────►◄
   └─owner.─┘
```

descriptor_name

```
►►─:host_identifier──────────────────────────────────────────────►◄
```

host_variable

```
►►─:host_identifier─┬────────────────────────────────┬───────────►◄
                    │ ┌─INDICATOR─┐                  │
                    └─┴───────────┴─:host_identifier─┘
```

host_variable_list

```
      ┌─,─────────────┐
      ▼               │
►►────host_variable───┴──────────────────────────────────────►◄
```

index_id

```
►►─short_ordinary_identifier─────────────────────────────────►◄
```

index_name

```
►►─┬────────┬─long_identifier────────────────────────────────►◄
   └─owner.─┘
```

owner_name

```
►►─short_ordinary_identifier─────────────────────────────────►◄
```

package_id

```
►►─short_ordinary_identifier─────────────────────────────────►◄
```

package_name

```
►►─┬────────┬─long_identifier────────────────────────────────►◄
   └─owner.─┘
```

package_spec

```
►►─┬─────────────────────────────┬─┬─short_ordinary_identifier─┬──►◄
   ├─short_ordinary_identifier.──┤ │                (1)        │
   └─host_identifier.────────────┘ └─host_identifier───────────┘
```

Notes:

1 Cannot be a qualified subfield name.

password

```
►►─short_ordinary_identifier─────────────────────────────────►◄
```

program_name

```
►►──short_ordinary_identifier──────────────────────────────────────────────►◄
```

routine_name

```
►►──┬────────┬──long_identifier────────────────────────────────────────────►◄
    └─owner.─┘
```

section_variable

```
►►──host_identifier────────────────────────────────────────────────────────►◄
```

server_name

```
►►──long_ordinary_name─────────────────────────────────────────────────────►◄
```

statement_name

```
►►──long_ordinary_identifier───────────────────────────────────────────────►◄
```

statement_variable

```
►►──long_ordinary_identifier───────────────────────────────────────────────►◄
```

subsystemid

```
►►──short_ordinary_identifier──────────────────────────────────────────────►◄
```

synonym

```
►►──┬────────┬──long_identifier────────────────────────────────────────────►◄
    └─owner.─┘
```

table_id

```
►►──short_ordinary_identifier──────────────────────────────────────────────►◄
```

table_name

```
►►──┬────────┬──long_identifier──────────────────────────────────►◄
    └─owner.─┘
```

view_id

```
►►──short_ordinary_identifier────────────────────────────────────►◄
```

view_name

```
►►──┬────────┬──long_identifier──────────────────────────────────►◄
    └─owner.─┘
```

Data Types

Result Set LOCATOR

For RESULT SET LOCATOR data. This data type is used to identify host variables that are used by the DB2 Server for VSE & VM requester to uniquely indicate a query result set returned by a stored procedure.

RESULT SET LOCATOR

```
├──┬─┤ Assembler ├─┬─────────────────────────────────────────────┤
   ├─┤ C ├─────────┤
   ├─┤ COBOL ├─────┤
   └─┤ PL/I ├──────┘
```

Assembler:

```
├──variable-name──┬─DC─┬──F──────────────────────────────────────┤
                  └─DS─┘
```

C:

```
├──┬──────────┬──┬──────────┬────────────────────────────────────►
   ├─auto─────┤  ├─const────┤
   ├─extern───┤  └─volatile─┘
   ├─static───┤
   └─_Packed──┘
```

```
                                        ┌,──────────────────────────┐
                                        ↓                           │
►─SQL TYPE IS RESULT_SET_LOCATOR VARYING───variable-name─┬──────────────┬┴──;─┤
                                                         └= init-value─┘
```

COBOL:

```
├──01──variable-name──SQL TYPE IS RESULT-SET-LOCATOR VARYING──.──────────────┤
```

PL/I:

```
├─┬DECLARE─┬──┬variable-name──────────────────┬────────────────────────────►
  └DCL─────┘  │    ┌,─────────────────┐       │
              │    ↓                  │       │
              └(───variable-name──────┴──)────┘

►─SQL TYPE IS RESULT_SET_LOCATOR VARYING─┬─────────────────────────────────────────┬──;─┤
                                         └Alignment and/or Scope and/or Storage──┘
```

CHARacter

For character data that has a fixed number of characters (*integers*). The maximum number of characters is 254.

```
            ┌─(1)──────┐
►►──CHARacter┼──────────┼──────────────────────────────────────────────►◄
            └─(integer)┘
```

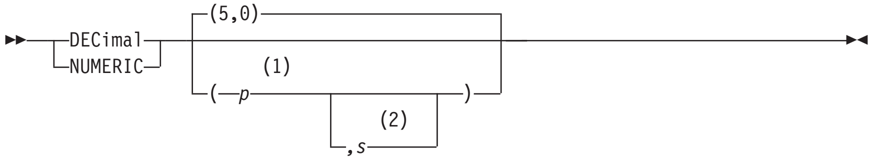
DATE

A three-part value that designates a point in time according to the Gregorian calendar. Internally represented as 4-byte packed decimal. The three parts are the year, month, and day. The date can be formatted in several ways. The range of year is 0001 to 9999. The range of month is 1 to 12. The range of day is 1 to n where n depends on the month.

```
►►──DATE───────────────────────────────────────────────────────────────►◄
```

DECimal

For decimal data. The *p* identifies the total number of decimal digits a number can have. The *s* identifies the number of digits to the right of the decimal point. For example, DECIMAL(5,2) creates a decimal column consisting of five digits, two of which are to the right of the decimal point. The NUMERIC parameter is a synonym for DECIMAL.



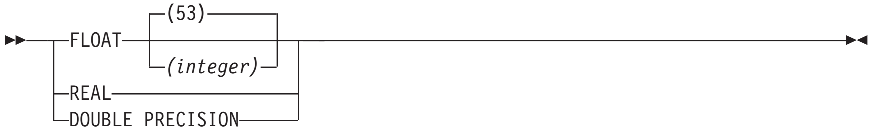
Notes:

1 The ***p*** is an integer value that defines the precision of the number.

2 The*s* is an integer value that defines the scale of the number.

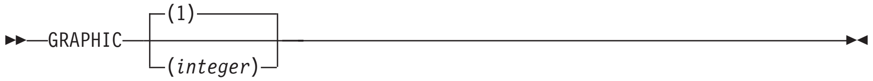
FLOAT

For floating-point numbers. Floating-point numbers range from 5.4E−79 to 7.2E+75. When *integer* is between 1 and 21, it is a single-precision floating-point number; REAL is a synonym for FLOAT in this situation. When *integer* is between 22 and 53, it is a double-precision floating-point number; DOUBLE PRECISION is a synonym for FLOAT in this situation.



GRAPHIC

For double-byte character set (DBCS) data that has a fixed number of DBCS characters *(integer)*. The maximum number of DBCS characters is 127.



INTeger

For large positive or negative whole numbers. The largest number that can be accommodated is 2147483647; the smallest number is −2147483648.

```
►►──INTeger──────────────────────────────────────────────────────────►◄
```

LONG VARCHAR

For character data that varies in length up to 32,767 characters.

```
                  (1)
►►──LONG VARCHAR──────────────────────────────────────────────────────►◄
```

Notes:

1 ISQL does not support INSERT, UPDATE, or SELECT of tables or views with LONG VARCHAR columns.

LONG VARGRAPHIC

For double-byte character set (DBCS) data that varies in length. A LONG VARGRAPHIC can be up to a maximum of 16,383 DBCS characters.

```
                     (1)
►►──LONG VARGRAPHIC───────────────────────────────────────────────────►◄
```

Notes:

1 ISQL does not support INSERT, UPDATE, or SELECT of tables or views with LONG VARGRAPHIC columns.

SMALLINT

For small positive or negative whole numbers. The largest number that can be accommodated is 32767; the smallest is −32768.

```
►►──SMALLINT─────────────────────────────────────────────────────────►◄
```

TIME

A three-part value in a number of formats that designates a time of day according to a 24-hour clock. Internally represented as 3-byte packed decimal. The three parts are the hour, minute, and second. The range of hour is 0 to 24, and the range of minute and second is 0 to 59.

```
►►──TIME─────────────────────────────────────────────────────────────►◄
```

TIMESTAMP

A seven-part value that designates a date and time, including a fractional part. Internally represented as 10-byte packed decimal. The seven parts are year, month, day, hour, minute, second, and microsecond.

```
►►──TIMESTAMP──────────────────────────────────────────────────────────►◄
```

VARCHAR

For character data that varies in length. The *integer* refers to the maximum number of characters for any entry and can be a value up to 32767. When the value is greater than 254, the data type is considered a long string.

```
                (1)
►►──VARCHAR────────(integer)───────────────────────────────────────────►◄
```

Notes:

1 ISQL does not support INSERT, UPDATE, or SELECT of tables or views with VARCHAR>254.

VARGRAPHIC

For double-byte character set (DBCS) data that varies in length. The *integer* is the number of DBCS characters for any entry; the maximum is 16383. When *integer* is greater than 127, the data type is considered a long string.

```
                   (1)
►►──VARGRAPHIC────────(integer)────────────────────────────────────────►◄
```

Notes:

1 ISQL does not support INSERT, UPDATE, or SELECT of tables or views with VARGRAPHIC>127.

String Representations of Dates and Times

Date Strings

A string representation of a date is a character string that starts with a digit and has a length of at least 8 characters.

Format Name	Abbrev.	Date Format	Example
International Standards Organization	ISO	yyyy-mm-dd	1993-12-12
IBM USA standard	USA	mm/dd/yyyy	12/12/1993
IBM European standard	EUR	dd.mm.yyyy	12.12.1993
Japanese Industrial Standard Christian Era	JIS	yyyy-mm-dd	1993-12-12
Site-defined	LOCAL	Any site-defined form	—

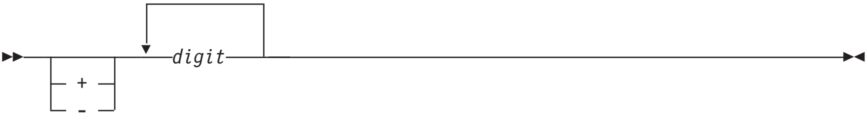
Time Strings

A string representation of a time is a character string that starts with a digit and has a length of at least 4 characters.

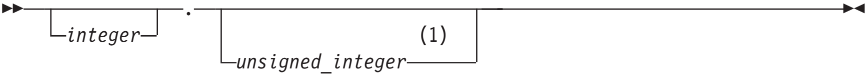
Format Name	Abbrev.	Time Format	Example
International Standards Organization	ISO	hh.mm.ss	13.30.05
IBM USA standard	USA	hh:mm AM or PM	1:30 PM
IBM European standard	EUR	hh.mm.ss	13.30.05
Japanese Industrial Standard Christian Era	JIS	hh:mm:ss	13:30:05
Site-defined	LOCAL	Any site-defined form	—

Constants

Integer Constant



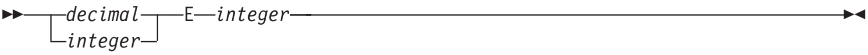
Decimal Constant



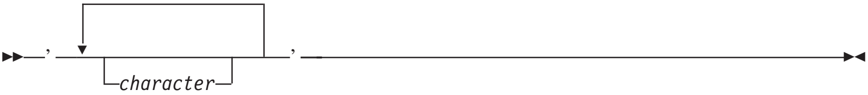
Notes:

1 At least one number is needed with the decimal point.

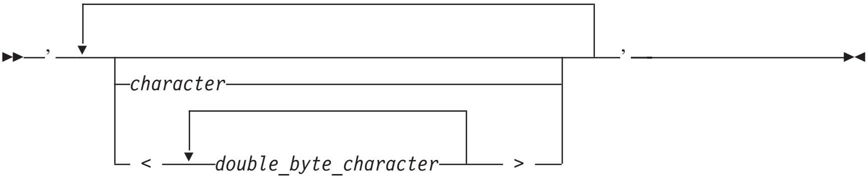
Floating-Point Constant



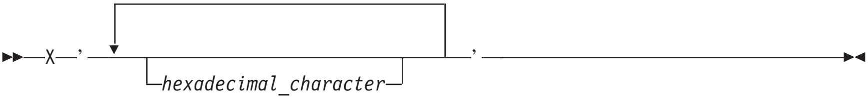
Character Constant - SBCS



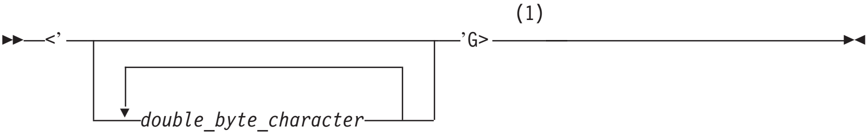
Character Constant - MIXED



Character Constant - Hexadecimal



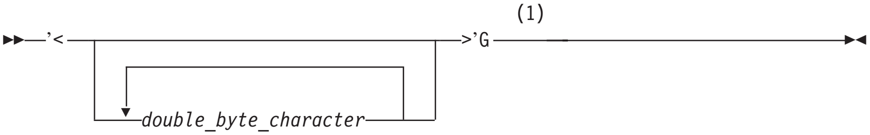
Graphic Constant - in PL/I Programs



Notes:

1 N is a synonym for G.

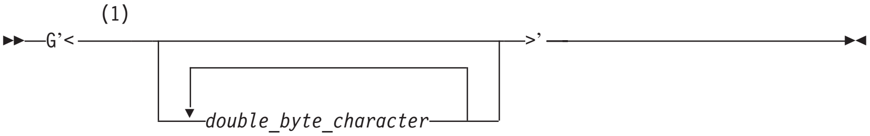
or



Notes:

1 N is a synonym for G.

Graphic Constant - In All Other Contexts



Notes:

1 N is a synonym for G.

Special Registers

The following special registers are supported by the database manager.

Special Registers	Description
USER	The runtime authorization ID
CURRENT DATE	The current date in the local time zone
CURRENT SERVER	The current application server
CURRENT TIME	The current time in the local time zone
CURRENT TIMESTAMP	The current timestamp in the local time zone
CURRENT TIMEZONE	A signed time duration as a DECIMAL(6,0) number containing the local time-zone value.

Expressions

An expression specifies a value. The form of an expression is as follows:

```
   ┌─| operator |──────────────────────────────────────┐
   ▼                                          (1)      │
►►─┴─┬───────┬──┬─column_name───────────────────┬───────┴────────────────────────►◄
     ├─  +  ─┤  ├─constant──────────────────────┤
     └─  -  ─┘  ├─(expression)──────────────────┤
                ├─function──────────────────────┤
                ├─host_variable─────────────────┤
                ├─| labeled_duration |──────────┤
                └─special_register──────────────┘
```

operator:

```
             (2)
├──┬─ CONCAT ──────┬──────────────────────────────────────────────────┤
   ├─ / ───────────┤
   ├─ * ───────────┤
   ├─ + ───────────┤
   └─ - ───────────┘
```

Notes:

1 Not all combinations of operands and operations are supported.

2 Either || or !! can be used as a synonym for CONCAT.

labeled_duration:

```
├──┬─column_name───┬──┬─DAY──────────┬──────────────────────────────────┤
   ├─constant──────┤  ├─DAYS─────────┤
   ├─(expression)──┤  ├─HOUR─────────┤
   ├─function──────┤  ├─HOURS────────┤
   └─host_variable─┘  ├─MICROSECOND──┤
                      ├─MICROSECONDS─┤
                      ├─MINUTE───────┤
                      ├─MINUTES──────┤
                      ├─MONTH────────┤
                      ├─MONTHS───────┤
                      ├─SECOND───────┤
                      ├─SECONDS──────┤
                      ├─YEAR─────────┤
                      └─YEARS────────┘
```

Date Arithmetic

```
          (1)                                      (1)
►►──date──────────┬─ + ─┬───┬─date_dur─────────────┬────────── = ──date──────────────────────►◄
                  └─ - ─┘   └─labeled_date_dur─────┘
```

Notes:

1 These operands can be specified in either order.

```
►►──┬─date──────────────┬── - ──┬─date──────────────┬── = ──date_dur──────────────────────►◄
    │              (1)  │       │              (1)  │
    └─date_str──────────┘       └─date_str──────────┘
```

Notes:

1 Only one of these two operands can be a string.

Time Arithmetic

```
          (1)                                      (1)
►►──time──────────┬─ + ─┬───┬─time_dur─────────────┬────────── = ──time──────────────────────►◄
                  └─ - ─┘   └─labeled_time_dur─────┘
```

Notes:

1 These operands can be specified in either order.

```
►►──┬─time──────────────┬── - ──┬─time──────────────┬── = ──time_dur──────────────────────►◄
    │              (1)  │       │              (1)  │
    └─time_str──────────┘       └─time_str──────────┘
```

Notes:

1 Only one of these two operands can be a string.

Timestamp Arithmetic

```
               (1)                                     (1)
►►──timestamp────────┬─ + ─┬───┬─date_dur──────────┬────────── = ──timestamp──────────────►◄
                     └─ - ─┘   ├─labeled_dur───────┤
                               ├─time_dur──────────┤
                               └─timestamp_dur─────┘
```

Notes:

1 These operands can be specified in either order.

```
►►──┬─timestamp──────────┬── - ──┬─timestamp──────────┬── = ──timestamp_dur──►◄
    │               (1)  │       │               (1)  │
    └─timestamp_str──────┘       └─timestamp_str──────┘
```

Notes:

1 Only one of these two operands can be a string.

Predicate

Specifies a condition that is true, false, or unknown about a row or group.

Basic Predicate

```
►►──exp──┬─ = ───────┬──┬─exp──────────┬──────────────────────────────►◄
         │     (1)   │  └─(subselect)──┘
         ├─ <> ──────┤
         ├─ < ───────┤
         ├─ > ───────┤
         ├─ <= ──────┤
         └─ >= ──────┘
```

Notes:

1 Either ˆ= or ˆ= can be used as an alternative to the <> operator.

BETWEEN Predicate

```
►►──exp──┬───────┬──BETWEEN──exp──AND──exp──────────────────────────►◄
         └─NOT───┘
```

EXISTS Predicate

```
►►──┬───────┬──EXISTS──(subselect)──────────────────────────────────►◄
    └─NOT───┘
```

IN Predicate

```
►►──exp──┬─────┬──IN──┬─(subselect)──────────────────────────┬──────────────►◄
         └─NOT─┘      │      ┌─,───────────────────────┐     │
                      │      ▼                         │     │
                      └─(────┬─constant──────────┬─────┴──)──┘
                             ├─host_variable_list─┤
                             └─special_register───┘
```

LIKE Predicate

```
►►──column_name──┬─────┬──LIKE──┬─USER────────────┬──────────────────────────►
                 └─NOT─┘        ├─host_variable───┤
                                └─str_constant────┘

►──┬──────────────────────────────────────────┬──────────────────────────────►◄
   └─ESCAPE──┬─host_variable──────────┬───────┘
             └─character_constant─────┘
```

NULL Predicate

```
►►──column_name──IS──┬─────┬──NULL───────────────────────────────────────────►◄
                     └─NOT─┘
```

Quantified Predicate

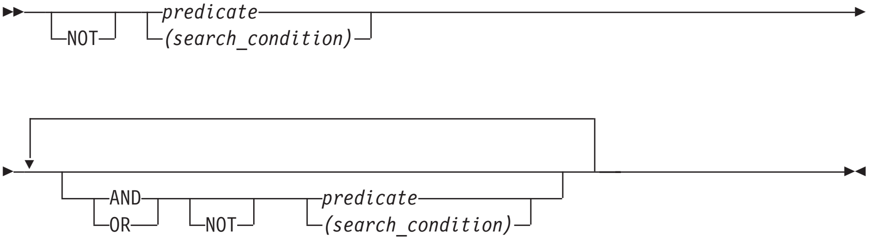
```
►►──exp──┬─ =  ───────┬──┬─SOME─┬──(subselect)───────────────────────────────►◄
         │     (1)    │  ├─ANY──┤
         ├─ <> ───────┤  └─ALL──┘
         ├─ <  ───────┤
         ├─ >  ───────┤
         ├─ <= ───────┤
         └─ >= ───────┘
```

Notes:

1 Either ˆ= or ˆ= can be used as an alternative to the <> operator.

Search Conditions

Specifies a condition that is true, false, or unknown about a row or group. The common form of a search condition is column_name operator value. Refer to “Predicate” on page 16 for additional functions available for search conditions.


NOT
predicate
(search_condition)
AND
OR
NOT
predicate
(search_condition)

Chapter 2. Functions

The two kinds of functions are column and scalar.

Column Functions

Produce a value from an argument having a collection of values that are derived from one or more columns.

AVG

Calculates the average of a group of column values.

```
             ┌─ALL─┐
►►──AVG──(──┬┴─────┴──numeric_exp──────┬──)───────────────────────►◄
            └─DISTINCT──column_name────┘
```

COUNT

Obtains the number of rows or distinct values in a collection of rows or column values.

```
►►──COUNT──┬─(──DISTINCT──column_name──)─┬────────────────────────►◄
           └─(*)─────────────────────────┘
```

MAX

Obtains the maximum value in a collection of column values.

```
             ┌─ALL─┐
►►──MAX──(──┬┴─────┴──exp──────────────────┬──)───────────────────►◄
            │               (1)            │
            └─DISTINCT──────────column_name┘
```

Notes:

1 Although the keyword DISTINCT is allowed, it does not affect the result of the function.

MIN

Obtains the minimum value in a set of column values.

```
           ┌─ALL─┐
►►─MIN─(─┬─┴─────┴─exp───────────────┬─)──────────────────────────►◄
         │          (1)              │
         └─DISTINCT──────column_name─┘
```

Notes:

1 Although the keyword DISTINCT is allowed, it does not affect the result of the function.

SUM

Obtains the total of all values in a group.

```
           ┌─ALL─┐
►►─SUM─(─┬─┴─────┴─numeric_exp──┬─)───────────────────────────────►◄
         └─DISTINCT─column_name─┘
```

Scalar Functions

Produce a single value from an argument having a single value.

CHAR

Creates a character representation of certain noncharacter data types.

```
►►─CHAR─(─┬─decimal_exp──────────────────┬─)──────────────────────►◄
          ├─timestamp_exp────────────────┤
          └─┬─date_exp─┬─┬──────────┬────┘
            └─time_exp─┘ ├─,ISO─────┤
                         ├─,USA─────┤
                         ├─,EUR─────┤
                         ├─,JIS─────┤
                         └─,LOCAL───┘
```

DATE

Creates a date from an expression or timestamp.

```
►►─DATE─(exp)─────────────────────────────────────────────────────►◄
```

DAY

Extracts the day part of a value.

```
►►─DAY─(─┬─date_exp──────────┬─)──────────────────────────────────────►◄
         ├─timestamp_exp─────┤
         ├─date_dur_exp──────┤
         └─timestamp_dur_exp─┘
```

DAYS

Extracts an integer representation of a date. The result is one more than the number of days from January 1, 0001, to the date.

```
►►─DAYS─(─┬─date_exp──────┬─)─────────────────────────────────────────►◄
          ├─timestamp_exp─┤
          └─date_str_exp──┘
```

DECIMAL

Returns a decimal representation of a numeric value.

```
►►─DECIMAL─(─numeric_exp─┬───────────────────────────────────────────┬─)─►◄
                         │                      ┌─,0──────────────┐  │
                         └─,precision_integer───┼─────────────────┼──┘
                                                └─,scale_integer──┘
```

DIGITS

Returns a character string representation of a number without a sign or decimal point.

```
►►─DIGITS─(─┬─integer_exp─┬─)─────────────────────────────────────────►◄
            └─decimal_exp─┘
```

FLOAT

Returns a floating-point representation of a number.

```
►►─FLOAT─(numeric_exp)────────────────────────────────────────────────►◄
```

HEX

Returns a hexadecimal representation of a value.

```
►►─HEX─(exp)──────────────────────────────────────────────────────────►◄
```

HOUR

Extracts the hour part of a value.

```
►►──HOUR──(──┬─time_exp────────────┬─)──────────────────────────────────►◄
             ├─timestamp_exp───────┤
             ├─time_dur_exp────────┤
             └─timestamp_dur_exp───┘
```

INTEGER

Returns an integer representation of a number.

```
►►──INTEGER──(numeric_exp)──────────────────────────────────────────────►◄
```

LENGTH

Returns the length of a value.

```
►►──LENGTH──(exp)───────────────────────────────────────────────────────►◄
```

MICROSECOND

Extracts the microsecond part of a value.

```
►►──MICROSECOND──(──┬─timestamp_exp───────┬─)───────────────────────────►◄
                    └─timestamp_dur_exp───┘
```

MINUTE

Extracts the minute part of a value.

```
►►──MINUTE──(──┬─time_exp────────────┬─)────────────────────────────────►◄
               ├─timestamp_exp───────┤
               ├─time_dur_exp────────┤
               └─timestamp_dur_exp───┘
```

MONTH

Extracts the month part of a value.

```
►►──MONTH──(──┬─date_exp────────────┬─)─────────────────────────────────►◄
              ├─timestamp_exp───────┤
              ├─date_dur_exp────────┤
              └─timestamp_dur_exp───┘
```

SECOND

Extracts the seconds part of a value.

```
►►──SECOND──(──┬─time_exp──────────┬──)──────────────────────────────►◄
               ├─timestamp_exp─────┤
               ├─time_dur_exp──────┤
               └─timestamp_dur_exp─┘
```

STRIP

Removes blanks or another specified character from the end or the beginning of a string.

```
                                ┌─,B──,' '──────────────────────────┐
►►──STRIP──(──┬─character_str───┼───────────────────────────────────┼─┬─)──────►◄
              │                 │                 ┌─,' '───────────┐ │ │
              │                 └─┬─,L─┬──────────┼────────────────┼─┘ │
              │                   ├─,T─┤          └─,strip_character─┘ │
              │                   └─,B─┘                               │
              │               ┌─,B──,'G< >'───────────────────────┐    │
              └─graphic_str───┼───────────────────────────────────┼────┘
                              │                 ┌─,'G< >'────────┐ │
                              └─┬─,L─┬──────────┼────────────────┼─┘
                                ├─,T─┤          └─,strip_character─┘
                                └─,B─┘
```

SUBSTR

Returns a part (substring) of string as indicated by the length and starting position.

```
►►──SUBSTR──(──str_exp,──start_integer_exp──┬─────────────────────┬──)──────►◄
                                            └─,length_integer_exp─┘
```

TIME

Creates a time from a value.

```
►►──TIME──(──┬─time_exp──────┬──)──────────────────────────────────►◄
             ├─timestamp_exp─┤
             └─time_str_exp──┘
```

TIMESTAMP

Creates a timestamp from a value or a pair of values that represent a date and time.

```
►►──TIMESTAMP──(──┬─┬─timestamp_exp─────────────────┬─────────┬──)──────────────►◄
                  │ ├─timestamp_str_exp─────────────┤         │
                  │ ├─yyyymmddhhmmss_character_exp──┤         │
                  │ └─370_storeclock_character_exp──┘         │
                  └─┬─date_exp─────┬──┬─,time_exp─────┬───────┘
                    └─date_str_exp─┘  └─,time_str_exp─┘
```

TRANSLATE

Changes one or more characters in a string expression into other characters. For example, it can be used to reorder characters in a string to uppercase.

```
►►──TRANSLATE──(──┬─character_str_exp──┤ parameters-a ├──┬──────────────►◄
                  ├─graphic_str_exp──┤ parameters-b ├────┤
                  └─)────────────────────────────────────┘
```

parameters-a:

```
   ┌─,'ABC...XYZ',──'abc...xyz' ─────────────────────────┐
├──┴──┬──────────────────────────────────────────────────┴┬──────────────────┤
      └─,to_str_exp──┬─,X'000102...FDFEFF' ────────────┬──┘
                     │                                 │
                     └─,from_str_exp──┬─,' ' ──────┬───┘
                                      └─,pad_character─┘
```

parameters-b:

```
                                        ┌─,G'< >' ─────┐
├──,to_str_exp──,from str_exp───────────┴──────────────┴─────────────────────┤
                                        └─,pad_character─┘
```

VALUE

Returns the first nonnull result in a series of SQL expressions.

```
                      ┌───────────┐
                      ▼           │
►►──VALUE──(──exp─────,exp────────┴──)──────────────────────────────►◄
```

VARGRAPHIC

Returns a graphic string representation of a character string.

```
►►──VARGRAPHIC──(exp)──────────────────────────────────────────────►◄
```

YEAR

Extracts the year part of a value.

```
►►──YEAR──(──┬─date_exp───────────┬──)──────────────────────────────►◄
             ├─timestamp_exp──────┤
             ├─date_dur_exp───────┤
             └─timestamp_dur_exp──┘
```

Chapter 3. Queries

subselect

Specifies a result table derived from the tables or views identified in the FROM clause. Subselect is a component of the fullselect statement, the CREATE VIEW statement, the INSERT statement, and certain predicates.

```
►►──select_clause──from_clause──┬──────────────┬──┬─────────────────┬──────►
                                └─where_clause─┘  └─group_by_clause─┘

►──┬───────────────┬────────────────────────────────────────────────────────►◄
   └─having_clause─┘
```

select-clause

Produces a final result table by selecting only the columns indicated by the *select list* from R, where R is the result of the previous operation. For example, if the group-by-clause and having-clause are not specified, R is the result of the where-clause.

```
           ┌─ALL──────┐
►►──SELECT─┤          ├──┬─*───────────────────────────┬──────────────►◄
           └─DISTINCT─┘  │   ┌─,─────────────────────┐ │
                         │   ▼                       │ │
                         └───┬─exp─────────────────┬─┴─┘
                             ├─table_name.*────────┤
                             ├─view_name.*─────────┤
                             └─correlation_name.*──┘
```

from-clause

Names a single table or view, or produces an intermediate result table. The intermediate result table contains all possible combinations of the rows of the named tables or views.

```
          ┌─,──────────────────────────────────────┐
          ▼                                        │
►►──FROM────┬─table_name─┬──┬──────────────────┬───┴──────────────►◄
            └─view_name──┘  └─correlation_name─┘
```

where-clause

Produces an intermediate result table by applying *search-condition* to each row of R, where R is the result of the FROM clause. The result table contains the rows of R for which the *search-condition* is true.

```
>>--WHERE--search_condition------------------------------------------><
```

group-by-clause

Produces an intermediate result table by grouping the rows of R, where R is the result of the previous clause.

```
              .-,-----------.
              V             |
>>--GROUP BY----column_name-+----------------------------------------><
```

having-clause

Produces an intermediate result table by applying *search-condition* to each group of R, where R is the result of the previous clause.

```
>>--HAVING--search_condition-----------------------------------------><
```

fullselect

Specifies a result table. If UNION is not used, the result of the fullselect is the result of the specified subselect.

```
                       .-----------------------------------------------.
                       V                                               |
>>--+-subselect----+------+------------------------------------------+-+--------><
    '-(fullselect)-'      '-+-UNION-----+----+-subselect----+--------'
                            '-UNION ALL-'    '-(fullselect)-'
```

select-statement

Is the form of a query that can be specified or referenced in a DECLARE CURSOR statement and in the interactive select statement.

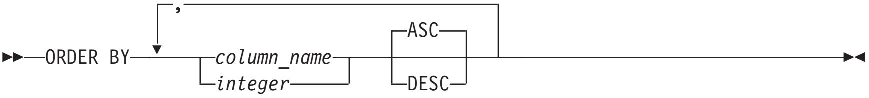
```
>>--fullselect--+-------------------+--+-------------+---------------><
                +-order_by_clause---+  '-with_clause-'
                |              (1)  |
                '-update_clause-----'
```

Notes:

1 The interactive select-statement does not incorporate the update-clause. That clause cannot be issued in ISQL or in the DBS Utility.

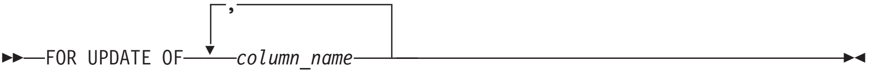
order-by-clause

Orders the rows of the result table by the values of the identified columns.



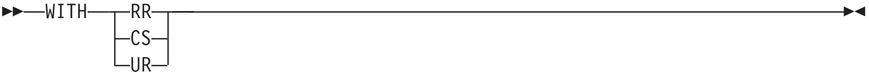
update-clause

Refers to the cursor in a positioned UPDATE statement. The UPDATE statement can update only columns in the *column-name* list. This update-clause cannot be used interactively.



with-clause

Specifies the isolation level at which the statement is executed.



Chapter 4. SQL Statements

Invocation

The letters **I** and **P**, printed to the right of the statement, indicate where each statement can be used. The **I** indicates the statement can be issued interactively, and the **P** indicates the statement can be embedded in an application program.

ACQUIRE DBSPACE (I,P)

Finds and names an available dbspace.

```
►►──ACQUIRE──┬─PUBLIC──┬──DBSPACE NAMED──dbspace_name──────────────────────────►
             └─PRIVATE─┘

►─┬──────────────────────────────────────────────────────┬─────────────────────►◄
  │    ┌─,───────────────────────────┐                   │
  │    ▼      (1)                    │                   │
  └─(─────┬─────────────────────────┬┴──)─┘
          │                ┌─8───────┐ │
          ├─NHEADER = ─────┴─integer─┴─┤
          │             ┌─128─────┐    │
          ├─PAGES = ────┴─integer─┴────┤
          │                 ┌─33──────┐│
          ├─PCTINDEX = ─────┴─integer─┴┤
          │                ┌─15──────┐ │
          ├─PCTFREE = ─────┴─integer─┴─┤
          │           ┌─PAGE────┐      │
          ├─LOCK = ───┼─DBSPACE─┼──────┤
          │           └─ROW─────┘      │
          └─STORPOOL = ──integer───────┘
```

Notes:

1 If any of these clauses is specified more than once, the value with the first specification is used.

ALLOCATE CURSOR (P)

Defines a cursor and associates it with a result set locator variable.

```
►►──ALLOCATE──cursor-name──CURSOR FOR RESULT SET──rs-locator-variable──────────►◄
```

ALTER DBSPACE (I,P)

Changes the percentage of free space and the type of locking of a PUBLIC dbspace.

```
►►──ALTER DBSPACE──dbspace_name──────────────────────────────────────────►

      ┌─,──────────────────────────────┐
      │     (1)                        │
►─(───▼────────────┬─PCTFREE = integer──┬──┴──)─────────────────────────────►◄
                   └─LOCK =───┬─PAGE─────┤
                              ├─DBSPACE──┤
                              └─ROW──────┘
```

Notes:

1 If either of these clauses is specified more than once, the value with the first specification is used.

ALTER PROCEDURE (I,P)

Alters the definition of a stored procedure.

ALTER PROCEDURE

```
►►──ALTER PROCEDURE──procedure-name──┬─────────────────┬──────────────────►
                                     └─AUTHID──authid──┘

   ┌─,───────────┐
   ▼             │   (1)
►───┤ options ├──┴─────────────────────────────────────────────────────────►◄
```

Notes:

1 One or more clauses may be specified, however each clause may be specified at most once.

options:

```
├──┬─LANGUAGE───┬─ASSEMBLE─┬───────────────────────────────┬───────────────────────────────────────────────┤
   │            ├─C────────┤                               │
   │            ├─COBOL────┤                               │
   │            └─PLI──────┘                               │
   ├─EXTERNAL NAME──external-program-name──────────────────┤
   ├─SERVER GROUP──┬───────────────────┬───────────────────┤
   │               └─server-group-name─┘                   │
   ├─┬─DEFAULT SERVER GROUP YES─┬──────────────────────────┤
   │ └─DEFAULT SERVER GROUP NO──┘                          │
   │                                   (2)                 │
   ├───┬─────────────────┬───┬─GENERAL─────────────────┬───┤
   │   └─PARAMETER STYLE─┘   │                    (3)  │   │
   │                         └─GENERAL WITH NULLS──────┘   │
   ├───STAY RESIDENT─┬─NO──┬───────────────────────────────┤
   │                 └─YES─┘                               │
   ├─┬─PROGRAM TYPE MAIN─────┬─────────────────────────────┤
   │ │                  (4)  │                             │
   │ └─PROGRAM TYPE SUB──────┘                             │
   ├─RUN OPTIONS──run-time-options─────────────────────────┤
   ├─RESULT──┬─SET──┬───integer────────────────────────────┤
   │         └─SETS─┘                                      │
   ├─COMMIT ON RETURN─┬─NO──┬──────────────────────────────┤
   │                  └─YES─┘                              │
   │                     (1)  (5)                          │
   ├─┬─NOT DETERMINISTIC───────────┬───────────────────────┤
   │ │               (1)  (6)      │                       │
   │ └─DETERMINISTIC───────────────┘                       │
   │                (1)                                    │
   ├─┬─CONTAINS SQL───────────┬────────────────────────────┤
   │ │        (1)             │                            │
   │ ├─NO SQL─────────────────┤                            │
   │ │                (1)     │                            │
   │ ├─READS SQL DATA─────────┤                            │
   │ │                   (1)  │                            │
   │ └─MODIFIES SQL DATA──────┘                            │
   │             (1)                                       │
   ├─┬─NO COLLID───────────────────┬───────────────────────┤
   │ │                       (1)   │                       │
   │ └─COLLID──collection-id───────┘                       │
   │                   (1)                                 │
   ├─┬─WLM ENVIRONMENT──────┬─name─────┬─┬─────────────────┤
   │ │                      └─(name,*)─┘ │                 │
   │ │                    (1)            │                 │
   │ └─NO WLM ENVIRONMENT────────────────┘                 │
   │         (1)                                           │
   ├─ASUTIME───────┬─NO LIMIT────────┬─────────────────────┤
   │               └─LIMIT──integer──┘                     │
   │    (1)                                                │
   ├─┬───────────────────────────────┬─────────────────────┤
   │ └─EXTERNAL SECURITY─┬─DB2─────┬─┘                     │
   │                     ├─USER────┤                       │
   │                     └─DEFINER─┘                       │
   │             (1)                                       │
   └─┬─NO DBINFO──────┬────────────────────────────────────┘
     │        (1)     │
     └─DBINFO─────────┘
```

Notes:

1 This parameter is included for compatibility with the DB2 family. If specified, it is ignored.

2 SIMPLE CALL may be used as an alternative to GENERAL. This is for compatibility within the DB2 family.

3 SIMPLE CALL WITH NULLS may be used as an alternative to GENERAL WITH NULLS. This is for compatibility within the DB2 family.

4 Currently, DB2 Server for VSE & VM supports stored procedures written as main programs only.

5 VARIANT may be specified as an alternative to NOT DETERMINISTIC. This is for compatibility within the DB2 family.

6 NOT VARIANT may be specified as an alternative to DETERMINISTIC. This is for compatibility within the DB2 family.

ALTER PSERVER (I,P)

Alters the definition of a stored procedure server.

```
                                          ┌─,────────────────────────────┐
                                          │     (1)                      │
►►──ALTER PSERVER──procedure-server───────▼─┬────────────────────────┬───┴───────►◄
                                            ├─GROUP─┬──────────────┬─┤
                                            │       └─group-name───┘ │
                                            ├─┬─AUTOSTART NO──┬──────┤
                                            │ └─AUTOSTART YES─┘      │
                                            └─DESCRIPTION──description┘
```

Notes:

1 One or more clauses may be specified, however each clause may be specified at most once.

ALTER TABLE (I,P)

Adds a new column, or adds, drops, activates, or deactivates a primary key, foreign key, or unique constraint on a specified table.

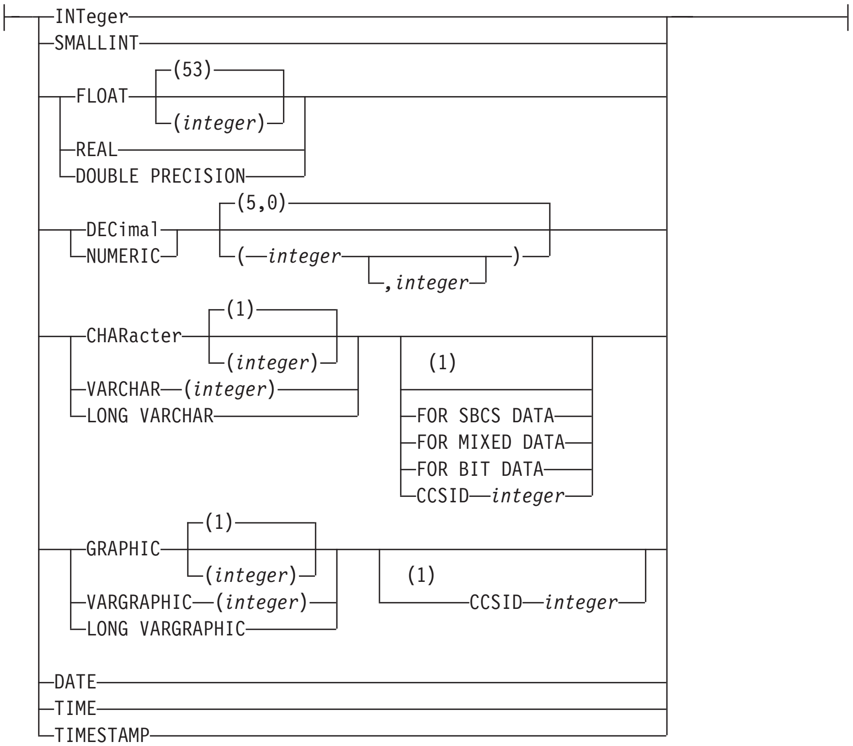
```
►►──ALTER TABLE──table_name──────────────────────────────────────────────────►
```

```
►►─┬─ADD──┤ column-definition-block ├──────────────────┬──────────────────────►◄
   │  ┌─ADD─┐                                           │
   ├──┴─────┴──┬─┤ primary-key-block ├───────────────────┤
   │           ├─┤ referential-constraint-block ├──┐     │
   │           └─┤ unique-block ├──────────────────┘     │
   ├─DROP──┬─PRIMARY KEY─────────────────────────────────┤
   │       ├─FOREIGN KEY──constraint_name──┐             │
   │       └─UNIQUE──constraint_name───────┘             │
   ├─ACTIVATE──┬─ALL─────────────────────────────────────┤
   │           ├─PRIMARY KEY─────────────────────┐       │
   │           ├─FOREIGN KEY──constraint_name────┤       │
   │           └─UNIQUE──constraint_name─────────┘       │
   ├─DEACTIVATE──┬─ALL───────────────────────────────────┤
   │             ├─PRIMARY KEY───────────────────────┐   │
   │             ├─FOREIGN KEY──constraint_name──────┤   │
   │             └─UNIQUE──constraint_name───────────┘   │
   └─DATA CAPTURE──┬─NONE─────┬──────────────────────────┘
                   └─CHANGES──┘
```

column-definition-block:

```
├──column_name──┤ data-type ├──┬────────────────────────────┬──────────────┤
                               │  (1)                       │
                               └──────┤ fieldproc-block ├───┘
```

data-type:

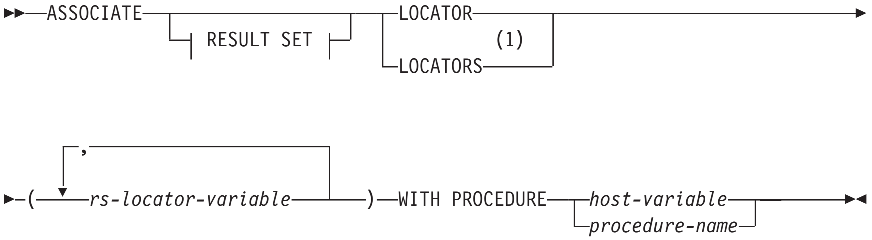


Notes:

1 These clauses may be specified in any order.

ASSOCIATE LOCATORS (P)

Obtains the RESULT SET LOCATOR value for each result set data type returned by a stored procedure.



Notes:

1 RESULT SET LOCATOR variables are only supported in client applications written in Assembler, C, COBOL, and PL/I.

BEGIN DECLARE SECTION (P)

Marks the beginning of a host variable declare section, including host structures.

```
►►──BEGIN DECLARE SECTION──────────────────────────────────────────────►◄
```

CALL (P)

Invokes a stored procedure with a list of input/output parameters.

```
►►──CALL──┬─procedure-name─┬──┬──────────────────────────────────────┬──────►◄
          └─host-variable──┘  ├─(──┬──────────────────────────┬──)──┤
                              │    │   ┌─,─────────────────┐  │     │
                              │    │   ▼                   │  │     │
                              │    └───┬─host-variable─┬───┴──┘     │
                              │        ├─constant──────┤            │
                              │        └─NULL──────────┘            │
                              └─USING DESCRIPTOR──descriptor-name───┘
```

CLOSE (P)

Closes the cursor identified by *cursor-name*.

```
►►──CLOSE──cursor_name─────────────────────────────────────────────────►◄
```

Extended CLOSE (P)

Closes the cursor identified by *cursor-variable*.

```
►►──CLOSE──cursor_variable─────────────────────────────────────────────►◄
```

COMMENT ON (I,P)

Adds or replaces comments in the catalog descriptions of tables, views, or columns.

```
►►──COMMENT ON──────────────────────────────────────────────────────────────────►

►──┬─┤ options_a ├────────────────────────────IS──str_constant──┬────────►◄
   └─┬─table_name─┬───(──┤ options_b ├──────────────────────)───┘
     └─view_name──┘
```

options_b

```
    ┌─,──────────────────────────┐
    ▼                            │
├────column_name──IS──str_constant─┴──────────────────────────────────────┤
```

options_a

```
├───┬─TABLE───┬─table_name─────────────┬──┬──────────────────────────────┤
    │         └─view_name──────────────┘  │
    └─COLUMN──┬─table_name.column_name──┬─┘
              └─view_name.column_name───┘
```

COMMENT ON PROCEDURE (I,P)

Adds or replaces comments in the catalog descriptions of stored procedures.

```
►►──COMMENT ON PROCEDURE──procedure_name──┬────────────────────┬──────────────►
                                          └─AUTHID──authid─────┘

►──IS──string_constant────────────────────────────────────────────────────────►◄
```

COMMIT (I,P)

Ends the current logical unit of work and commits any changes.

```
          ┌─WORK─┐
►►──COMMIT─┴──────┴──┬──────────┬──────────────────────────────────────────────►◄
                     └─RELEASE──┘
```

CONNECT (I,P)

Connects an application process or a user, or both, to an application server.

```
►►──CONNECT──────────────────────────────────────────────────────────────►

►──┬─────────────────────────────────────────────────────────────────┬──►
   └─┬─authorization_name─┬──IDENTIFIED BY──┬─password──────┬─┘
     └─host_variable──────┘                 └─host_variable─┘

►──┬────────────────────────────┬────────────────────────────────────►◄
   └─TO──┬─server_name───┬──────┘
         └─host_variable─┘
```

CREATE INDEX (I,P)

Creates an index on one or more columns of a table.

```
►►──CREATE──┬────────┬──INDEX──index_name────────────────────────────►◄
            └─UNIQUE─┘

                          ┌─,──────────────────────┐
                          ▼             ┌─ASC──┐   │
►►──ON──table_name──(─────column_name───┼──────┼───┴─)───────────────►◄
                                        └─DESC─┘

    ┌─PCTFREE=10──────┐
►►──┼─────────────────┼──────────────────────────────────────────────►◄
    └─PCTFREE=integer─┘
```

CREATE PACKAGE (P)

Creates a package.

```
                 (1)
►►──CREATE PACKAGE───────────────────────────────────────────────────►
```

```
►─package_spec─┬────────────────────────────────────────────┬──────────►◄
               │                  ┌──────────────────┐       │
               │                  ▼  (2)             │       │
               └─USING OPTIONs─┬──────────option─────┴─┬─────┘
                               └─host_variable─────────┘
```

Notes:

1 PROGRAM is equivalent to PACKAGE, and is provided for compatibility with some older versions of the SQL/DS product.

2 An option may be specified only once.

Using Options

Table 1.

CCSIDSbcs (integer)

CCSIDMixed (integer)

CCSIDGraphic (integer)

CHARSUB (Sbcs | Mixed | Bit)

DATE (ISO | USA | EUR | JIS | LOCAL)

EXPLAIN (NO | YES)

ISOLation (RR | RS | CS | UR | USER)

KEEP | REVOKE

LABEL (label-text)

NOBLock | BLock | SBLock

NOCHECK | CHECK | ERROR

NODESCRIBE | DESCRIBE

NOEXIST | EXIST

NOMODIFY | MODIFY

OWner (authorization-name)

QUALifier (collection-id)

RELease (COMMIT | DEALLOCATE)

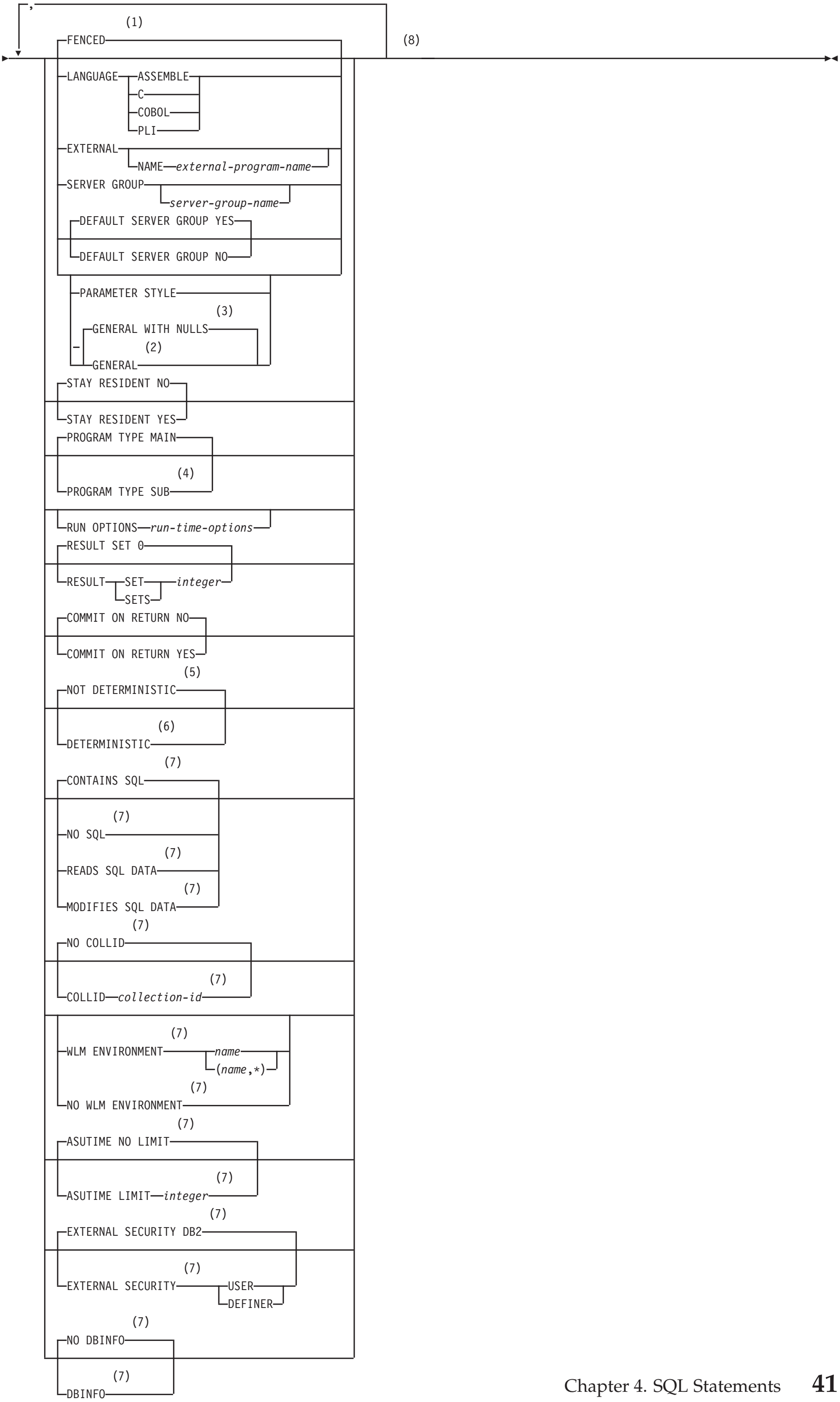
REPLACE | NEW

TIME (ISO | USA | EUR | JIS | LOCAL)

CREATE PROCEDURE (I,P)

Defines a stored procedure.

```
►►─CREATE PROCEDURE─procedure-name─┬──────────────┬─(─┬──────────────┬─)──────────►
                                   └─AUTHID─authid─┘   └┤ parameters ├┘
```

,
FENCED (1)
LANGUAGE
ASSEMBLE
C
COBOL
PLI
EXTERNAL
NAME external-program-name
SERVER GROUP
server-group-name
DEFAULT SERVER GROUP YES
DEFAULT SERVER GROUP NO
PARAMETER STYLE
GENERAL WITH NULLS (3)
GENERAL (2)
STAY RESIDENT NO
STAY RESIDENT YES
PROGRAM TYPE MAIN
PROGRAM TYPE SUB (4)
RUN OPTIONS run-time-options
RESULT SET 0
RESULT SET integer
SETS
COMMIT ON RETURN NO
COMMIT ON RETURN YES
NOT DETERMINISTIC (5)
DETERMINISTIC (6)
CONTAINS SQL (7)
NO SQL (7)
READS SQL DATA (7)
MODIFIES SQL DATA (7)
NO COLLID (7)
COLLID collection-id (7)
WLM ENVIRONMENT (7) name
(name,*)
NO WLM ENVIRONMENT (7)
ASUTIME NO LIMIT (7)
ASUTIME LIMIT integer (7)
EXTERNAL SECURITY DB2 (7)
EXTERNAL SECURITY (7) USER
DEFINER
NO DBINFO (7)
DBINFO (7)
(8)

Notes:

1 This parameter is included for compatibility with the DB2 family. If specified, it is ignored.

2 As an alternative to GENERAL, SIMPLE CALL may be used. This is for compatibility within the DB2 family.

3 As an alternative to GENERAL WITH NULLS, SIMPLE CALL WITH NULLS may be used. This is for compatibility within the DB2 family.

4 Currently, DB2 Server for VSE & VM supports stored procedures written as main programs only.

5 VARIANT may be specified as an alternative to NOT DETERMINISTIC. This is for compatibility within the DB2 family.

6 NOT VARIANT may be specified as an alternative to DETERMINISTIC. This is for compatibility within the DB2 family.

7 This parameter is included for compatibility with the DB2 family. If specified, it is ignored.

8 One or more clauses may be specified, however each clause may be specified at most once.

parameters:

```
   ┌,──────────────────────────────────────────────────────────────────┐
   │ ┌─IN────┐                                                          │
├──▼─┼───────┼──┬────────────────┬──┤ data-type ├──┬──────────────────┬──┴──┤
     ├─OUT───┤  └─parameter-name─┘                │              (1) │
     └─INOUT─┘                                    └─AS LOCATOR───────┘
```

data-type:

```
├──┬─INT──────────────────────────────────────────────────────────┬──────────┤
   ├─INTEGER──────────────────────────────────────────────────────┤
   ├─SMALLINT─────────────────────────────────────────────────────┤
   ├─REAL─────────────────────────────────────────────────────────┤
   ├─FLOAT────────────────────────────────────────────────────────┤
   ├─DOUBLE───────────────────────────────────────────────────────┤
   ├─DOUBLE PRECISION─────────────────────────────────────────────┤
   ├──┬─DECIMAL─┬──┬──────────────────────────────────┬──────────┤
   │  └─DEC─────┘  └─(─integer──┬──────────────┬──)─┘           │
   │                            └─,integer─────┘                 │
   ├──┬─CHARACTER─┬──┬─────────────┬──┬────────────────────────┬─┤
   │  └─CHAR──────┘  └─(integer)───┘  └─FOR──┬─SBCS──┬──DATA──┘ │
   │                                         ├─MIXED─┤          │
   │                                         └─BIT───┘          │
   ├─VARCHAR(integer)──┬──────────────────────────┬──────────────┤
   │                   └─FOR──┬─SBCS──┬──DATA──┘               │
   │                          ├─MIXED─┤                         │
   │                          └─BIT───┘                         │
   ├─GRAPHIC(integer)─────────────────────────────────────────────┤
   └─VARGRAPHIC(integer)──────────────────────────────────────────┘
```

Notes:

1 This parameter is included for compatibility with the DB2 family. If specified, it is ignored.

CREATE PSERVER (I,P)

Defines a stored procedure server.

```
                                      ┌─,───────────────────────────┐
                                      ▼                             │
►►──CREATE PSERVER──procedure-server────┬──────────────────────────┬┴───►◄
                                        │  (1)                     │
                                        ├──────────────────────────┤
                                        ├─GROUP──group-name────────┤
                                        │ ┌─AUTOSTART NO──┐        │
                                        ├─┤               ├────────┤
                                        │ └─AUTOSTART YES─┘        │
                                        └─DESCRIPTION──description─┘
```

Notes:

1 One or more clauses may be specified, however each clause may be specified at most once.

CREATE SYNONYM (I,P)

Defines an alternative name for a table or view.

```
►►──CREATE SYNONYM──synonym──FOR──┬─qualified_table_name─┬──────────────►◄
                                  └─qualified_view_name──┘
```

CREATE TABLE (I,P)

Creates a new table.

```
►►──CREATE TABLE──table_name─────────────────────────────────────────────►

      ┌─,─────────────────────────────────────┐
      │   (1)                                 │
      ▼                                       │
►──(──────────┬─┤ column-definition-block ├───┴──)───────────────────────►
              │                       (2)     │
              ├─┤ primary-key-block ├─────────┤
              ├─┤ referential-constraint-block ├┤
              └─┤ unique-block ├──────────────┘

   ┌──────────────────────────┐
   ▼                          │
►──────┬─IN──dbspace_name──────┬┴──────────────────────────────────────►◄
       │                  (3)  │
       └─DATA CAPTURE──┬─NONE────┬┘
                       └─CHANGES─┘
```

Notes:

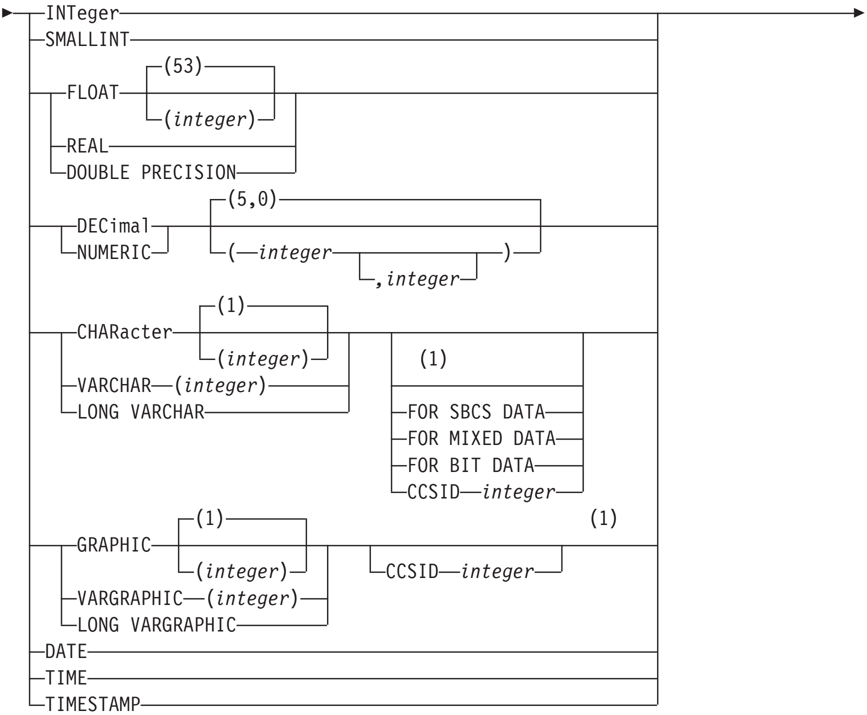
1 There can be up to 255 columns in a table.

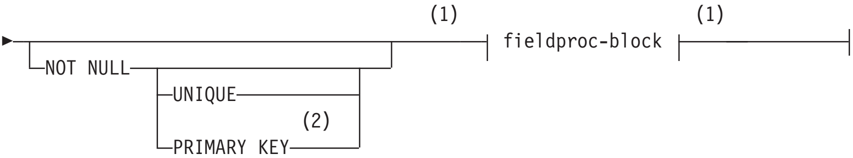
2 Only one primary key may be defined (either in a primary-key-block or as a column attribute).

3 The same clause must not be specified more than once.

column-definition-block:

```
├──column_name────────────────────────────────────────────────────────────►
```



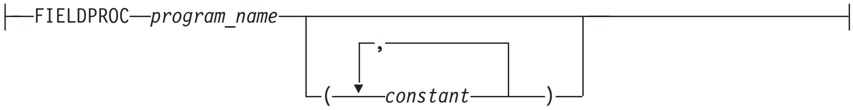


Notes:

1 These clauses may be specified in any order.

2 Only one primary key may be defined (either in a primary-key-block or as a column attribute).

fieldproc-block:



primary-key-block:

```
                   ┌─,────────────────────────────┐
                   ▼      (1)          ┌─ASC──┐   │
├──PRIMARY KEY──(──────────column_name─┼──────┼───┴──)──────────────►
                                       └─DESC─┘

   ┌─PCTFREE = 10─────┐
►──┼──────────────────┼─────────────────────────────────────────────┤
   └─PCTFREE=integer──┘
```

Notes:

1 A PRIMARY KEY can have up to 16 columns.

referential-constraint-block:

```
                                            ┌─,───────────┐
                                            ▼             │
├──FOREIGN KEY──┬─────────────────┬──(──────column_name───┴─────────►
                └─constraint_name─┘

►──)──REFERENCES──table_name──┬─────────────────────────────┬──────┤
                              │             ┌─RESTRICT─┐    │
                              └─ON DELETE───┼─CASCADE──┼────┘
                                            └─SET NULL─┘
```

unique-block:

```
                                          ┌─,──────────────────────────┐
                                          ▼     (1)          ┌─ASC──┐  │
├──UNIQUE──┬─────────────────┬──(──────────column_name───────┼──────┼──┴──)──►
           └─constraint_name─┘                               └─DESC─┘

   ┌─PCTFREE=10───────┐
►──┼──────────────────┼─────────────────────────────────────────────┤
   └─PCTFREE=integer──┘
```

Notes:

1 There can be up to 16 columns on a unique constraint.

CREATE VIEW (I,P)

Creates a view on one or more tables or views.

```
►►──CREATE VIEW──view_name──┬───────────────────────────┬──────────────────►
                            │     ┌─,───────────┐       │
                            │     ▼             │       │
                            └─(───column_name───┴───)───┘

►─AS──subselect──┬───────────────────┬──────────────────────────────────────►◄
                 └─WITH CHECK OPTION─┘
```

DECLARE CURSOR (P)

Declares the cursor that you can use to fetch or put the results of a prepared statement.

```
►►──DECLARE──cursor-name──CURSOR──┬────────────────┬──FOR──────────────────►
                                  ├─WITH RETURN────┤
                                  │          (1)   │
                                  └─WITH HOLD──────┘

►──┬─select-statement─┬──────────────────────────────────────────────────►◄
   └─statement-name───┘
```

Notes:

1 Note that DB2 Server for VSE & VM does not support CURSOR WITH HOLD.

Extended DECLARE CURSOR (P)

Declares the cursor that you can use to fetch or put the results of a prepared statement.

```
►►──DECLARE──cursor_variable──CURSOR FOR──section_variable─────────────────►

►─IN──package_spec─────────────────────────────────────────────────────────►◄
```

DELETE (I,P)

Deletes one or more rows from a table or view. Deleting a row from a view deletes the row from the table on which the view is based.

Searched delete (I,P)

```
►►──DELETE FROM──┬─table_name─┬──┬──────────────────┬────────────────────►◄
                 └─view_name──┘  └─correlation_name─┘


►►──┬─────────────────────────┬──┬─────────────────┬──────────────────────►◄
    └─WHERE──search_condition─┘  └─WITH──┬─RR─┬────┘
                                         └─CS─┘
```

Positioned delete (P)

```
                                                    (1)
►►──DELETE FROM──┬─table_name─┬──WHERE CURRENT OF────────cursor_name──────►◄
                 └─view_name──┘
```

Notes:

1 A Positioned DELETE in FORTRAN, and programs prepared using Extended dynamic SQL cannot be used with DRDA protocol.

DESCRIBE (P)

Retrieves information about an SQL select-statement previously prepared with a PREPARE statement.

```
►►──DESCRIBE──statement_name──INTO──descriptor_name──┬────────────────────┬──►◄
                                                     └─USING──┬─NAMES──┬──┘
                                                              ├─ANY────┤
                                                              ├─BOTH───┤
                                                              └─LABELS─┘
```

Extended DESCRIBE (P)

Retrieves information about an SQL SELECT statement previously prepared with an Extended PREPARE statement.

```
►►──DESCRIBE──statement_variable──IN──package_spec─────────────────────────►
```

```
►─INTO──descriptor_name──┬─────────────────────────┬──────────────────────────►◄
                         │             ┌─NAMES──┐  │
                         └─USING───────┼─ANY────┼──┘
                                       ├─BOTH───┤
                                       └─LABELS─┘
```

DESCRIBE CURSOR (P)

Obtains information about the result set that is associated with the cursor and puts that information into a descriptor.

```
►►──DESCRIBE CURSOR──┬─cursor-name───┬───INTO──descriptor-name───────────────►◄
                     └─host-variable─┘
```

DESCRIBE PROCEDURE (P)

Obtains information about the result sets returned by a stored procedure and puts that information into a descriptor.

```
►►──DESCRIBE PROCEDURE──┬─host-variable──┬───INTO──descriptor-name───────────►◄
                        └─procedure-name─┘
```

DROP (I,P)

Deletes an object. Objects that are directly or indirectly dependent on that object are also deleted.

```
►►──DROP──┬─DBSPACE──dbspace_name─────┬──────────────────────────────────────►◄
          ├─INDEX──index_name─────────┤
          │         (1)  (2)          │
          ├─PACKAGE────────package_spec─┤
          ├─SYNONYM──synonym──────────┤
          ├─TABLE──table_name─────────┤
          └─VIEW──view_name───────────┘
```

Notes:

1 PROGRAM is equivalent to PACKAGE and is provided for compatibility with prior releases of SQL/DS.

2 DROP PACKAGE cannot support a qualified structure subfield name. A host structure subfield name can be used as normal host variables, but must be unqualified. If being unqualified results in an ambiguous reference, the subfield cannot be used.

DROP PROCEDURE (I,P)

Removes a stored procedure.

```
►►──DROP PROCEDURE──procedure-name──┬─────────────────┬──┬────────────┬──────────►◄
                                    └─AUTHID──authid──┘  └─RESTRICT───┘
```

DROP PSERVER (I,P)

Removes a stored procedure pserver.

```
►►──DROP PSERVER──procedure-server──────────────────────────────────────────────►◄
```

DROP STATEMENT (P)

Selectively deletes a statement from a package.

```
►►──DROP STATEMENT──section_variable──IN──package_spec──────────────────────────►◄
```

END DECLARE SECTION (P)

Marks the end of a *host-variable* declare section.

```
►►──END DECLARE SECTION─────────────────────────────────────────────────────────►◄
```

EXECUTE (P)

Executes a prepared SQL statement.

```
►►──EXECUTE──statement_name──┬──────────────────────────────────────────┬───────►◄
                             ├─USING──host_variable_list────────────────┤
                             └─USING DESCRIPTOR──descriptor_name────────┘
```

Extended EXECUTE (P)

Executes a statement previously prepared by an Extended PREPARE statement.

```
►►──EXECUTE──section_variable──IN──package_spec─────────────────────────────────►
```

```
►──┬─────────────────────────────────────┬───────────────────────────────────────►
   └─USING DESCRIPTOR──descriptor_name_1─┘
```

```
►──┬────────────────────────────────────────────┬────────────────────────────────►◄
   └─USING OUTPUT DESCRIPTOR──descriptor_name_2─┘
```

EXECUTE IMMEDIATE (P)

Prepares an executable form of an SQL statement from a character string form of the statement, executes the SQL statement, and then destroys the executable form.

```
►►──EXECUTE IMMEDIATE──┬─string_constant─┬───────────────────────────────────────►◄
                       └─host_variable───┘
```

EXPLAIN (I,P)

Retrieves information about the access path chosen for the execution of the SQL query, and about the structure and execution performance of a DELETE, INSERT, UPDATE or select-statement.

```
►►──EXPLAIN──┬─ALL──────────────────┬─┬──────────────────────────────────────┬──►
             │   ┌─,──────────────┐ │ │                        (1)   (2)     │
             │   ▼                │ │ └─SET QUERYNO =──integer───────────────┘
             └─────┬─COST──────┬──┴─┘
                   ├─PLAN──────┤
                   ├─REFERENCE─┤
                   └─STRUCTURE─┘
```

```
►─FOR──explainable_sql_statement──────────────────────────────────────────────────►◄
```

Notes:

1 The QUERYNO can be up to 2,147,483,647.

2 The integer constant must not be preceded by a sign.

FETCH (P)

Positions a cursor on the next row of its result table and assigns the values of that row to the host variables.

```
►►──FETCH──cursor_name──┬─INTO──host_variable_list──────────┬──────────────►◄
                        └─USING DESCRIPTOR──descriptor_name──┘
```

Extended FETCH (P)

Positions a cursor on the next row of its result table and assigns the values of that row to the host variables.

```
►►──FETCH──cursor_variable──USING DESCRIPTOR──descriptor_name──────────────►◄
```

GRANT Package Privileges (I,P)

Grants the privilege to execute a package.

```
                                                 ┌─,──────────────────┐
                   (1)                           ▼                    │
►►──GRANT EXECUTE ON──────package_name──TO──────┬─authorization_name─┬┴──────►
                                                └─PUBLIC─────────────┘

►──┬──────────────────┬─────────────────────────────────────────────────────►◄
   └─WITH GRANT OPTION─┘
```

Notes:

1 RUN can be specified as a synonym for EXECUTE to support applications developed for previous releases of SQL/DS.

GRANT System Authorities (I,P)

Grants authorities to users and changes passwords.

```
►►──GRANT──┬───┬─CONNECT──┬─TO──┤ AUTH ├──┬────────┬──┬────────────────►◄
           │   ├─DBA──────┤              └─┤ ID ├─┘  │
           │   └─RESOURCE─┘                          │
           │                                         │
           │              ┌─,──────────────────┐     │
           │              ▼                    │     │
           ├─CONNECT TO──┬─authorization_name─┬┴─────┤
           │             │             (1)    │      │
           │             └─ALLUSERS───────────┘      │
           └─SCHEDULE TO──subsystemid──IDENTIFIED BY──password──┘
```

AUTH:

```
         ┌─,──────────────────────┐
         ▼                        │
├────────authorization_name───────┴──────────────────────────────────┤
```

ID:

```
                   ┌─,────────┐
                   ▼          │
├──IDENTIFIED BY─────password─┴──────────────────────────────────────┤
```

Notes:

1 ALLUSERS can only be specified once and is not applicable to a VSE application server.

GRANT Table Privileges (I,P)

Grants privileges on a table or view.

```
                      ┌─PRIVILEGES─┐
►►──GRANT──┬──ALL─────┴────────────┴────────────────────────┬──────────────►
           │   ┌─,────────────────────────────────────────┐ │
           │   ▼       (1)                                │ │
           └─────┬─ALTER────────────────────────────────┬─┴─┘
                 ├─DELETE───────────────────────────────┤
                 │         (1)                          │
                 ├─INDEX────────────────────────────────┤
                 ├─INSERT───────────────────────────────┤
                 │             (1)                      │
                 ├─REFERENCES───────────────────────────┤
                 ├─SELECT───────────────────────────────┤
                 └─UPDATE──┬──────────────────────────┬─┘
                           │      ┌─,──────────┐      │
                           │      ▼            │      │
                           └─(────column_name──┴──)───┘
```

```
                                 ┌─,──────────────────────┐
                                 ▼                        │
►──ON──┬─table_name─┬──TO──────┬─authorization_name─┬─────┴──────────────►
       └─view_name──┘          └─PUBLIC─────────────┘
```

```
►─┬─────────────────┬──────────────────────────────────────────────────►◄
  └─WITH GRANT OPTION─┘
```

Notes:

1 ALTER, INDEX, and REFERENCES do not apply to views.

INCLUDE (P)

Inserts declarations or statements into a source program.

```
►►──INCLUDE──┬─SQLCA──────────┬────────────────────────────────────────►◄
             ├─SQLDA──────────┤
             └─text_file_name─┘
```

INSERT (I,P)

Inserts rows into a table or view. Inserting a row in a view inserts the row into the tables on which the view is based.

```
►►──INSERT INTO──┬─table_name─┬──┬────────────────────────────────┬────►
                 └─view_name──┘  │      ┌─,────────────────┐      │
                                 │      ▼                  │      │
                                 └─(────┬─────────────┬────┴─)────┘
                                        └─column_name─┘
```

```
       ┌─,─────────────────────────┐
       ▼                           │
►─┬─VALUES──(───┬─constant───────────┬─┴──)──┬─────────────────────────►◄
  │             ├─host_variable_list─┤       │
  │             ├─NULL───────────────┤       │
  │             └─special_register───┘       │
  └─subselect──┬───────────────────┬─────────┘
               └─WITH──┬─RR─┬──────┘
                       └─CS─┘
```

LABEL ON (I,P)

Adds or replaces labels in the catalog descriptions of tables, views, or columns.

```
►►──LABEL ON───────────────────────────────────────────────────────────►
```

```
►──┬─┤ options_a ├─────────────────────────────────IS─str_constant─┬──────►◄
   └─┬─table_name─┬──(─┤ options_b ├──────────────────────────)──┘
     └─view_name──┘
```

options_b

```
    ┌─,─────────────────────────────┐
    ▼                               │
├──────column_name─IS─str_constant──┴─────────────────────────────────────┤
```

options_a

```
├──┬─TABLE───┬─table_name─┬──────────────┬─────────────────────────────────┤
   │         └─view_name──┘              │
   └─COLUMN──┬─table_name.column_name────┤
             └─view_name.column_name───┘
```

LOCK DBSPACE (I,P)

Acquires a shared or exclusive lock on the *dbspace-name* specified.

```
►►──LOCK DBSPACE─dbspace_name─IN──┬─SHARE─────┬─MODE──────────────────────►◄
                                  └─EXCLUSIVE─┘
```

LOCK TABLE (I,P)

Acquires a shared or exclusive lock on the named table.

```
►►──LOCK TABLE─table_name─IN──┬─SHARE─────┬─MODE──────────────────────────►◄
                              └─EXCLUSIVE─┘
```

OPEN (P)

Opens a cursor.

```
►►──OPEN─cursor_name──┬─────────────────────────────────────┬─────────────►◄
                      ├─USING─host_variable_list────────────┤
                      └─USING DESCRIPTOR─descriptor_name────┘
```

Extended OPEN CURSOR (P)

Opens a cursor.

```
►►──OPEN──cursor_variable──┬──────────────────────────────────────┬──────────►◄
                           └─USING DESCRIPTOR──descriptor_name──┘
```

PREPARE (P)

Prepares an SQL statement for execution from a character string form of the statement.

```
►►──PREPARE──statement_name──FROM──┬─string_constant─┬─────────────────►◄
                                   └─host_variable───┘
```

Extended PREPARE (P)

Basic Extended PREPARE

Adds an SQL statement to an existing package.

```
►►──PREPARE FROM──host_variable──────────────────────────────────────────►◄

►►──SETTING──section_variable──IN──package_spec─────────────────────────►

►──┬─────────────────────────────────────┬───────────────────────────────►◄
   └─USING DESCRIPTOR──descriptor_name──┘
```

Single Row Extended PREPARE

Indicates that the SELECT statement in the *host-variable* is a single row SELECT.

```
►►──PREPARE SINGLE ROW FROM──host_variable───────────────────────────────►◄

►►──SETTING──section_variable──IN──package_spec─────────────────────────►

►──┬─────────────────────────────────────┬───────────────────────────────►◄
   └─USING DESCRIPTOR──descriptor_name──┘
```

Empty Extended PREPARE

Allows for the creation of an indefinite section in a program.

```
►►──PREPARE FROM NULL SETTING──section_variable──IN──package_spec──────────────►◄
```

Temporary Extended PREPARE

Prepares the statement in the *host-variable* and associates the output with a previously created indefinite section.

```
►►──PREPARE FROM──host_variable──FOR──section_variable──IN──package_spec───────►◄
```

PUT (P)

Inserts a row into a table.

```
►►──PUT──cursor-name───────────────────────────────────────────────────────────►◄
```

```
►►─────┬────────────────────────────────────┬──────────────────────────────────►◄
       ├─FROM──host-variable-list───────────┤
       └─USING DESCRIPTOR──descriptor-name──┘
```

Extended PUT (P)

Inserts a row into a table.

```
►►──PUT──cursor_variable──┬────────────────────────────────────┬───────────────►◄
                          ├─FROM──host_variable_list───────────┤
                          └─USING DESCRIPTOR──descriptor_name──┘
```

REVOKE Package Privileges (I,P)

Revokes the privilege to execute a package.

```
                    (1)
►►──REVOKE EXECUTE───────ON──package_name──────────────────────────────────────►
```

```
          ┌─,────────────────────┐
          ▼                      │
►─FROM──────┬─authorization_name─┬─┴────────────────────────────────────────────►◄
            │         (2)        │
            └─PUBLIC─────────────┘
```

Notes:

1 RUN can be used as a synonym for EXECUTE and is provided for compatibility with previous versions of SQL/DS.

2 PUBLIC is specified only once.

REVOKE System Authorities (I,P)

Revokes system authorities.

```
                             ┌─,────────────────────┐
                             ▼                      │
►►──REVOKE──┬─CONNECT FROM─────┬─authorization_name─┬─┴──┬──────────────►◄
            │                  │            (1)     │    │
            │                  └─ALLUSERS───────────┘    │
            │                         ┌─,──────────────┐ │
            │                         ▼                │ │
            ├──┬─DBA──────┬──FROM──────authorization_name─┴─┤
            │  └─RESOURCE─┘                              │
            └─SCHEDULE FROM──subsystemid─────────────────┘
```

Notes:

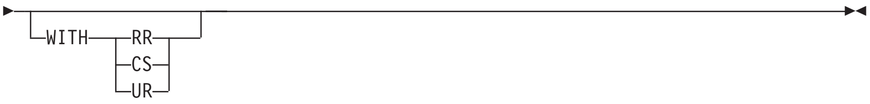
1 ALLUSERS can only be specified once.

REVOKE Table Privileges (I,P)

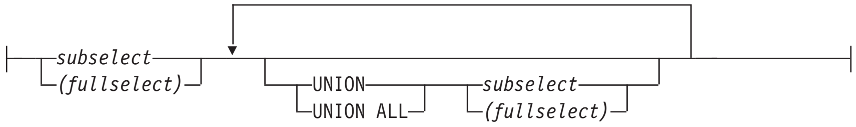
Revokes privileges on a table or view.

interactive-select-statement

```
►►──┤ fullselect ├──┬──────────────────────────────────────────────┬──►
                    │            ┌─,──────────────────────────┐  │
                    │            ▼                  ┌─ASC──┐   │  │
                    └─ORDER BY─────┬─column_name─┬──┼──────┼───┴──┘
                                   └─integer─────┘  └─DESC─┘
```

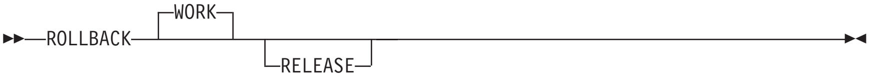


fullselect:



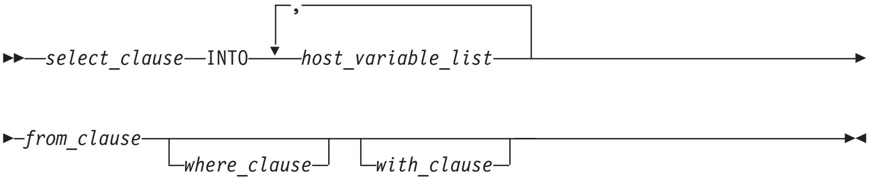
ROLLBACK (I,P)

Ends a logical unit of work without committing any changes.



SELECT INTO (P)

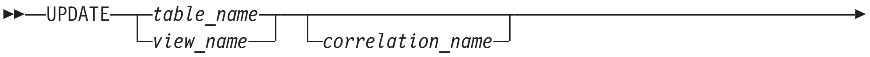
Produces a result table consisting of one row, and assigns the values in that row to host variables.

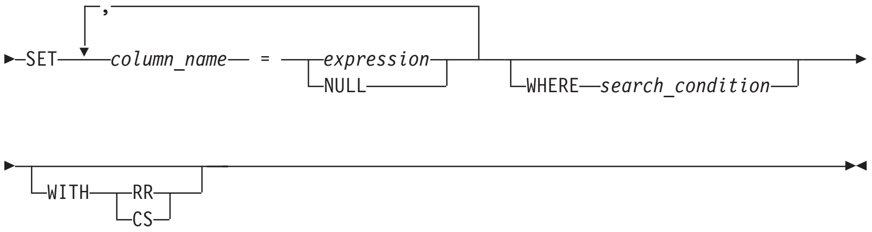


UPDATE (I,P)

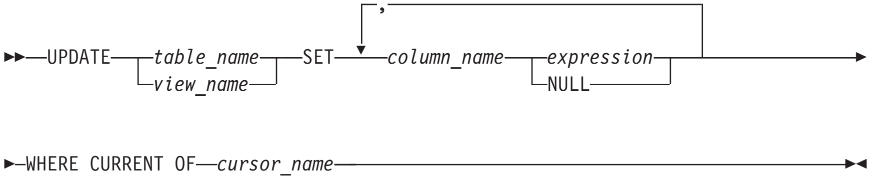
Updates one or more column values in one or more rows of a table or view. Updating a row of a view updates a row of its base tables.

Searched update (I,P)



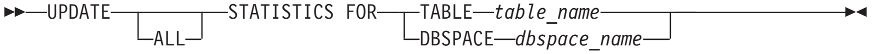


Positioned update (P)



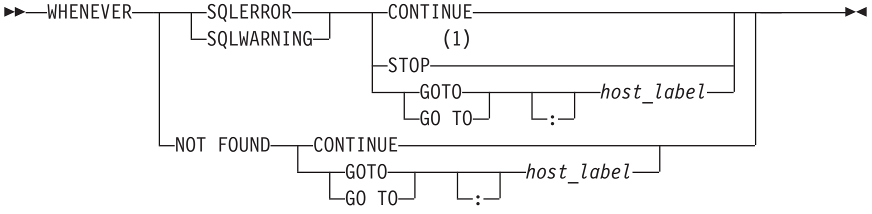
UPDATE STATISTICS (I,P)

Updates the statistics in the catalog about the tables and indexes.



WHENEVER (P)

Specifies the action to be taken when a specified exception condition occurs.



Notes:

1 STOP is not valid for C and FORTRAN.

Chapter 5. Preprocessing the Program

Program Preparation Command - VM Users

```
►►──SQLPREP──┬─ASM─────┬──PrepParm──────────────────────────────────────────►
             ├─C───────┤
             ├─COBol───┤
             ├─FORTran─┤
             └─PLI─────┘

►──(──PREPname=──┬────────────────┬─package_id──────────────────────────────►
                 └─collection_id.─┘

►──┬──────────────────────────────────────┬─────────────────────────────────►
   ├─,PrepFile=──(──┤ fileparms ├──)──────┤
   └─┤ prepparms ├────────────────────────┘

►──┬────────────────────────────────────────────┬──)────────────────────────►
   └─,USERid=──authorization_name/password──────┘

►──┬──────────────────────────────────────┬─────────────────────────────────►
   └─sysIN──(──┬─┤ fileparms ├─┬──)───────┘
               └─Reader────────┘

►──┬─────────────────────────────────────────┬──────────────────────────────►
   └─sysPRint──(──┬─┤ fileparms ├─┬──)───────┘
                  ├─Printer───────┤
                  └─Terminal──────┘

►──┬─────────────────────────────────────────┬──────────────────────────────►
   └─sysPUnch──(──┬─┤ fileparms ├─┬──)───────┘
                  └─Punch─────────┘

                                     (2)  (1)
►──┬─┤ multiple-user-mode-parms ├──────────────┬───────────────────────────►◄
   │                                 (1)       │
   └─┤ single-user-mode-parms ├────────────────┘
```

Notes:

1 Valid for DB2 Server for VM only.

2 Optional for multiple-user-mode.

fileparms:

```
├──filename──┬──────────────────────────────┬────────────────────────────────────┤
             └─filetype──┬────────────────┬─┘
                         └─filemode───────┘
```

prepparms:

```
   ┌─,APOST──────┐  ┌─,NOBIND──────┐  ┌─,NOBLocK───┐
├──┼─────────────┼──┼──────────────┼──┼────────────┼──────────────────────────►
   │        (1)  │  │        (4)   │  ├─,BLocK─────┤
   └─,Quote──────┘  └─,BIND────────┘  │        (2) │
                                      └─,SBLocK────┘

►──┬────────────────────────────────────┬──┬──────────────────────────────────┬──►
   └─,CCSIDGraphic──(integer)───────────┘  └─,CCSIDMixed──(integer)───────────┘

►──┬────────────────────────────┬──┬─────────────────────────────────────┬──────►
   └─,CCSIDSbcs──(integer)──────┘  └─,CHARSUB──(──┬─Sbcs────┬──)────────┘
                                                  ├─Mixed───┤
                                                  └─Bit─────┘

   ┌─,NOCHECK──┐
►──┼───────────┼──┬─────────────┬──┬──────────────┬─────────────────────────────►
   ├─,CHECK────┤  │        (1)  │  │        (1)   │
   └─,ERROR────┘  └─,COB2───────┘  └─,COBRC───────┘

   ┌─,CTOKEN──(NO)──────────────────┐                                        ┌─,NOEXIST──┐
►──┼────────────────────────────────┼──┬──────────────────────────────────┬──┼───────────┼──►
   └─,CTOKEN──(──┬─NO───┬──)────────┘  └─,DATE──(──┬─EUR─────┬──)─────────┘  └─,EXIST────┘
                 └─YES──┘                          ├─ISO─────┤
                                                   ├─JIS─────┤
                                                   ├─LOCAL───┤
                                                   └─USA─────┘

   ┌─,EXPLAIN──(NO)──────────────────┐
►──┼─────────────────────────────────┼──┬──────────────┬──┬──────────────┬──────┤
   └─,EXPLAIN──(──┬─NO───┬──)────────┘  │        (3)   │  │        (5)   │
                  └─YES──┘              └─,NOFOR───────┘  └─,DYNALC──────┘
```

Notes:

1 COBOL only.

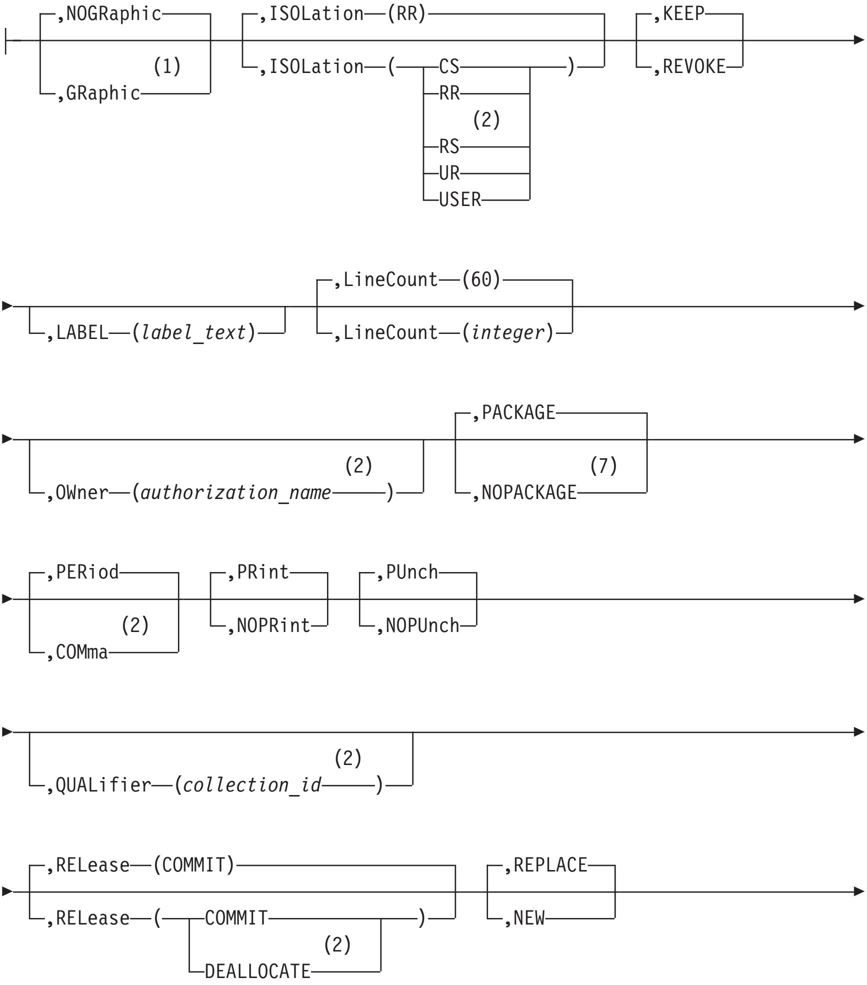
2 Not meaningful for DB2 Server for VSE.

3 Implied if STDSQL(89) is specified.

4 Valid for DB2 Server for VSE only.

5 COBOL, PL/I, C and ASSEMBLER only.

prepparms (continued):



```
   ┌─,SEQuence─────┐  ┌─,SQLApost─────────┐
►──┼───────────────┼──┼───────────────────┼──┬─────────────┬───────────►
   │          (3)  │  │       (2)   (4)   │  │        (5)  │
   └─,NOSEQuence───┘  └─,SQLQuote─────────┘  └─,NOSQLCA────┘
```

```
►──┬─────────────────────────────────────────────┬───────────────────────►
   └─,SQLFLAG──(──┬─SAA────────────────────┬──)──┘
                  └─89──┬───────────────┬──┘
                        └─(COMPLETE)────┘
```

```
   ┌─,STDSQL──(NO)──────────────┐
►──┼────────────────────────────┼────────────────────────────────────────►
   └─,STDSQL──(──┬─NO──────┬──)─┘
                 │     (6) │
                 └─89──────┘
```

```
►──┬──────────────────────────────┬──────────────────────────────────────┤
   └─,TIME──(──┬─EUR────┬──)──────┘
               ├─ISO────┤
               ├─JIS────┤
               ├─LOCAL──┤
               └─USA────┘
```

Notes:

1 COBOL and PL/I only.

2 Only meaningful for a non-DB2 Server for VM or -DB2 Server for VSE application server.

3 C only.

4 COBOL only.

5 Implied if STDSQL(89) is specified.

6 86 is a synonym for 89.

7 Valid for DB2 Server for VSE only.

multiple-user-mode-parms:

```
├──┬─DBFile──(──┤ fileparms ├──)─────┬───────────────────────────────────┤
   │            ┌────────────────┐   │
   │            ▼                │   │
   └─DBList──(──────server_name──┴─)─┘
```

single-user-mode-parms:

```
├──Dbname──(server_name)──┬────────────────────┬──────────────────────────►
                          └─dcssID──(dcss_id)──┘

►──┬────────────────────────────────────────────────────────────┬───────────────┤
   └─LOGmode──(──┬─A─┬──)──┬──────────────────────────┬──┘
                 ├─L─┤     └─PARMID──(filename)──┘
                 ├─N─┤
                 └─Y─┘
```

Program Preparation Command - VSE Users

Multiple User Mode

```
                             (1)
►►──//──EXEC──PGM=ARIPRPx──────┬─────────────┬────────────────────────────────►
                               └─,SIZE=AUTO─┘

►─,PARM=──' ──PREPname=──┬─────────────────┬─package_id───────────────────────►
                         └─collection_id.─┘

►─,USERid=──authorization_name/password──┬────────────────────────────┬───────►
                                         │                    (2)     │
                                         └─DBNAME=──server_name───────┘

►──┬────────────────────────────────────────────────┬─' ──────────────────────►◄
   │                               ┌─.A──────────┐  │
   ├─PrepFile=──(──membername──────┼─────────────┼──)─┤
   │                               └─.membertype─┘  │
   │             (3)                                │
   └─| prepparms |──────────────────────────────────┘
```

Notes:

1 The x represents A for assembler, B for C, C for COBOL, F for FORTRAN, and P for PL/I.

2 From 1 to 18 characters, and identifies the DBNAME for the application server.

3 Same as prepparms for VM on page “Program Preparation Command - VM Users” on page 61.

Single User Mode

```
►►──//──EXEC──PGM=ARISQLDS──,SIZE=AUTO────────────────────────────────────────►

          ┌──────────────────────────────┐
          ▼                              │
►─,PARM=─────┬──────────────────────────┬┴─────────────────────────────────────►
             └─initialization_parameter,┘

                       (1)
►─PROGNAME=ARIPRPx──────────,PARM=──'─────────────────────────────────────────►

►─PREPname=──┬───────────────┬─package_id─────────────────────────────────────►
             └─collection_id.┘

►─,USERid=──authorization_name/password─┬─────────────────────────────┬──────►
                                        │                         (3) │
                                        └─DBNAME=──server_name────────┘

►─┬──────────────────────────────────────────────────┬─'──────────────────►◄
  │                                   ┌─.A──────────┐│
  ├─PrepFile=──(──membername──────────┼─────────────┼──)──┤
  │                                   └─.membertype─┘│
  │                (3)                               │
  └─| prepparms |────────────────────────────────────┘
```

Notes:

1. x represents A for assembler, B for C, C for COBOL, F for FORTRAN and P for PL/I.

2. Same as prepparms for VM on page "Program Preparation Command - VM Users" on page 61.

3. From 1 to 18 characters, and identifies the DBNAME for the application server.

Program BIND Command - VSE Users

```
►►──CBND──PACKAGE(──┬───────────────┬──┬────────────┬──┬───┬──────────────────►
                    └─collection_id.┘  └─package_id─┘  └─)─┘
```

```
►──┬────────────────────────────────┬───────────────────────────────────────►
   └─ INfile──(──private_file_id──)──┘
```

```
►──┬──────────────────────────────────────────────────────────────┬──────────►◄
   │   ┌──────────────────────────────────────────────────────┐   │
   │   ▼                                                      │   │
   ├──────┬─┬─NEW──────────────────────┬────────────────────┬─┴───┤
   │      │ └─REPLACE──┬────────────┬──┘                    │     │
   │      │            ├─KEEP───────┤                       │     │
   │      │            └─REVOKE─────┘                       │     │
   │      ├─OWner──(──authorization_name──)─────────────────┤     │
   │      ├─QUALifier──(──collection_id──)──────────────────┤     │
   │      ├─┬─NOCHECK──┬────────────────────────────────────┤     │
   │      │ ├─CHECK────┤                                    │     │
   │      │ └─ERROR────┘                                    │     │
   │      │             ┌──────────────────┐                │     │
   │      │             ▼                  │                │     │
   │      ├─DBList──(─────server_name──────┴──)─────────────┤     │
   │      ├─USERid──(──authorization_name/password──)───────┤     │
   │      └─MSGQueue──(──name──)────────────────────────────┘     │
   └─QRY──────────────────────────────────────────────────────────┘
```

Note: For any options not specified on the CBND transaction, the default option will be the option specified when the package was preprocessed, unless otherwise noted.

Chapter 6. Interactive SQL Commands

Starting and Stopping ISQL - VM Users

To use ISQL in a VM environment, follow these steps:

1. Log on to VM.
2. IPL CMS.
3. Start ISQL.

To start ISQL, type the following 4-character transaction identifier and press ENTER:

```
ISQL
```

To stop communication with the DB2 Server for VM database manager through ISQL, type the following command in the input area and press ENTER:

```
EXIT
```

To end your VM terminal session, type the following command and press ENTER.

```
LOGOFF
```

Starting and Stopping ISQL - VSE Users

To start ISQL, type the following 4-character transaction identifier and press ENTER:

```
ISQL
```

If your installation has not defined a default authorization name, enter the authorization-name and password on the ISQL welcome panel, or use the explicit SQL CONNECT statement as follows:

```
CONNECT authorization-name IDENTIFIED BY password
```

To stop communication with the DB2 Server for VSE database manager through ISQL, type the following command in the input area and press ENTER:

```
EXIT
```

BACKOUT

Nullifies changes made since the last SAVE command or, if no previous SAVE command was issued, since the start of the INPUT command. Use this command only while applying the INPUT command with AUTOCOMMIT on.

```
►►─BACKOUT─────────────────────────────────────────────────────────────►◄
```

BACKWARD

Displays rows that occur before those that are currently displayed.

```
►►─BACKWARD──┬──────────────┬──────────────────────────────────────────►◄
             ├─rows_integer─┤
             └─MAX──────────┘
```

CANCEL

Cancels a command, an SQL statement, or a logical unit of work that is in progress.

```
►►─CANCEL──────────────────────────────────────────────────────────────►◄
```

CHANGE

Modifies the current SQL statement in the command buffer and displays the result.

```
►►─CHAnge─/─replaced_string─/─┬────────────────────┬───────────────────►◄
                              └─replacing_string─/─┘
```

COLUMN

Displays the result of the interactive select-statement so that it begins with the specified column at the left edge of the display.

```
            ┌─1──────────────────────┐
►►─COLumn───┼────────────────────────┼─────────────────────────────────►◄
            └─position_in_select_list┘
```

DISPLAY

Shows the results of the associated interactive select-statement on the display. This command can only be issued from a routine.

```
►►──DISPLAY──────────────────────────────────────────────────────────►◄
```

END

Ends the display of either a SELECT, COUNTER, SHOW or INPUT command, or a DISPLAY command in a routine.

```
►►──END──────────────────────────────────────────────────────────────►◄
```

ERASE

Erases stored SQL statements.

```
            ┌────────────────────────┐
            ▼                        │
►►──ERASE─────stored_statement_name──┴───────────────────────────────►◄
```

EXIT

Ends the current ISQL terminal session.

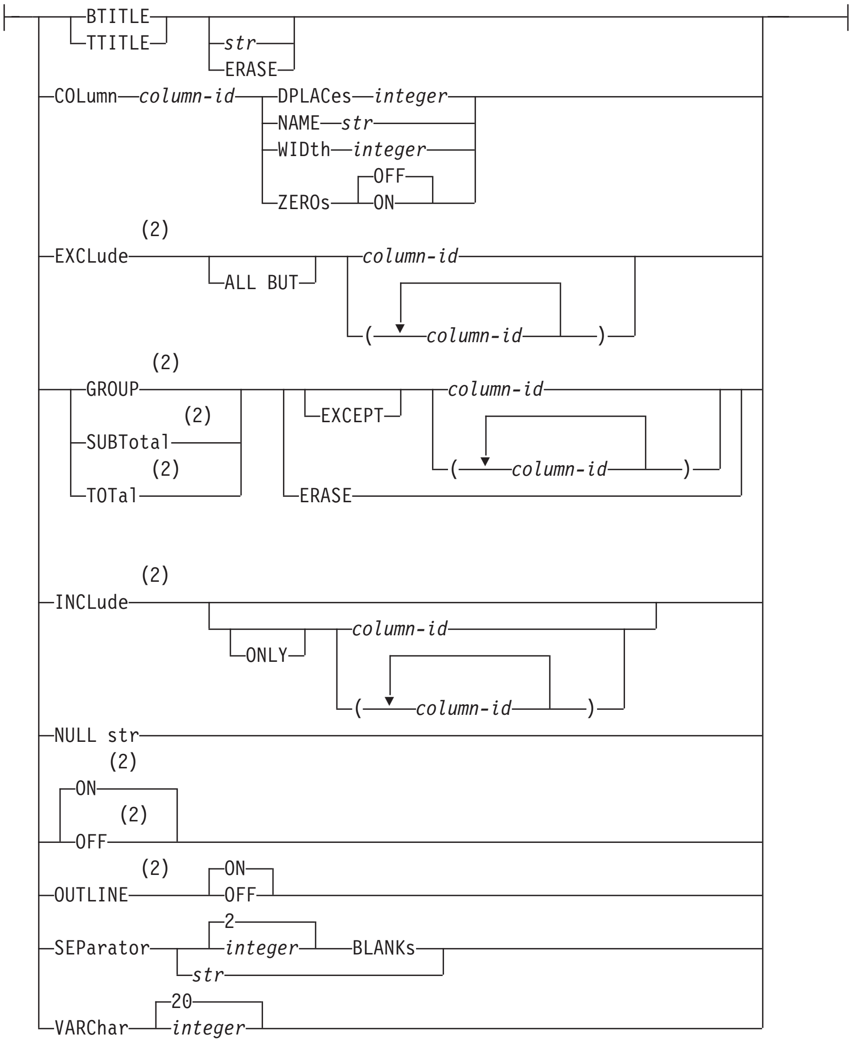
```
►►──EXIT─────────────────────────────────────────────────────────────►◄
```

FORMAT

Controls the format of the query result that is displayed.

```
                   ┌─────────────────────┐
         (1)       ▼                     │
►►──FORMAT──────────┤ Format options ├───┴───────────────────────────►◄
```

Format options:



Notes:

1 Formatting can be performed only on the first 45 columns of a query result.

2 Any formatting command containing the keywords EXCLude, GROUP, INCLude, ON/OFF, OUTLINE, SUBTotal, or TOTal, causes the query to be reexecuted.

FORWARD

Moves your display forward through a query result.

```
►►──FORWARD──┬────────────────┬──────────────────────────────────────►◄
             ├─rows_integer───┤
             └─MAX────────────┘
```

HELP

Retrieves online documentation (available as an option during installation of the database manager).

```
►►──HELP──┬───────────────┬──────────────────────────────────────────►◄
          ├─CONTENTs──────┤
          └─topic_name────┘
```

HOLD

Prevents an SQL statement from being processed when it is entered.

```
►►──HOLD──sql_statement──────────────────────────────────────────────►◄
```

IGNORE

Nullifies a partially entered, multiple line command.

```
►►──IGNORE───────────────────────────────────────────────────────────►◄
```

INPUT

Inserts one or more rows of data into a table or view.

```
►►──INPut──┬─table_name─┬──┬─────────────────────────────┬───────────►◄
           └─view_name──┘  │      ┌─,──────────────┐      │
                           │      ▼                │      │
                           └─(────┴─column_name──)─┴──────┘
```

Interactive Select

The Interactive Select statement retrieves data from a table.

interactive-select-statement

```
►►─┤ fullselect ├──┬──────────────────────────────────────────────────┬─────►
                   │             ┌─,───────────────────────────┐      │
                   │             ▼                             │      │
                   └─ORDER BY────┬─column_name─┬──┬─ASC──┬──────┴──────┘
                                 └─integer─────┘  └─DESC─┘

►──┬─────────────────┬──────────────────────────────────────────────────────►◄
   └─WITH──┬─RR─┬────┘
           ├─CS─┤
           └─UR─┘
```

fullselect:

```
            ┌──────────────────────────────────────────────────────────┐
            ▼                                                          │
├──┬─subselect────┬───┬───────────────────────────────────────────────┬┴───────┤
   └─(fullselect)─┘   └──┬─UNION─────┬──────┬─subselect────┬──────────┘
                         └─UNION ALL─┘      └─(fullselect)─┘
```

subselect:

```
              ┌─ALL──────┐
├──SELECT─────┼──────────┼──┬─*───────────────────────────────────┬──────────►
              └─DISTINCT─┘  │    ┌─,──────────────────────────┐  │
                            │    ▼                            │  │
                            └────┬─expression─────────────────┴──┘
                                 ├─table_name.*──────────┤
                                 ├─view_name.*───────────┤
                                 └─correlation_name.*────┘

             ┌─,──────────────────────────────────────────────┐
             ▼                                                │
►──FROM──────┬─table_name─┬───┬──────────────────┬────────────┴─────────────►
             └─view_name──┘   └─correlation_name─┘
```

```
►─┬──────────────────────────┬──┬──────────────────────────────┬──────────►
  └─WHERE──search_condition──┘  │              ┌─,──────────┐  │
                                └─GROUP BY─────▼──column_name─┴──┘

►─┬──────────────────────────┬────────────────────────────────────────────┤
  └─HAVING──search_condition─┘
```

ISQLTRACE

Traces activities within ISQL.

```
►►──ISQLTrace──┬─DUMP────┬────────────────────────────────────────────►◄
               ├─DISPLAY─┤
               └─integer─┘
```

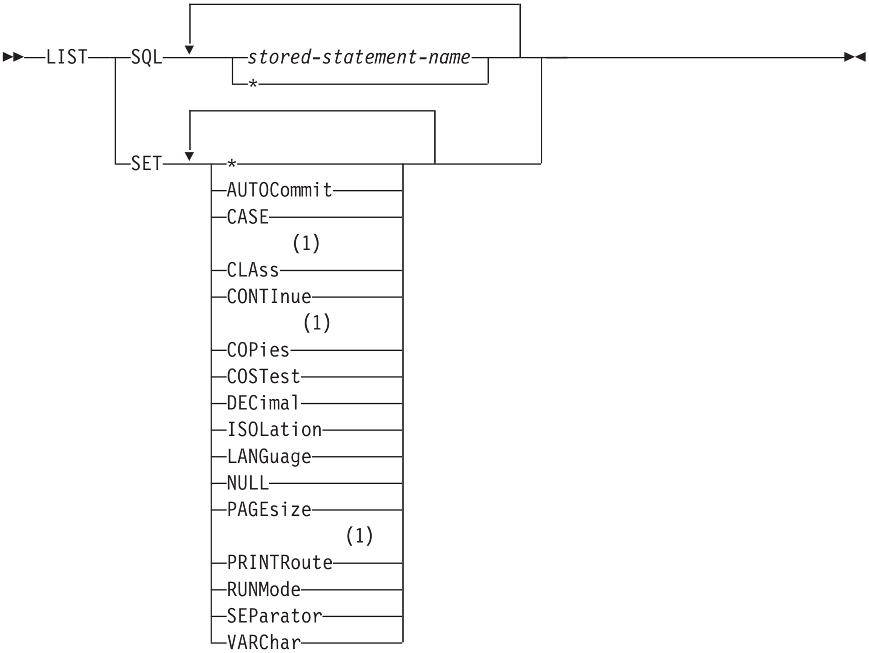
LEFT

Starts the display *integer* columns to the left, counting from the leftmost column on the display.

```
           ┌─1───────┐
►►──LEFT───┤         ├────────────────────────────────────────────────►◄
           └─integer─┘
```

LIST

Lists information about stored SQL statements, or lists the settings of certain operational characteristics set by the SET command.

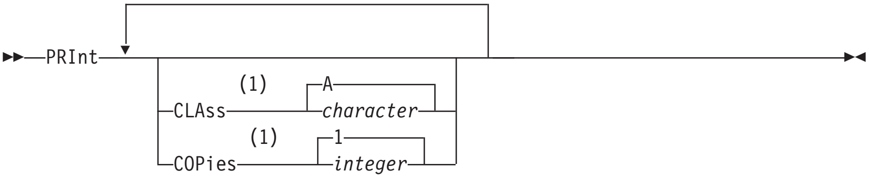


Notes:

1 The parameters CLAss, COPies, and PRINTRoute are not applicable to VM.

PRINT - VM Users

Requests printed copies of a query result by sending it to the system printer.



Notes:

1 The value for CLASS and COPIES specified in the PRINT command remain in effect until changed by a subsequent PRINT or CP SPOOL command.

PRINT - VSE Users

Requests printed copies of a query result by sending it to the system printer, POWER remote printer, or CICS/VSE terminal.

```
          ┌─────────────────────────────┐
          ▼                             │
►►──PRInt───┬─────────────────────────┬─┴─────────────────────────────────────►◄
            │           ┌─?─────────┐ │
            ├─CLAss─────┴─character─┴─┤
            │            ┌─1───────┐  │
            ├─COPies─────┴─integer─┴──┤
            ├─TERMid──termid──────────┤
            ├─DESTid──wkstat──────────┤
            ├─SYStem──────────────────┤
            └─TOUSER──┬──────────┬────┘
                      └─userid───┘
```

RECALL

Retrieves a stored SQL statement.

```
►►──RECALL──┬───────────────────────┬─────────────────────────────────────────►◄
            ├─stored_statement_name─┤
            └─PREVIOUS──────────────┘
```

RENAME

Renames a stored SQL statement.

```
►►──RENAME──old_stored_statement_name─────────────────────────────────────────►◄


►►──new_stored_statement_name─────────────────────────────────────────────────►◄
```

RIGHT

Starts the display *integer* columns to the right, counting from the leftmost column of the display.

```
          ┌─1───────┐
►►──RIGHT─┼─────────┼─────────────────────────────────────────────────────────►◄
          └─integer─┘
```

RUN

Initiates the processing of a routine. Multiple parameters are separated by blanks. Enclose a parameter in single quotation marks if it contains a blank.

```
>>-RUN--routine_name--+-----------------------------+----------><
                      |     .--------------.        |
                      |     V              |        |
                      '-(-------parameter--+----)---'
```

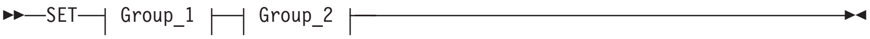
SAVE

Saves all changes since the last SAVE command or, if no SAVE command was issued, since the start of the INPUT command. SAVE is used while you are using the INPUT command with AUTOCOMMIT on only.

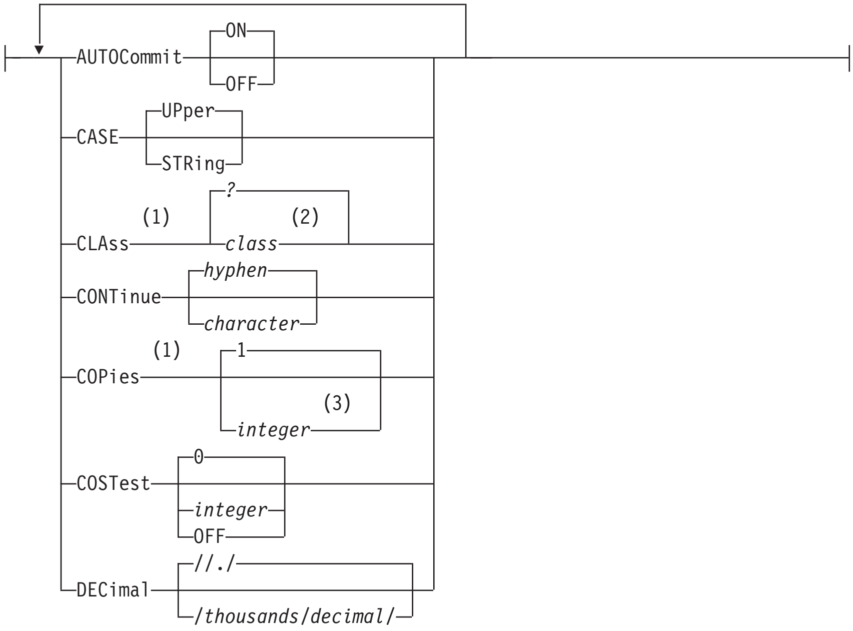
```
>>-SAVE--------------------------------------------------------><
```

SET

Sets specified operational characteristics.



Group_1:



Notes:

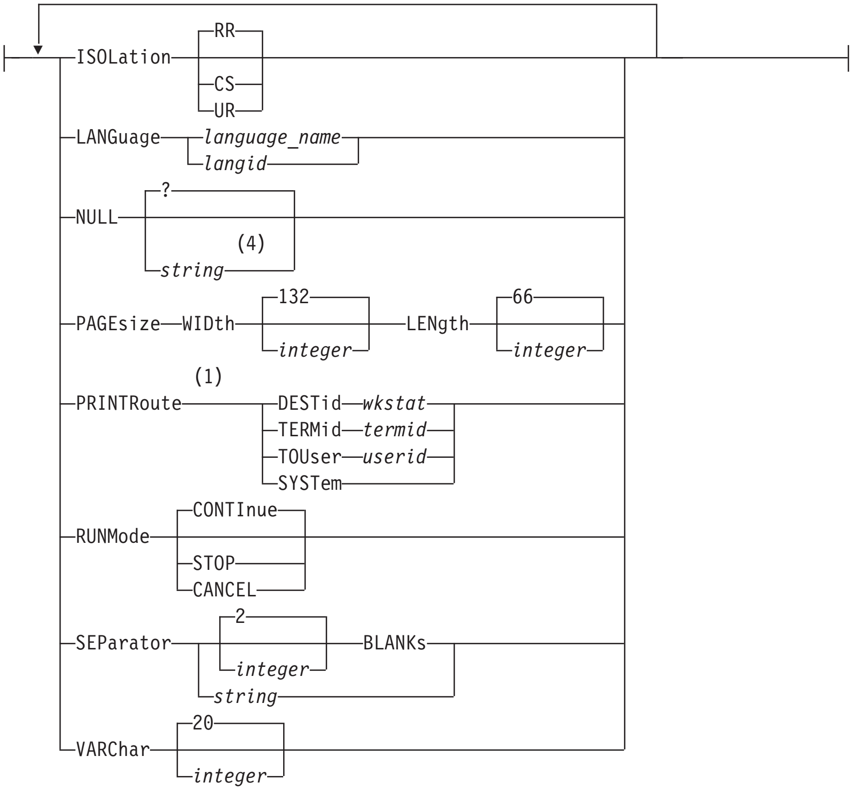
1 The parameters CLAss, COPies, and PRINTRoute are not applicable to VM.

2 Output class wanted (letters from A to Z).

3 Number of copies to be printed.

4 A maximum of 20 characters can be used for null values.

Group_2:



Notes:

1 The parameters CLAss, COPies, and PRINTRoute are not applicable to VM.

2 Output class wanted, (letters from A to Z).

3 Number of copies to be printed.

4 A maximum of 20 characters can be used for null values.

START

Processes the current SQL statement or a stored SQL statement.

```
►►──START──┬────────────────────────┬──┬─────────────────────────────┬──────────►◄
           └─stored_statement_name──┘  │    ┌──────────────────┐     │
                                       │    ▼             (1)  │     │
                                       └─(──────parameter──────┴──)──┘
```

Notes:

1 Enclose a parameter in single quotation marks when it contains a blank.

STORE

Saves the current SQL statement for later use. The statement remains stored until erased.

```
►►──STORE──stored_statement_name──┬─────────┬────────────────────────────────────►◄
                                  └─REPLACE─┘
```

Notes:

1 Related display formatting information can also be stored with the statement, however, some limitations exist. Refer to the *DB2 Server for VSE & VM Interactive SQL Guide and Reference* manual for more information about storing display information.

TAB

Displays all characters of a column that are too wide to fit on the display.

```
        ┌─1───────┐
►►──TAB─┼─────────┼──────────────────────────────────────────────────────────────►◄
        └─integer─┘
```

Notes:

1 The TAB command is valid only for CHAR and VARCHAR columns.

ISQL Program Function Keys

The following default Program Function (PF) keys are provided through ISQL.

PF1, PF13 Issues a HELP command, which retrieves an explanation of the use of online help information and provides a list of topics available.

PF2, PF14 Issues a START command, which starts the command in the SQL command buffer (the **current** SQL command).

PF3, PF15 Issues an END command, which ends the display of a query.

PF4, PF16 Issues a PRINT command, which requests the currently displayed query result to be printed on the system or workstation printer.

PF5, PF17 Issues a RECALL command, which displays the contents of the SQL command buffer.

PF6, PF18 Not assigned.

PF7, PF19 Issues a BACKWARD command.

PF8, PF20 Issues a FORWARD command, which can be used to scroll through the query result half a screen at a time.

PF9, PF21 Issues a HOLD command, which prevents an SQL command from being processed when it is entered.

PF10, PF22 Issues a LEFT 1 command, which moves your view of the query result one column to the left.

PF11, PF23 Issues a RIGHT 1 command, which moves your view of the query result one column to the right.

PF12, PF24 Performs the RETRIEVE function, which moves the previously entered line into the input area.

Note: The CP SET command can be used to set the PF key functions.

CMS Subset VM Users

This CMS subset section applies to VM users. There is no equivalent section for VSE.

CMS or CP commands can be entered during an ISQL session. Supported commands from the CMS subset environment can be entered without terminating the ISQL session. To enter the CMS subset environment, type CMS on the command line, and press ENTER. Refer to Figure 1.

While in the CMS subset environment, you should not use any commands, programs, or execs that access the database manager. If they are used, the results are unpredictable and error conditions could occur.

To exit from the CMS environment, type RETURN in the command line, and press ENTER.

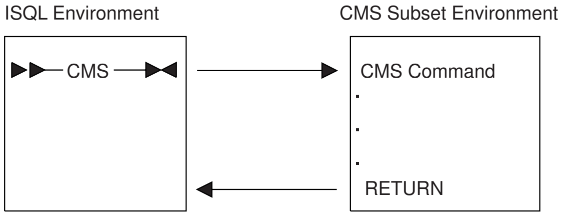


Figure 1. Using the CMS Subset Environment

Chapter 7. Operator Commands

The following operator commands can be used within ISQL, or entered from the database machine operator console. For the complete set of operator commands, see the *DB2 Server for VSE & VM Operation* manual.

COUNTER

Used primarily to monitor system performance.

```
►►──COUNTER──┬─*──────────┬──────────────────────────────────────────►◄
             │ ┌────────┐ │
             │ ▼        │ │
             └───name───┴─┘
```

Valid names are:

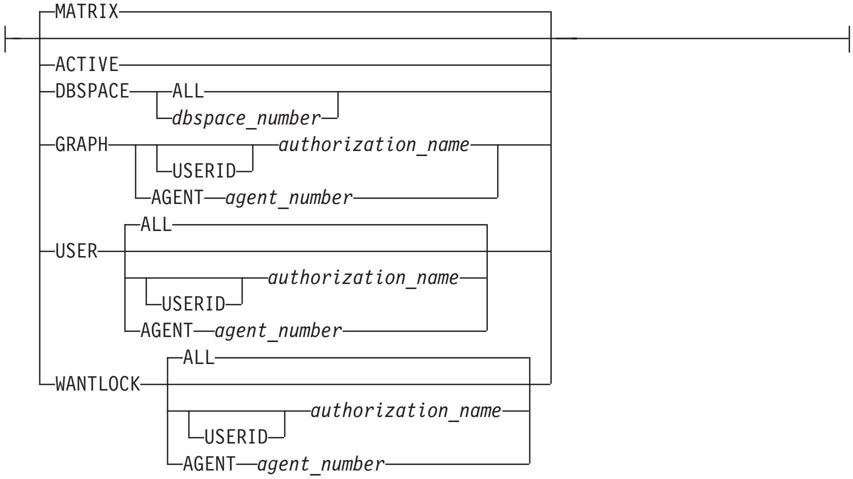
BEGINLUW
CHKPOINT
DASDIO
DASDREAD
DASDWRIT
DBSSCALL
DEADLCK
DIRREAD
DIRWRITE
ESCALATE
LDIRBUFF
LOCKLMT
LOGREAD
LOGWRITE
LPAGBUFF
PAGEREAD
PAGWRITE
RDSCALL
ROLLBACK
WAITLOCK

SHOW

Monitors system activity.

```
►►──SHOW──┬─ACTIVE────────────────────────────────┬───────────────────────►◄
          ├─ADDRESS──module_name──────────────────┤
          ├─BUFFERS───────────────────────────────┤
          │         ┌─ALL─────────────────────┐   │
          ├─CONNECT─┼─────────────────────────┼───┤
          │         ├─┬──────────┬──userid1───┤   │
          │         │ └─USERID───┘            │   │
          │         ├─AGENT──agent_no─────────┤   │
          │         ├─LUWID──luwid1───────────┤   │
          │         ├─ACTIVE──────────────────┤   │
          │         ├─INACTIVE────────────────┤   │
          │         │          (1)            │   │
          │         └─WAITING─────────────────┘   │
          ├─DBCONFIG──────────────────────────────┤
          ├─DBEXTENT──────────────────────────────┤
          ├─DBSPACE──dbspace_number───────────────┤
          ├─INVALID───────────────────────────────┤
          ├─LOG───────────────────────────────────┤
          │         ┌─ALL─────┐                   │
          ├─LOGHIST─┴─integer─┴──┬─────────────┬──┤
          │                      └─SERVICE─────┘  │
          ├─LOCK──┤ lock parameters ├─────────────┤
          │      ┌─ALL─────────────┐              │
          ├─POOL─┼─────────────────┼──────────────┤
          │      ├─SUMMARY─────────┤              │
          │      ├─DELETED─────────┤              │
          │      └─pool_number─────┘              │
          ├─STORAGE───────────────────────────────┤
          ├─SYSTEM────────────────────────────────┤
          │      (2)                              │
          └─USERS─────────────────────────────────┘
```

lock parameters



Notes:

1 The SHOW CONNECT WAITING is not supported in the VSE environment.

2 The keyword USERS is not applicable to a VSE system.

Chapter 8. Database Services Utility Commands

The Database Services Utility (DBS Utility) processes commands that are unique to the DBS Utility and certain SQL statements.

Starting and Stopping the DBS Utility

Before using the DB2 Server for VM or DB2 Server for VSE database manager for any application, initialize your database. The method of initialization depends on whether the DBS Utility is run in multiple user mode or single user mode.

Starting the DBS Utility - VM Users

SQLINIT EXEC

This portion of the SQLINIT command should be issued before the SQLDBSU EXEC to initialize DB2 Server for VM database access and load the multiple user support system routines.

```
►►──SQLINIT──Dbname──(dbname)──┬─────────────────────┬──────────────────►◄
                               └─dcssID──(dcss_id)───┘
```

For more information about using the SQLINIT EXEC, see the *DB2 Server for VSE & VM Database Administration* manual.

SQLDBSU EXEC

Used to invoke the DBS Utility in single or multiple user mode.

```
►►──SQLDBSU──┬──────────────────────────────────────┬────────────────────►
             └─sysIN──(──┬─Reader──────────┬─)──────┘
                         ├─Terminal────────┤
                         └─┤ fileparms ├───┘

                                                            (1)
►─┬─────────────────────────────────────────┬─Dbname──(server_name)──────►
  └─sysPRint──(──┬─Printer─────────┬─)──────┘
                 ├─Terminal────────┤
                 └─┤ fileparms ├───┘
```

```
                                  (1)                                  (1)
►──┬──────────────────────┬───────┬─────────────────────────────────┬────────►
   └─dcssID──(dcss_id)────┘       └─LOGmode──(──┬──A──┬──)──────────┘
                                                ├──L──┤
                                                ├──N──┤
                                                └──Y──┘
```

```
                                        (1)
►──┬────────────────────────────────┬─────────────────────────────────────►◄
   └─PARMID──(──file_name──)────────┘
```

fileparms:

```
├──file_name──┬───────────────────────────────────┬───────────────────────────┤
              └─file_type──┬──────────────────┬───┘
                           └─file_mode────────┘
```

Notes:

1 Only applicable to single user mode.

Exiting from the DBS Utility - VM Users

Batch Mode

If a control file is supplied with the SYSIN option, an exit is made from the utility automatically after all the commands in the control file are processed.

Interactive Mode

If a control file is not supplied, the DBS Utility is used interactively. To exit, type the following command, and press ENTER.

```
EXIT;
```

Any uncommitted work is committed, and an exit is made from the utility to the conversational monitor system (CMS).

Starting the DBS Utility - VSE Users

Multiple User Mode

To invoke the DBS Utility, use the following EXEC statement as part of the JCL:

```
// EXEC PGM=ARIDBS,SIZE=AUTO
```

Single User Mode

To invoke the DBS Utility, use either:

```
// EXEC PROC=ARISDBSD
```

or

```
// EXEC ARISQLDS,SIZE=AUTO,                    X
//   PARM='SYSMODE=S,LOGMODE=N,PROGNAME=ARIDBS'
```

Exiting from the DBS Utility - VSE Users

The DBS Utility automatically ends after all commands in the control file are processed; that is, when /* is encountered.

COMMENT

Documents input by supplying the Database Services Utility COMMENT commands at appropriate points within the Database Services Utility control command input stream. The utility displays the comments in the Database Services Utility message file listing.

```
►►──COMMENT──'string_constant'─────────────────────────────────────►◄
```

CREATE SCHEMA

The sequential SCHEMA input file contains one CREATE SCHEMA statement, which is the first statement in the file.

```
                                              ┌─;─┐
►►──CREATE SCHEMA AUTHORIZATION──authorization_id─┴───┴─────────────────►

   ┌────────────────────────────┐
   ▼                            │
►──┬──────────────────────────┬─┴──────────────────────────────────►◄
   │                ┌─;─┐     │
   └─schema_statement─┴───┴─────┘
```

DATALOAD

Loads or adds rows into existing tables from data in a sequential input file. DATALOAD and its subcommands are on more than one input record with each record in general containing data for a table row. Refer to the syntax diagram on page 93.

Table-Column-ID-Subcommand (TCI)

Identifies the location of the input records of the data for a table column. Each table-column-id-subcommand occupies a separate input record. Data must be in the same record positions in all records that relate to the table. Refer to the syntax diagram on page 93.

```
►►──DATALOAD TABLE──(──table_name──)──────────────────────────────────────────►

                                                                (5)
►──┬────────────────────────────────────────────────────────────┬────────────►
   └─IF──POS──(──startpos──┬───────────────┬──)──┬─ = ──┬──constant─┘
                           │           (2) │     ├─ <> ─┤
                           └─-endpos───────┘     ├─ ^= ─┤
                                                 ├─ < ──┤
                                                 ├─ > ──┤
                                                 ├─ <= ─┤
                                                 └─ >= ─┘

    ┌───────────────────────────────────┐
    ▼                                   │
►───┤ table-column-id-subcommand ├──────┴────────────────────────────────────►

►─┤ infile-subcommand ├──┬─────────────────────────────────────────┬──────────►◄
                         │  ┌─────────────────────┐                │
                         │  ▼                 (3) │        (4)     │
                         └──user_data_records─────┴──ENDDATA───────┘
```

table-column-id-subcommand:

```
    ┌──────────────────────────────────────────────────────────────────────┐
    ▼                                      ┌─CHARacter─┐                   │
├───col_name──startpos──┬─────────────┬────┼───────────┼──┬──────────────────────┬─┴──────┤
                        └─-endpos─────┘    └─data_type─┘  │             (1) (6)  │
                                                          └─┤ nc-clause ├────────┘
```

nc-clause:

```
   (1)                       (6)
├─────────┬─CURRENT DATE──────┬──────────────────────────────────────────────►
          ├─CURRENT TIME──────┤
          ├─CURRENT TIMESTAMP─┤
          └─NULL──────────────┘

                                                                  (6)
►──┬──────────────────────────────────────────────────────────┬──────────────┤
   │ ┌─IF─┐                                                    │
   └─┴────┴──POS──(──startpos──┬─────────────┬──)──┬─=──┬──constant─┘
                               └─-endpos─────┘     ├─<>─┤
                                                   ├─^=─┤
                                                   ├─<──┤
                                                   ├─>──┤
                                                   ├─<=─┤
                                                   └─>=─┘
```

Notes:

1 nc-clause is the short form for null-current-clause.

2 No blanks are allowed between startpos, hyphen, and endpos.

3 The *user-data-records* contain the data referenced by the preceding DATALOAD subcommands.

4 ENDDATA identifies the end of user-supplied data embedded in the control file. The command is valid if the previous DBS Utility command was an INFILE(*) subcommand.

5 These options must appear on the same physical line as DATALOAD TABLE.

6 This clause must appear on the same physical line.

Infile-subcommand

Identifies the file containing the data referenced by the preceding DATALOAD and TCI subcommands. When INFILE is followed by an asterisk, the data is in the control file and immediately follows the subcommand.

infile-subcommand - VM

```
►►──INFILE──────────────────────────────────────────────────────────────────►

                                                                       (3)
►──┬─(──*──┬────────────────────────────────┬─┤ options ├──────────────)──┬──►
   │       │                ┌─NO──┐    (3)  │                             │
   │       └─CONTINUED──(──┴─────┴──)──────┘                             │
   │                        └─YES─┘                                       │
   └─(ddname)──────────────────────────────────────────────────────────────┘

►──┬─────────────────────────┬──┬────────────────────────────┬──────────────►◄
   └─COMMITCOUNT──(ccount)──┘  └─RESTARTCOUNT──(rcount)────┘
```

options

```
├──┬──────────────────────────────────┬──────────────────────────────────┤
   │            ┌─YES──┐   (1)  (2)   │
   └─LIST──(───┴─NO───┴──)──────────┘
```

Notes:

1 No blanks are allowed between the keywords CONTINUED and LIST and the value specified for CONTINUED and LIST.

2 LIST can be specified before CONTINUED.

3 These options must appear on the same physical line as INFILE.

infile-subcommand - VSE

```
►►──INFILE──────────────────────────────────────────────────────────────────►
```

```
►──┬─(──*──┬──────────────────────────────────┬──┬───────────────────────────────┬──)─┬──►
   │       │          (1)  (2)      ┌─NO──┐   │  │       (1)  (2)     ┌─YES─┐    │    │
   │       └─CONTINUED──────────(──┴─YES─┴──)─┘  └─LIST─────────────(──┴─NO──┴──)─┘    │
   └─(──ddname──┬───────────────────────┬──)──────────────────────────────────────────┘
                └─┤ tape/disk options ├─┘

►──┬──────────────────────────┬──┬─────────────────────────────┬──────────────────────►◄
   └─COMMITCOUNT──(ccount)──┘  └─RESTARTCOUNT──(rcount)──┘
```

Notes:

1 No blanks are allowed between the keywords CONTINUED and LIST and the value specified for CONTINUED and LIST. LIST can be specified before CONTINUED.

2 LIST can be specified before CONTINUED.

tape/disk options:

```
├──┬───────────────────────┬──┬──────────────────────────────────────────┬──────►
   │          ┌─2048─┐     │  │                    ┌─REWIND───┐          │
   └─BLKSZ──(─┴─size─┴──)──┘  │          ┌─(TAPE)──┴─NOREWIND─┴─┐        │
                              └─PDEV──┴─(DASD)───────────────────┴───────┘

►──┬───────────────────────┬──┬──────────────────────┬──────────────────────┤
   └─RECFM──(format)──┘     └─RECSZ──(size)──┘
```

DATAUNLOAD

Selectively unloads data from tables and views to a user-defined sequential file of data.

Data-Field-Identification Subcommand

Identifies the location in the output record where the data for a column in the select-list parameter should be placed, and identifies the output record data-field data-type.

```
              (1)
►►──DATAUNLOAD──────────────────────────────────────────────────────────────►

                                    (1)  (2)
►──interactive_select_statement;──────────┬────────────────────────────────────────┬──►
                                          │ ┌──────────────────────────────────┐   │
                                          │ ▼                       (1)  (2)   │   │
                                          └───┤ data-field-id-subcommand ├─────┴───┘
```

```
►─┤ outfile-subcommand ├──────────────────────────────────────────────────►◄
```

Notes:

1 Each of these must be a separate record.

2 Both must appear on the same physical line.

data-field-id-subcommand (DFI):

```
├──┬─column_reference─┬─startpos──┬─────────┬──────────────────────────────►
   └─integer──────────┘           └─-endpos─┘

   ┌─CHARacter─┐
►──┼───────────┼───┬────────────────────┬──────────────────────────────────┤
   └─data_type─┘   └─┤ set-null-clause ├─┘
```

set-null-clause:

```
    ┌─IF─┐        ┌─SET─┐
├───┴────┴──NULL──┴─────┴──POS──(──startpos──┬─────────┬──)── = ──value────┤
                                             └─-endpos─┘
```

outfile-subcommand - VM:

```
            (1)
├──OUTFILE──────(ddname)───────────────────────────────────────────────────┤
```

Notes:

1 The outfile-subcommand identifies the sequential output file that contains the data referenced by the preceding DATAUNLOAD subcommands.

outfile-subcommand - VSE:

```
            (1)
├──OUTFILE──────(──ddname──┬────────────────┬──────────────────────────────►
                           └─BLKSZ──(size)──┘

►──┬───────────────────────────────────────┬──┬──────────────────┬─────────►
   │                ┌─NOREWIND─┐           │  └─RECFM──(format)──┘
   │         ┌─TAPE─┴─REWIND───┴──┐        │
   └─PDEV────┴─(DASD)─────────────┴────────┘
```

```
►─┬────────────────────┬──────────────────────────────────────────────┤
  └─RECSZ──(size)──────┘
```

Notes:

1 The outfile-subcommand identifies the sequential output file that contains the data referenced by the preceding DATAUNLOAD subcommands.

REBIND PACKAGE

Rebinds an existing package.

```
►►──REBIND PACKAGE──(package_name)───────────────────────────────────────►◄
```

RELOAD DBSPACE

Identifies a RELOAD DBSPACE request and identifies a DBSPACE to be loaded.

VM Users

```
►►──RELOAD DBSPACE──(dbspace_name)──┬─NEW───┬──INFILE──(ddname)──────────►
                                    └─PURGE─┘

►─┬────────────────────────────┬──┬──────────────────────────────────┬───►
  └─COMMITCOUNT──(──ccount──)──┘  └─RESTARTTABLE──(──table_name──)───┘

►─┬─────────────────────────────┬────────────────────────────────────────►◄
  └─RESTARTCOUNT──(──rcount──)──┘
```

VSE Users

```
►►──RELOAD DBSPACE──(dbspace_name)──┬─NEW───┬────────────────────────────►
                                    └─PURGE─┘

►─INFILE──(──ddname─┬─────────────────────────────┬──────────────────────►
                    │              ┌─2048─┐       │
                    └─BLKSZ──(─────┴─size─┴───)───┘
►──┬────────────────────────────────────┬──)──┬──────────────────────────────┬──►
   │                  ┌─REWIND─────┐    │     └─COMMITCOUNT──(──ccount──)────┘
   │          ┌─(TAPE)─┴─NOREWIND───┴─┐  │
   └─PDEV─────┴─(DASD)────────────────┴──┘
```

```
►──┬───────────────────────────────────┬──┬──────────────────────────────────┬──►◄
   └─RESTARTTABLE──(──table_name──)────┘  └─RESTARTCOUNT──(──rcount──)───────┘
```

RELOAD PACKAGE

Identifies a RELOAD PACKAGE request and a package to be loaded. The UNLOAD PACKAGE output file becomes the input file for the RELOAD PACKAGE command.

VM Users

```
►►──RELOAD PACKAGE──(package_name)──┬─NEW─────────────────────┬──────────────►
                                    │           ┌─KEEP───┐    │
                                    └─REPLACE───┴─REVOKE─┴────┘
```

```
►──┬──────────────────────────────────────────┬──INFILE──(ddname)──────────►◄
   │          ┌─,───────────────┐             │
   └─TO──(────▼──server_name────┴────)────────┘
```

VSE Users

```
►►──RELOAD PACKAGE──(package_name)──┬─NEW─────────────────────┬──────────────►
                                    │           ┌─KEEP───┐    │
                                    └─REPLACE───┴─REVOKE─┴────┘
```

```
►──┬──────────────────────────────────────────┬──────────────────────────────►
   │          ┌─,───────────────┐             │
   └─TO──(────▼──server_name────┴────)────────┘
```

```
►─INFILE──(──ddname─┬─────────────────────────────┬──────────────────────────►
                    │             ┌─2000─┐        │
                    └─BLKSZ──(────┴─size─┴────)───┘

►─┬──────────────────────────────────────────┬─)──────────────────────────────►◄
  │                       ┌─REWIND───┐       │
  │            ┌─(TAPE)───┴─NOREWIND─┴──┐    │
  └─PDEV───────┴─(DASD)─────────────────┴────┘
```

RELOAD TABLE

Identifies a RELOAD TABLE request and a table to be loaded.

VM Users

```
►►──RELOAD TABLE──(table_name)───┬─PURGE─────────────────┬──────────────────►
                                 └─NEW──(dbspace_name)───┘

►─┬──────────────────────────┬─INFILE──(ddname)──────────────────────────────►
  └─INTABLE──(table_name)────┘

►─┬─────────────────────────────┬──┬─────────────────────────────┬──────────►◄
  └─COMMITCOUNT──(──ccount──)───┘  └─RESTARTCOUNT──(──rcount──)──┘
```

VSE Users

```
►►──RELOAD TABLE──(table_name)───┬─PURGE─────────────────┬──────────────────►
                                 └─NEW──(dbspace_name)───┘

►─┬──────────────────────────┬───────────────────────────────────────────────►
  └─INTABLE──(table_name)────┘

►─INFILE──(──ddname─┬─────────────────────────────────┬──────────────────────►
                    │             ┌─2048──────┐       │
                    └─BLKSZ──(────┴──┬──────┬─┴──)────┘
                                     └─size─┘
►──┬──────────────────────────────────────┬──)──┬─────────────────────────────┬──►
   │                 ┌─REWIND───┐         │     └─COMMITCOUNT──(──ccount──)──┘
   │       ┌─(TAPE)──┴─NOREWIND─┴──┐      │
   └─PDEV──┴─(DASD)────────────────┴──────┘

►──┬─────────────────────────────┬────────────────────────────────────────────►◄
   └─RESTARTCOUNT──(──rcount──)──┘
```

REORGANIZE INDEX

Corrects index fragmentation and skewing of index key values without first having to drop the index and then create it by using the DROP INDEX and CREATE INDEX SQL statements.

```
►►──REORGANIZE INDEX──(index_name)──┬──────────────────────┬──────────►◄
                                    └─PCTFREE =──integer───┘
```

SCHEMA

Specifies an authorization ID and a list of table, view, and privilege definitions using the syntax of the SQL CREATE TABLE, CREATE VIEW, and GRANT statements. The SCHEMA command reads the text of a schema file and processes the statements in it.

VM Users

```
►►──SCHEMA INFILE──(ddname)──┬─────────────────────┬──────────────────►◄
                             └─IN──(dbspace_name)──┘
```

VSE Users

```
►►──SCHEMA INFILE──(──ddname──┤ option-c ├──)──┬─────────────────────┬──────►◄
                                               └─IN──(dbspace_name)──┘
```

option-c:

```
├──┬──────────────────────────┬──┬────────────────────────────────────────┬──┤
   │             ┌─2000─┐     │  │                     ┌─REWIND───┐       │
   └─BLKSZ──(────┴─size─┴──)──┘  │           ┌─(TAPE)──┴─NOREWIND─┴──┐    │
                                 └─PDEV──────┴─(DASD)────────────────┴────┘
```

SET AUTOCOMMIT

Activates or suppresses the automatic execution of the SQL COMMIT WORK statements. The SET AUTOCOMMIT command cannot span input records.

```
                   ┌─(─┐          ┌─)─┐
►►──SET AUTOCOMMIT─┴───┴──┬─ON──┬─┴───┴───────────────────────────►◄
                          └─OFF─┘
```

SET ERRORMODE

- Suspends the normal DBS Utility actions taken after a command processing error is detected and causes the DBS Utility to continue processing commands after an error has occurred
- Forces the DBS Utility to enter error mode processing
- Resumes normal DBS Utility command processing.

The SET ERRORMODE command cannot span input records. If you do not supply a SET ERRORMODE command in the input records, the DBS Utility operates as if you issued SET ERRORMODE OFF.

```
                  ┌─(─┐              ┌─)─┐
►►──SET ERRORmode─┴───┴──┬─ON───────┬─┴───┴───────────────────────►◄
                         ├─OFF──────┤
                         └─CONTINUE─┘
```

SET FORMAT

Identifies whether the Database Services Utility should use column-or-block format, column-or-list format, or only list format for SQL select-statement results. If not specified, Database Services Utility processing uses column-or-block format for SQL select-statement output.

```
               ┌─(─┐        ┌─)─┐
►►──SET FORMAT─┴───┴──┬─CB─┬─┴───┴────────────────────────────────►◄
                      ├─CL─┤
                      └─LO─┘
```

SET ISOLATION

Controls the isolation level used for Database Services Utility processing. Each time the Database Services Utility runs, the isolation level is initialized to repeatable read (RR). SQL processing through the Database Services Utility is performed at the RR isolation level until a SET ISOLATION command is encountered.

The other isolation level settings are cursor stability (CS) and uncommitted read (UR).

```
                     ┌─(─┐          ┌─)─┐
►►──SET ISOLation────┴───┴─────┬─CS─┬─┴───┴──────────────────────────────────►◄
                               ├─RR─┤
                               └─UR─┘
```

SET LINECOUNT (LINEWIDTH)

Defines the number of lines per page for Database Services Utility message file output and the number of print data positions used in each Database Services Utility message file record containing SQL SELECT statement output.

The SET LINECOUNT command cannot span input records. You must specify either the LINEWIDTH(*www*) parameter or the LINECOUNT(*ccc*) parameter, or both, to prevent a Database Services Utility processing error from occurring.

VM Users

```
►►──SET──┬─LineCount──(ccc)──┬──────────────────────────────┬─┬──────────────►◄
         │                   │                ┌─80──┐        │ │
         │                   │                ├─120─┤        │ │
         │                   └─LineWidth──(───┴─www─┴───)────┘ │
         └─LineWidth──(www)──┬─────────────────────────────┬───┘
                             │                ┌─60──┐       │
                             └─LineCount──(───┴─ccc─┴──)────┘
```

VSE Users

```
►►──SET──┬─LineCount──(ccc)──┬──────────────────────────────┬─┬──────────────►◄
         │                   │                ┌─120─┐        │ │
         │                   └─LineWidth──(───┴─www─┴───)────┘ │
         └─LineWidth──(www)──┬─────────────────────────────┬───┘
                             │                ┌─60──┐       │
                             └─LineCount──(───┴─ccc─┴──)────┘
```

SET UPDATE STATISTICS

Controls the automatic UPDATE STATISTICS processing performed during Database Services Utility RELOAD TABLE, RELOAD DBSPACE, and DATALOAD TABLE command processing. The command cannot span input records. If you do not supply a SET UPDATE STATISTICS command in the input records, the Database Services Utility operates as if you issued SET UPDATE STATISTICS ON.

```
         ┌─UPDATE─┐              ┌─(─┐            ┌─)─┐
►►──SET──┴────────┴──STATISTICS──┴───┴──┬─ON──┬───┴───┴──────────────────►◄
                                        └─OFF─┘
```

UNLOAD DBSPACE

Unloads all tables of the specified DBSPACE to a sequential output file.

VM Users

```
►►──UNLOAD DBSPACE──(dbspace_name)──OUTFILE──(ddname)──────────────────►◄
```

VSE Users

```
►►──UNLOAD DBSPACE──(dbspace_name)─────────────────────────────────────►

►─OUTFILE──(──ddname──┬──────────────────────────────┬─────────────────►
                      │               ┌─2048─┐       │
                      └─BLKSZ──(──────┴─size─┴────)──┘

►─┬────────────────────────────────────┬─)──────────────────────────────►◄
  │           ┌─NOREWIND─┐             │
  │     ┌─(TAPE)──┴─REWIND───┴─┐       │
  └─PDEV─┴─(DASD)──────────────┴───────┘
```

UNLOAD PACKAGE

Unloads a specific package to a file.

VM Users

```
►►──UNLOAD PACKAGE──(package_name)──┬──────────────────────┬──OUTFILE──────►
                                    └─FROM──(server_name)──┘

►─(ddname)─────────────────────────────────────────────────────────────►◄
```

VSE Users

```
►►──UNLOAD PACKAGE──(package_name)──┬──────────────────────┬───────────────►
                                    └─FROM──(server_name)──┘
```

```
►─OUTFILE──(──ddname──┬──────────────────────────────────┬──────────────────────►
                      │              ┌─2000─┐            │
                      └─BLKSZ──(─────┴─size─┴──────)─────┘

►─┬──────────────────────────────────────────┬─)──────────────────────────────►◄
  │                    ┌─NOREWIND─┐          │
  │         ┌─(TAPE)───┴─REWIND───┴──┐       │
  └─PDEV────┴─(DASD)─────────────────┴───────┘
```

UNLOAD TABLE

Unloads a specific table or view to an output file.

VM Users

```
►►──UNLOAD TABLE──(table_name)──OUTFILE──(ddname)─────────────────────────────►◄
```

VSE Users

```
►►──UNLOAD TABLE──(table_name)────────────────────────────────────────────────►

►─OUTFILE──(──ddname──┬──────────────────────────────────┬──────────────────────►
                      │              ┌─2048─┐            │
                      └─BLKSZ──(─────┴─size─┴──────)─────┘

►─┬──────────────────────────────────────────┬─)──────────────────────────────►◄
  │                    ┌─NOREWIND─┐          │
  │         ┌─(TAPE)───┴─REWIND───┴──┐       │
  └─PDEV────┴─(DASD)─────────────────┴───────┘
```

Chapter 9. SQLCA and SQLDA

SQL Communication Area (SQLCA)

The SQLCA is a collection of variables that are updated at the end of the execution of every SQL statement.

Note: The field names are those provided by the SQL INCLUDE statement for Assembler, COBOL and PL/I. The names for C and FORTRAN are similar.

Field Name and Data Type	Description
SQLCAID CHAR(8)	The constant SQLCA.
SQLCABC INTEGER	Length of SQLCA. Always 136.
SQLCODE INTEGER	Negative: error condition, Zero: successful execution, Positive: warning condition.
SQLERRML SMALLINT	Length of SQLERRMC.
SQLERRMC VARCHAR(70)	Zero, one, or more tokens separated by X'FF'.
SQLERRP CHAR(8)	Characters 1 to 3 identify the product. All 8 characters identify the module if the SQLCODE < 0.
SQLERRD(1) INTEGER	RDS return code.
SQLERRD(2) INTEGER	DBSS return code.
SQLERRD(3) INTEGER	Number of rows affected by INSERT, UPDATE or DELETE.
SQLERRD(4) INTEGER	Rough estimate of resources.
SQLERRD(5) INTEGER	For DELETE with RI: number of dependent rows. For local time exit: function number.
SQLERRD(6) INTEGER	Reserved.
SQLWARN array:	
SQLWARN0 CHAR(1)	'' if all indicators blank, 'S' if SQLWARN6 = 'S', 'W' otherwise.
SQLWARN1 CHAR(1)	'W' if value truncated when assigned to the host variable. 'Z' if invalid mixed data value is truncated.
SQLWARN2 CHAR(1)	'W' if null values eliminated from function result.
SQLWARN3 CHAR(1)	'W' if not enough host variables for column values.

Field Name and Data Type	Description
SQLWARN4 CHAR(1)	'W' if WHERE missing from prepared UPDATE or DELETE.
SQLWARN5 CHAR(1)	'W' indicates performance degradation.
SQLWARN6 CHAR(1)	'S' if database manager is in unusable state due to error. 'W' if database manager was forced to terminate LUW.
SQLWARN7 CHAR(1)	'W' if adjustment made for last day of month. 'Z' if loss of digits on decimal division.
SQLWARN8 CHAR(1)	'W' if statement disqualified for blocking. 'Z' if substitute character used on conversion.
SQLWARN9 CHAR(1)	'W' if not enough storage for blocking.
SQLWARNA CHAR(1)	'V' if error in converting SQLCA at application requester. 'W' if blocking factor could not be maintained.
SQLSTATE CHAR(5)	Standard cross-product return code.

SQL Descriptor Area (SQLDA)

An SQLDA is a collection of variables that is required for execution of the DESCRIBE statement, and can optionally be used by the OPEN, FETCH, EXECUTE, PUT, and Extended PREPARE statements.

Note: The field names in the following table are those provided by the SQL INCLUDE statement for Assembler and PL/I. The names for C are the same except that they are in lower case.

	Use in DESCRIBE	Use in Other Statements
Field Name Data Type		
SQLDAID CHAR(8)	The constant SQLDA.	Not used.
SQLDABC INTEGER	Length of SQLDA, equal to SQLN*44+16.	Same.
SQLN SMALLINT	Number of occurences of SQLVAR.	Same.
SQLD SMALLINT	Number of columns being described (times 2 if BOTH specified). Zero for a non-SELECT statement.	Number of host variables described by occurrences of SQLVAR.

Field Name Data Type	Use in DESCRIBE	Use in Other Statements
SQLVAR array:		
SQLTYPE SMALLINT	384/385 date 388/389 time 392/393 timestamp 448/449 short varchar 452/453 character 456/457 long varchar 460/461 464/465 short vargraphic 468/469 graphic 472/473 long vargraphic 480/481 float 484/485 packed decimal 496/497 large integer 500/501 small integer 504/505	384/385 char containing date 388/389 char containing time 392/393 char containing timestamp 448/449 short varchar 452/453 character 456/457 long varchar 460/461 NUL-terminated string 464/465 short vargraphic 468/469 graphic 472/473 long vargraphic 480/481 float 484/485 packed decimal 496/497 large integer 500/501 small integer 504/505 DISPLAY SIGN LEADING SEPARATE
SQLLEN SMALLINT	External length of column.	External length of host variable.
SQLDATA CHAR(4) or pointer	SBCS: X'0000' ccsid, mixed: X'0000' ccsid, bit: X'0000FFFF', graphic: X'0000' ccsid, otherwise: unused.	Address of host variable.
SQLIND CHAR(4) or pointer	1st byte set for character data: SBCS: X'01' MIXED: X'02' BIT: X'FF', otherwise: unused. [1]	Address of indicator variable, if there is one.
SQLNAME VARCHAR(30)	Name or label of the column.	SBCS: X'0000' ccsid, mixed: X'0000' ccsid, bit: X'0000FFFF', graphic: X'0000' ccsid, otherwise: unused.

Chapter 10. Catalog Tables

The DB2 Server for VM and DB2 Server for VSE database management systems maintain a set of tables, called catalog tables, that store information about the database. The catalog tables are automatically updated by the database manager during normal operation and in response to SQL data definition and control statements. Following is a Roadmap.

Roadmap

Item	Catalog Table	Page
authorization	SYSUSERAUTH	114
	SYSUSERLIST	114
character conversion	SYSSTRINGS	114
character set	SYSCHARSETS	111
coded character set identifiers	SYSCCSIDS	111
	SYSSTRINGS	114
column	SYSCOLUMNS	111
	SYSKEYCOLS	112
column update privilege	SYSCOLAUTH	111
column with field procedure	SYSFIELDS	112
constraint	SYSKEYS	112
dbspace	SYSDBSPACES	112
	SYSUSAGE	114
	SYSDROP	112
dbspace waiting to be dropped	SYSDROP	112
default	SYSOPTIONS	113
dropped dbspace	SYSDROP	112
dropped table	SYSDROP	112
field procedures	SYSFPARMS	112
	SYSFIELDS	112
foreign key	SYSKEYS	112
index	SYSINDEXES	112
	SYSUSAGE	114
index column statistics	SYSCOLSTATS	111
	SYSCOLUMNS	111
	SYSINDEXES	112
key	SYSKEYS	112
key column	SYSKEYCOLS	112
language for character set	SYSLANGUAGE	113
option	SYSOPTIONS	113

[1] Not available for the DRDA protocol.

Item	Catalog Table	Page
package	SYSACCESS SYSUSAGE	110 114
package run privilege	SYSPROGAUTH	113
password	SYSUSERAUTH	114
privilege	SYSCOLAUTH SYSPROGAUTH SYSTABAUTH	111 113 114
primary key	SYSKEYS	112
statistics	SYSCATALOG SYSCOLSTATS SYSCOLUMNS SYSDBSPACES SYSINDEXES	110 111 111 112 112
stored procedures	SYSPARMS SYSPSERVERS SYSROUTINES	113 114 113
synonym	SYSSYNONYMS	114
table	SYSCATALOG SYSCOLUMNS SYSUSAGE	110 111 114
table privilege	SYSTABAUTH	114
table waiting to be dropped	SYSDROP	112
unique constraint	SYSKEYS	112
view	SYSVIEWS SYSCATALOG SYSCOLUMNS SYSACCESS SYSUSAGE	115 110 111 110 114
view privilege	SYSTABAUTH	114

Catalog Table Descriptions

Following is a description of the catalog tables:

SYSACCESS Records information about the tables in which packages are stored.

CONSTKN
CREATOR
DBSPACENO
FIRSTROW
LINKID
PLABEL
TABID
TABTYPE
TIMESTAMP
TNAME
VALID

SYSCATALOG Contains a row for each table or view in the database, including one for itself and one for each catalog table.

AVGROWLEN
CLUSTERROW
CLUSTERTYPE
CREATOR
DATACAPTURE
DBSPACENAME
DBSPACENO
DEPENDENTS
INACTIVE
LFDDBSPACE
LFDLINK
LFDTABID
NCOLS
NOVERFLOW
NPAGES
PARENTS
PCTPAGES
REMARKS
ROWCOUNT
TABID
TABLETYPE
TLABEL
TNAME

SYSCCSIDS Contains a row for every CCSID supported by the database manager.

CCSID
CHARNAME
DBCSID
SBCSID
SUBTYPE

SYSCHARSETS Rows contain information about various EBCDIC character sets; information is based on what is specified in the CHARNAME initialization parameter.

CHARCLASS
CHARTRANS
NAME

SYSCOLAUTH Records grants of the UPDATE privilege on tables and views when the privilege is granted on a column-by-column basis.

COLNAME
CREATOR
GRANTEE
GRANTOR
TIMESTAMP
TNAME

SYSCOLSTATS Keeps column statistics for a column which is the first column of an index.

CNAME
CREATOR
FREQ1PCT
FREQ2PCT
FREQ1VAL
FREQ2VAL
TNAME
VAL10
VAL50
VAL90

SYSCOLUMNS Contains a more detailed description of the database than SYSCATALOG.

AVGCOLLEN
CCSID
CLABEL
CNAME
COLCOUNT
COLINFO
COLNO
COLTYPE
CREATOR
FLDPROC
HIGH2KEY
LENGTH
LOW2KEY
NULLS
ORDERFIELD
REMARKS
SUBTYPE
SYSLENGTH
TNAME

SYSDBSPACES Contains a row for each PUBLIC and PRIVATE DBSPACE in the database, including those DBSPACEs that no user has yet acquired.

DBSPACENAME
DBSPACENO
DBSPACETYPE
FREEPCT
LOCKMODE
NACTIVE
NPAGES
NRHEADER
NTABS
OWNER
PCTINDX
POOL

SYSDROP Contains a list of objects waiting to be dropped.

DBSPACENO
QUALF
TABID

SYSFIELDS Contains a row for each column that has a field procedure associated with it.

CNAME
COLNO
CREATOR
FLDLENGTH
FLDTYPE
FPEXITPARML
FPNAME
FPPARMLIST
FPWORKAREA
TNAME

SYSFPARMS Holds the field procedure value block contents for each field procedure.

CNAME
CREATOR
FPEXITPARM
FPNAME
SEQNO
TNAME

SYSINDEXES Contains a row for every index currently in existence, including the indexes that the database manager maintains on its own catalog tables.

CLUSTER
CLUSTERRATIO
COLNAMES
COLNUMBERS
CREATOR
FIRSTKEYCOUNT
FULLKEYCOUNT
ICREATOR
IID
INAME
INDEXTYPE
IPCTFREE
KEYLEN
KEYTYPE
LOCKMODE
NLEAF
NLEVELS
RELEASE
TNAME

SYSKEYCOLS Contains a row for every column in every key.

CCSID
CNAME
DATACODE
FLDPROC
KEYNAME
KEYORD
KEYTYPE
SYSLENGTH
TABLEORD
TCREATOR
TIMESTAMP
TNAME

SYSKEYS Contains a row for each primary and each foreign key.

DELETERULE
KEYNAME
REFTNAME
TIMESTAMP

INAME
KEYCOLS
KEYTYPE
REFTCREATOR
STATUS
TCREATOR
TNAME

SYSLANGUAGE Contains the names of all national languages currently installed, a unique four-character code for each language, and a brief description of each language.

LANGID
LANGKEY
LANGUAGE
REMARKS

SYSOPTIONS Contains the options and defaults that may be implemented for this database.

REMARKS
SQLOPTION
VALUE

The following named rows describe the options and defaults that may be implemented for this database:

CCSIDGRAPHIC
CCSIDMIXED
CCSIDSBCS
CHARNAME
CHARSUB
DATE
DBCS
DEFAULT LANGUAGE
LDATELEN
LTIMELEN
MCCSIDGRAPHIC
MCCSIDMIXED
MCCSIDSBCS
RELEASE
TIME

SYSPARMS Describes the parameters for the stored procedures defined.

NAME
ROUTINEID
TYPENAME
AUTHID
ROWTYPE
DATATYPEID
PARMNAME
ORDINAL
LENGTH
SUBTYPE
CCSID
SCALE

SYSPSERVERS Defines the stored procedure servers where stored procedures run and puts them in groups.

PSERVER
SERVGROUP
AUTOSTART
DESCRIPTION

SYSPROGAUTH Records privileges of users to run programs, and to grant these privileges to other users.

CREATOR
GRANTEE
GRANTOR
PROGNAME
RUNAUTH
TIMESTAMP

SYSROUTINES Specifies the load module or phase name and package name for a given stored procedure.

NAME
AUTHID
LOADMOD
ROUTINEID
PARMCOUNT
LANGUAGE
PARAMETERSTYLE
STAYRESIDENT
PROGRAMTYPE
COMMITONRETURN
RESULTSETS
SERVGROUP
DEFSERV
RUNOPTS
REMARKS

SYSSTRINGS Contains a list of the valid combinations for source and target CCSID tags when using the remote unit of work feature.

ERRORBYTE
INCCSID
OUTCCSID
SUBBYTE
TRANSPROC
TRANSTAB1
TRANSTAB2
TRANSTYPE

SYSSYNONYMS Contains a row for every synonym currently in effect.

ALTNAME
CREATOR
TNAME
USERID

SYSTABAUTH Records privileges owned by users to access tables and views, and privileges on tables and views exercised by programs.

ALTERAUTH
DELETEAUTH
GRANTEE
GRANTEETYPE
GRANTOR
INDEXAUTH
INSERTAUTH
REFAUTH
SCREATOR
SELECTAUTH
STNAME
TCREATOR
TIMESTAMP
TTNAME
UPDATEAUTH
UPDATECOLS

SYSUSAGE Records dependencies of one database object on another.

BCREATOR
BNAME
BTYPE
DCREATOR
DNAME
DTYPE
TIMESTAMP

SYSUSERAUTH Records special privileges of DBA, RESOURCE, SCHEDULE or CONNECT authority held by a user or a special privilege exercised by a program.

AUTHOR
DBAAUTH
NAME
PASSWORD
RESOURCEAUTH
SCHEDULEAUTH

SYSUSERLIST Records special privileges for access by users who do not have DBA authority.

AUTHOR
NAME
SCHEDULEAUTH

DBAAUTH
RESOURCEAUTH

SYSVIEWS Contains the definitions of all views.

SEQNO
VCREATOR
VIEWCHECK
VIEWMAT
VIEWNAME
VIEWTEXT

Chapter 11. Application Server Support for VSE

Up to 36 DB2 Server for VSE application servers can be active at the same time in your VSE system.

DBNAME Directory

The DBNAME directory is a user-definable directory of application server names, contained in an A-type source member called ARISDIRD. Each entry in this directory is an 80 byte record in the following format:

- Comment, column 1
- Transaction Program Name (TPN), columns 2 to 5
- Application Identifier (APPLID), columns 10 to 17
- System default marker (SYSDEFAULT), column 21
- DBNAME columns 22 to 39
- Partition name (PDEFAULT) columns 44 and 45
- Privileged (PRIVILEGE) column 50.

For more information about the DBNAME directory, refer to the *DB2 Server for VSE System Administration* manual.

Chapter 12. SQL Reserved Words

Following is a list of SQL reserved words you should avoid using:

ACQUIRE
ADD
ALL
ALTER
AND
ANY
AS
ASC
AVG

BETWEEN
BY

CALL
CHAR
CHARACTER
COLUMN
COMMENT
COMMIT
CONCAT
CONNECT
COUNT
CREATE
CURRENT

DBA
DBSPACE
DELETE
DESC
DISTINCT
DOUBLE
DROP

EXCLUSIVE
EXECUTE
EXISTS
EXPLAIN

FIELDPROC
FOR
FROM

GRANT
GRAPHIC
GROUP

HAVING

IDENTIFIED
IN
INDEX
INSERT
INTO
IS

LIKE
LOCK
LONG

MAX
MIN
MODE

NAMED
NHEADER
NOT
NULL

OF
ON
OPTION
OR
ORDER

PACKAGE
PAGE
PAGES
PCTFREE
PCTINDEX
PRIVATE
PRIVILEGES
PROGRAM
PUBLIC

RESOURCE
REVOKE
ROLLBACK
ROW
RUN

SCHEDULE
SELECT
SET
SHARE
SOME
STATISTICS
STORPOOL
SUM
SYNONYM

TABLE
TO

UNION
UNIQUE
UPDATE
USER

VALUES
VIEW

WHERE
WITH
WORK

Chapter 13. DBS Utility Reserved Words

In addition to the SQL reserved words, do not use the following words in Database Services Utility commands as the name for a table, view, column, or DBSPACE, unless they are enclosed in double quotation marks ("):

DATALOAD
DATAUNLOAD
INFILE
INMOD
OUTFILE
REBIND
RELOAD
REORGANIZE
SCHEMA
UNLOAD

Chapter 14. Notes

Notices

IBM may not offer the products, services, or features discussed in this document in all countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10594-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Mail Station P300
522 South Road
Poughkeepsie, NY 12601-5400
U.S.A

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurement may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements, or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information may contain examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information may contain sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing, or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Programming Interface Information

This book documents intended Programming Interfaces that allow the customer to write programs to obtain services of DB2 Server for VSE & VM.

Trademarks

The following terms are trademarks of International Business Machines Corporation in the United States, or other countries, or both:

CICS
CICS/VSE
DATABASE 2
DB2
DRDA
IBM
IBMLink
PROFS
SAA

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Other company, product, and service names may be trademarks or service marks of others.

Readers' Comments — We'd Like to Hear from You

DB2 Server for VSE & VM
Quick Reference
Version 7 Release 1

Publication No. SC09-2988-00

Overall, how satisfied are you with the information in this book?

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Overall satisfaction	□	□	□	□	□

How satisfied are you that the information in this book is:

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Accurate	□	□	□	□	□
Complete	□	□	□	□	□
Easy to find	□	□	□	□	□
Easy to understand	□	□	□	□	□
Well organized	□	□	□	□	□
Applicable to your tasks	□	□	□	□	□

Please tell us how we can improve this book:

Thank you for your responses. May we contact you? □ Yes □ No

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

Name

Address

Company or Organization

Phone No.

Readers’ Comments — We’d Like to Hear from You

SC09-2988-00

Cut or Fold
Along Line

Fold and Tape **Please do not staple** Fold and Tape

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

IBM CANADA LTD.
DB2 Server for VSE & VM
2S/240/1150/TOR
1150 Eglinton Avenue East
North York, Ontario, Canada M3C 1H7

Fold and Tape **Please do not staple** Fold and Tape

Cut or Fold
Along Line

File Number: S370/4300-50
Program Number: 5697-F42

Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.

SC09-2988-00