

IBM DB2 Universal Database



# Command Reference

*Version 5*



IBM DB2 Universal Database



# Command Reference

*Version 5*

Before using this information and the product it supports, be sure to read the general information under Appendix E, "Notices" on page 451.

This document contains proprietary information of IBM. It is provided under a license agreement and is protected by copyright law. The information contained in this publication does not include any product warranties and any statements provided in this manual should not be interpreted as such.

Order publications through your IBM representative or the IBM branch office serving your locality or by calling 1-800-879-2755 in U.S. or 1-800-IBM-4YOU in Canada.

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

© **Copyright International Business Machines Corporation 1993, 1997. All rights reserved.**

Note to U.S. Government Users — Documentation related to restricted rights — Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corp.

---

# Contents

<b>About This Book</b> . . . . .	vii
Who Should Use this Book . . . . .	vii
How this Book is Structured . . . . .	vii
<b>Chapter 1. System Commands</b> . . . . .	1
How the Command Descriptions are Organized . . . . .	1
db2admin - DB2 Administration Server . . . . .	3
db2autold - Autoloader . . . . .	5
db2batch - Benchmark Tool . . . . .	7
db2bfd - Bind File Description Tool . . . . .	12
db2cc - Start Control Center . . . . .	13
db2cidmg - Remote Database Migration . . . . .	14
db2ckmig - Database Pre-migration Tool . . . . .	15
db2cli - DB2 Interactive CLI . . . . .	17
db2cmd - Open DB2 Command Window . . . . .	18
db2drdat - DRDA Trace . . . . .	19
db2empfa - Enable Multi-page File Allocation . . . . .	21
db2eva - Event Analyzer . . . . .	22
db2evmon - Event Monitor Productivity Tool . . . . .	24
db2exfmt - Explain Table Format Tool . . . . .	25
db2expln - DB2 SQL Explain Tool . . . . .	27
db2gov - DB2 Governor . . . . .	29
db2govlg - DB2 Governor Log Query . . . . .	37
db2icrt - Create Instance . . . . .	38
db2idrop - Remove Instance . . . . .	39
db2ilist - List Instances . . . . .	40
db2imigr - Migrate Instance . . . . .	41
db2ipxad - Get IPX/SPX Internetwork Address . . . . .	42
db2iupdt - Update Instances . . . . .	43
db2look - DB2 Statistics Extraction Tool . . . . .	44
db2migdr - Local Database Directory Migration . . . . .	47
db2rbind - Rebind all Packages . . . . .	48
db2sampl - Create Sample Database . . . . .	49
db2set - DB2 Profile Registry Command . . . . .	50
db2split - Data Declustering Tool . . . . .	53
db2sql92 - SQL92 Compliant SQL Statement Processor . . . . .	54
db2start - Start DB2 . . . . .	57
db2stop - Stop DB2 . . . . .	58
db2tbst - Get Tablespace State . . . . .	59
db2trc - Trace . . . . .	60
db2uiddl - Prepare Unique Index Conversion to V5 Semantics . . . . .	63
db2untag - Release Container Tag . . . . .	64
db2vexp - Dynamic Visual Explain . . . . .	66
<b>Chapter 2. Command Line Processor (CLP)</b> . . . . .	69

Command Line Processor Invocation and Options . . . . .	69
Using the Command Line Processor . . . . .	78
<b>Chapter 3. CLP Commands . . . . .</b>	<b>83</b>
DB2 CLP Commands . . . . .	83
ACTIVATE DATABASE . . . . .	87
ADD NODE . . . . .	89
ATTACH . . . . .	91
BACKUP DATABASE . . . . .	93
BIND . . . . .	98
CATALOG APPC NODE . . . . .	111
CATALOG APPCLU NODE . . . . .	114
CATALOG APPN NODE . . . . .	116
CATALOG DATABASE . . . . .	118
CATALOG DCS DATABASE . . . . .	121
CATALOG GLOBAL DATABASE . . . . .	124
CATALOG IPX/SPX NODE . . . . .	126
CATALOG LOCAL NODE . . . . .	129
CATALOG NAMED PIPE NODE . . . . .	131
CATALOG NETBIOS NODE . . . . .	133
CATALOG ODBC DATA SOURCE . . . . .	135
CATALOG TCP/IP NODE . . . . .	136
CHANGE DATABASE COMMENT . . . . .	139
CHANGE ISOLATION LEVEL . . . . .	141
CREATE DATABASE . . . . .	143
DEACTIVATE DATABASE . . . . .	149
DEREGISTER . . . . .	151
DESCRIBE . . . . .	152
DETACH . . . . .	156
DROP DATABASE . . . . .	157
DROP NODE VERIFY . . . . .	159
ECHO . . . . .	160
EXPORT . . . . .	161
FORCE APPLICATION . . . . .	167
GET ADMIN CONFIGURATION . . . . .	169
GET AUTHORIZATIONS . . . . .	172
GET CONNECTION STATE . . . . .	174
GET DATABASE CONFIGURATION . . . . .	175
GET DATABASE MANAGER CONFIGURATION . . . . .	184
GET DATABASE MANAGER MONITOR SWITCHES . . . . .	194
GET INSTANCE . . . . .	196
GET MONITOR SWITCHES . . . . .	197
GET SNAPSHOT . . . . .	199
HELP . . . . .	211
IMPORT . . . . .	212
INITIALIZE TAPE . . . . .	221
INVOKE STORED PROCEDURE . . . . .	222
LIST ACTIVE DATABASES . . . . .	224

LIST APPLICATIONS . . . . .	225
LIST BACKUP/HISTORY . . . . .	227
LIST COMMAND OPTIONS . . . . .	229
LIST DATABASE DIRECTORY . . . . .	231
LIST DCS APPLICATIONS . . . . .	235
LIST DCS DIRECTORY . . . . .	237
LIST DRDA INDOUBT TRANSACTIONS . . . . .	239
LIST INDOUBT TRANSACTIONS . . . . .	241
LIST NODE DIRECTORY . . . . .	245
LIST NODEGROUPS . . . . .	248
LIST NODES . . . . .	250
LIST ODBC DATA SOURCES . . . . .	251
LIST PACKAGES/TABLES . . . . .	253
LIST TABLESPACE CONTAINERS . . . . .	256
LIST TABLESPACES . . . . .	258
LOAD . . . . .	262
LOAD QUERY . . . . .	280
MIGRATE DATABASE . . . . .	282
PRECOMPILE PROGRAM . . . . .	284
PRUNE HISTORY . . . . .	303
QUERY CLIENT . . . . .	304
QUIESCE TABLESPACES FOR TABLE . . . . .	305
QUIT . . . . .	308
REBIND . . . . .	309
REDISTRIBUTE NODEGROUP . . . . .	312
REGISTER . . . . .	315
REORGANIZE TABLE . . . . .	317
REORGCHK . . . . .	320
RESET ADMIN CONFIGURATION . . . . .	327
RESET DATABASE CONFIGURATION . . . . .	329
RESET DATABASE MANAGER CONFIGURATION . . . . .	331
RESET MONITOR . . . . .	333
RESTART DATABASE . . . . .	335
RESTORE DATABASE . . . . .	337
REWIND TAPE . . . . .	343
ROLLFORWARD DATABASE . . . . .	344
RUNSTATS . . . . .	350
SET CLIENT . . . . .	353
SET RUNTIME DEGREE . . . . .	356
SET TABLESPACE CONTAINERS . . . . .	358
SET TAPE POSITION . . . . .	360
START DATABASE MANAGER . . . . .	361
STOP DATABASE MANAGER . . . . .	365
TERMINATE . . . . .	368
UNCATALOG DATABASE . . . . .	369
UNCATALOG DCS DATABASE . . . . .	371
UNCATALOG NODE . . . . .	372
UNCATALOG ODBC DATA SOURCE . . . . .	374

UPDATE ADMIN CONFIGURATION . . . . .	375
UPDATE COMMAND OPTIONS . . . . .	377
UPDATE DATABASE CONFIGURATION . . . . .	379
UPDATE DATABASE MANAGER CONFIGURATION . . . . .	381
UPDATE MONITOR SWITCHES . . . . .	383
UPDATE RECOVERY HISTORY FILE . . . . .	385
<b>Chapter 4. Using Command Line SQL Statements . . . . .</b>	<b>387</b>
<b>Appendix A. How to Read the Syntax Diagrams . . . . .</b>	<b>393</b>
<b>Appendix B. Naming Conventions . . . . .</b>	<b>397</b>
<b>Appendix C. IMPORT/EXPORT/LOAD Utility File Formats . . . . .</b>	<b>399</b>
Delimited ASCII (DEL) File Format . . . . .	399
Sample DEL File . . . . .	400
DEL Data Type Descriptions . . . . .	402
PC Version of IXF File Format . . . . .	404
PC/IXF Record Types . . . . .	406
PC/IXF Data Types . . . . .	414
PC/IXF Data Type Descriptions . . . . .	419
General Rules Governing PC/IXF File Import into Databases . . . . .	424
Data Type-Specific Rules Governing PC/IXF File Import into Databases . . . . .	426
FORCEIN Option . . . . .	429
Differences between Version 1 PC/IXF and Version 0 System/370 IXF . . . . .	435
Non-delimited ASCII (ASC) File Format . . . . .	436
Sample ASC File . . . . .	437
ASC Data Type Descriptions . . . . .	437
<b>Appendix D. How the DB2 Library Is Structured . . . . .</b>	<b>441</b>
SmartGuides . . . . .	441
Online Help . . . . .	442
DB2 Books . . . . .	444
About the Information Center . . . . .	448
<b>Appendix E. Notices . . . . .</b>	<b>451</b>
Trademarks . . . . .	451
Trademarks of Other Companies . . . . .	452
<b>Index . . . . .</b>	<b>453</b>
<b>Contacting IBM . . . . .</b>	<b>455</b>



---

## About This Book

This book provides information about the use of system commands and the IBM DB2 Universal Database command line processor (CLP) to execute database administrative functions.

---

## Who Should Use this Book

It is assumed that the reader has an understanding of database administration and a knowledge of Structured Query Language (SQL).

---

## How this Book is Structured

This book provides the reference information needed to use the CLP.

The following topics are covered:

- |            |   |
|------------|---|
| Chapter 1  | Describes the commands that can be entered at an operating system command prompt or in a shell script to access the database manager. |
| Chapter 2  | Explains how to invoke and use the command line processor, and describes the CLP options.   |
| Chapter 3  | Provides a description of all database manager commands.  |
| Chapter 4  | Provides information on how to use SQL statements from the command line.  |
| Appendix A | Explains the conventions used in syntax diagrams.   |
| Appendix B | Explains the conventions used to name objects such as databases and tables.   |
| Appendix C | Describes external file formats supported by the database manager import, export, and load utilities.                                 |



---

## Chapter 1. System Commands

This chapter provides information about the commands that can be entered at an operating system command prompt, or in a shell script, to access and maintain the database manager.

**Note:** Slashes (/) in directory paths are specific to UNIX based systems, and are equivalent to back slashes (\) in directory paths on OS/2 and Windows operating systems.

---

### How the Command Descriptions are Organized

A short description of each command precedes some or all of the following subsections.

#### Scope

The command's scope of operation within the instance. In a single-node system, the scope is that single node only. In a multi-node system, it is the collection of all logical nodes defined in the node configuration file, `db2nodes.cfg`.

#### Authorization

The authority required to successfully invoke the command.

#### Required Connection

One of the following: database, instance, none, or establishes a connection. Indicates whether the function requires a database connection, an instance attachment, or no connection to operate successfully. An explicit connection to the database or attachment to the instance may be required before a particular command can be issued. Commands that require a database connection or an instance attachment can be executed either locally or remotely. Those that require neither cannot be executed remotely; when issued at the client, they affect the client environment only. For information about database connections and instance attachments, see the *Administration Guide*.

#### Command Syntax

For information about syntax diagrams, see Appendix A, "How to Read the Syntax Diagrams" on page 393.

#### Command Parameters

A description of the parameters available to the command.

#### Usage Notes

Other information.

**See Also**

A cross-reference to related information.

---

### db2admin - DB2 Administration Server

This utility is used to manage the DB2 Administration Server. For more information about the DB2 Administration Server, see the *Administration Guide*.

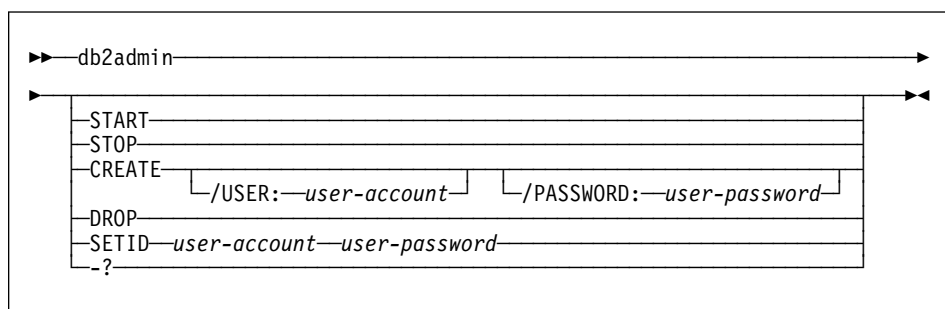
#### Authorization

Local administrator on Windows operating systems, or a system administrator on OS/2.

#### Required Connection

None

#### Command Syntax



#### Command Parameters

**Note:** If no parameters are specified, and the DB2 Administration Server exists, this command returns the DB2 Administration Server instance.

##### START

Start the DB2 Administration Server.

##### STOP

Stop the DB2 Administration Server.

##### CREATE

**/USER:** *user-account*

**/PASSWORD:** *user-password*

Create the DB2 Administration Server. If a user name and password are specified, the DB2 Administration Server instance will be associated with this user account. If the specified values are not valid, the utility returns an authentication error. The specified user account must be a valid SQL identifier, and must exist in the security database. It is recommended that a user account be specified to ensure that all DB2 Administration Server functionality can be accessed.

##### DROP

Deletes the DB2 Administration Server instance.

## db2admin - DB2 Administration Server

**SETID** *user-account/user-password*

Establishes or modifies the user account associated with the DB2 Administration Server instance.

**-?**

Display help information. When this option is specified, all other options are ignored, and only the help information is displayed.

---

### db2autold - Autoloader

Autoloader is a tool for splitting and loading data in an MPP environment. This utility can:

- Transfer data from one system (MVS, for example) to an AIX system (RS/6000 or SP2)
- Partition or split data in parallel
- Load data simultaneously on corresponding nodes.

Autoloader can run in one of four modes:

#### Split\_And\_Load

In this mode, data is FTPed from the input source (if it is remote). It is then split in parallel and loaded simultaneously on the corresponding nodes.

#### Split\_Only

In this mode, data is FTPed from the input source (if it is remote). It is then split in parallel, and the output from the splitters is written to files in the current Autoloader working directory.

#### Load\_Only

In this mode, data is assumed to be pre-split and contained in files named as follows: *filename.xxx*, where *xxx* represents the node number. The split process is skipped, and data is loaded simultaneously on the corresponding nodes. It is assumed that the files reside in the current Autoloader working directory.

#### Analyze

In this mode, an optimal partitioning map with even distribution across all nodes is generated.

**Note:** This utility does not split LOBs.

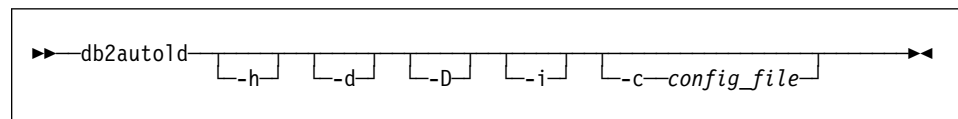
### Authorization

*sysadm*

### Required Connection

None. This command establishes a database connection.

### Command Syntax



### Command Parameters

**-h**

Display help information. When this option is specified, all other options are ignored, and only the help information is displayed.

## db2autold - Autoloader

- d** In the case of an abnormal exit from Autoloader, it is necessary to run Autoloader with this option to clean up all associated temporary directories, files, and hanging processes.
- D** Turn on debug mode.
- i** Turn on interactive mode for clean-up.
- c *config\_file*** Name of the file containing configuration data required by the tool. The default name is `auto1oad.cfg`.

### Usage Notes

Autoloader creates a log file called `auto1oad.logfile`. This file contains messages from the main autoloader script, and should be checked to ensure that all pipes and temporary directories have been set up correctly. The utility also creates a file called `1oad_1og.xxx`. This file contains messages from the load process on node `xxx`. The utility also creates a file called `split_1og.xxx`. This file contains messages from the split process on node `xxx`.

If the order of input data is to be maintained during a load operation, only one node should be used for splitting the data. Parallel splitting does not guarantee that the data will be loaded in the order that it was received.

If LOBs are stored in separate files (that is, the LOAD LOBSINFILE file type modifier is being used), all directories containing the LOB files should be made accessible to all the nodes involved in the load operation.

Autoloader ignores the LOAD *messages* parameter, and directs all messages from the LOAD command into `1oad_1og.xxx`.

If the STATISTICS YES option is specified with the LOAD command, Autoloader chooses only one output node on which to collect statistics.

For more information about **db2autold**, see the *Administration Guide*.

### See Also

“db2split - Data Declustering Tool” on page 53  
“LOAD” on page 262.



## db2batch - Benchmark Tool

Reads SQL statements from either a flat file or standard input, dynamically prepares and describes the statements, and returns an answer set. This tool is provided in the misc subdirectory of the instance sqllib directory.

### Authorization

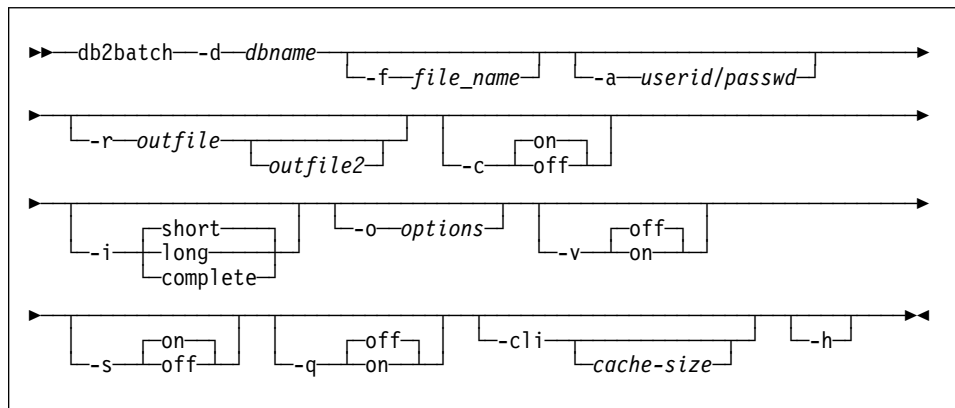
One of the following:

*sysadm*

### Required Connection

None. This command establishes a database connection.

### Command Syntax



### Command Parameters

**-d** *dbname*

An alias name for the database against which SQL statements are to be applied. The default is the value of the **DB2DBDFT** environment variable.

**-f** *file\_name*

Name of an input file containing SQL statements. The default is standard input.

Identify comment text with two hyphens at the start of each line, that is, `--<comment>`. If it is to be included in the output, mark the comment as follows: `--#COMMENT <comment>`.

A *block* is a number of SQL statements that are treated as one, that is, information is collected for all of those statements at once, instead of one at a time. Identify the beginning of a block of queries as follows: `--#BGBLK`. Identify the end of a block of queries as follows: `--#EOBLK`.

## db2batch - Benchmark Tool

Specify one or more control options as follows: `--#SET <control option> <value>`. Valid control options are:

### *ROWS\_FETCH*

Number of rows to be fetched from the answer set. Valid values are -1 to *n*. The default value is -1 (all rows are to be fetched).

### *ROWS\_OUT*

Number of fetched rows to be sent to output. Valid values are -1 to *n*. The default value is -1 (all fetched rows are to be sent to output).

### *PERF\_DETAIL*

Specifies the level of performance information to be returned. Valid values are:

- 0 No timing is to be done.
- 1 Return elapsed time only.
- 2 Return elapsed time and CPU time.
- 3 Return a summary of monitoring information.
- 4 Return a snapshot for the database manager, the database, the application, and the statement (the latter is returned only if autocommit is off, and single statements, not blocks of statements, are being processed).
- 5 Return a snapshot for the database manager, the database, the application, and the statement (the latter is returned only if autocommit is off, and single statements, not blocks of statements, are being processed). Also return a snapshot for the bufferpools, table spaces and FCM (an FCM snapshot is only available in a multi-mode environment).

The default value is 1. A value >1 is only valid on DB2 Version 2 servers.

### *DELIMITER*

A one- or two-character end-of-statement delimiter. The default value is a semicolon (;).

### *SLEEP*

Number of seconds to sleep. Valid values are 1 to *n*.

### *PAUSE*

Prompts the user to continue.

### *TIMESTAMP*

Generates a time stamp.

### **-a** *userid/passwd*

Name and password used to connect to the database. The slash (/) must be included.

### **-r** *outfile*

An output file that will contain the query results. An optional *outfile2* will contain a results summary. The default is standard output.

### **-c**

Automatically commit changes resulting from each SQL statement.

**-i**

An elapsed time interval (in seconds).

*short*

The time taken to open the cursor, complete the fetch, and close the cursor.

*long*

The elapsed time from the start of one query to the start of the next query, including pause and sleep times, and command overhead.

*complete*

The time to prepare, execute, and fetch, expressed separately.

**-o options**

Control options. Valid options are:

**f** *rows\_fetch*

Number of rows to be fetched from the answer set. Valid values are -1 to *n*. The default value is -1 (all rows are to be fetched).

**r** *rows\_out*

Number of fetched rows to be sent to output. Valid values are -1 to *n*. The default value is -1 (all fetched rows are to be sent to output).

**p** *perf\_detail*

Specifies the level of performance information to be returned. Valid values are:

0 No timing is to be done.

1 Return elapsed time only.

2 Return elapsed time and CPU time.

3 Return a summary of monitoring information.

4 Return a snapshot for the database manager, the database, the application, and the statement (the latter is returned only if autocommit is off, and single statements, not blocks of statements, are being processed).

5 Return a snapshot for the database manager, the database, the application, and the statement (the latter is returned only if autocommit is off, and single statements, not blocks of statements, are being processed). Also return a snapshot for the bufferpools, table spaces and FCM (an FCM snapshot is only available in a multi-mode environment).

**-o** *query\_optimization\_class*

Sets the query optimization class. For a description of valid values, see the *Administration Guide*.

**-e** *explain\_mode*

Sets the explain mode under which **db2batch** runs. The explain tables must be created prior to using this option. Valid values are:

0 Run query only (default).

## db2batch - Benchmark Tool

- 1 Populate explain tables only. This option populates the explain tables and causes explain snapshots to be taken.
- 2 Populate explain tables and run query. This option populates the explain tables and causes explain snapshots to be taken.

**-v**

Verbose. Send information to standard error during query processing. The default value is off.

**-s**

Summary Table. Provide a summary table for each query or block of queries, containing elapsed time (if selected), CPU times (if selected), the rows fetched, and the rows printed. The arithmetic and geometric means for elapsed time and CPU times are provided if they were collected.

**-q**

Query output. Valid values are:

*on* Print only the *non-delimited* output of the query.

*off* Print the output of the query and all associated information. This is the default.

*del* Print only the *delimited* output of the query.

**-p**

Parallel (MPP only). Valid values are:

*-s* Single table or collocated join query. If this option is specified, the NODENUMBER function will be added to the WHERE clause of the query, and a temporary table will not be created. This option is valid only if the query contains a single table in the FROM clause, or if the tables contained in the FROM clause are collocated.

*-t table* Specifies the table to use for an INSERT INTO statement. If the query contains multiple tables in the FROM clause, and the tables are not collocated, the result set must first be inserted into a temporary table, and then a SELECT from this temporary table must be performed on all nodes.

If neither the *-s* nor the *-t* option is specified, the tool creates a temporary table by default.

If a *local* output file is specified (using the *-r* option), the output from each node will go into a separate file with the same name on each node). If a file that is on an NFS-mounted file system is specified, all of the output will go into this file.

**-cli**

Run **db2batch** in CLI mode. The default is to use embedded dynamic SQL. The statement memory can be set manually, using the *cache-size* parameter.

*cache-size*

Size of the statement memory, expressed as number of statements. The default value is 25. If the utility encounters an SQL statement that has

## db2batch - Benchmark Tool

already been prepared, it will reuse the old plans. This parameter can only be set when **db2batch** is run in CLI mode.

**-h**

Display help information. When this option is specified, all other options are ignored, and only the help information is displayed.

### Example

For a detailed discussion on the use of **db2batch**, see the *Administration Guide*.

### Usage Notes

Although SQL statements can be up to 32 698 characters in length, no text line in the input file can exceed 3 898 characters, and long statements must be divided among several lines. Statements must be terminated by a delimiter (the default is a semicolon).

SQL statements are executed with the repeatable read (RR) isolation level.

### See Also

“db2sql92 - SQL92 Compliant SQL Statement Processor” on page 54.

## db2bfd - Bind File Description Tool

---

### db2bfd - Bind File Description Tool

Displays the contents of a bind file. This utility, which can be used to examine and to verify the SQL statements within a bind file, as well as to display the precompile options used to create the bind file, may be helpful in problem determination related to an application's bind file.

On UNIX based systems, the tool is located in the `misc` subdirectory of the `sql1ib` directory of the instance; on OS/2 or the Windows operating system, it is located in the `bin` subdirectory of the `sql1ib` directory of the instance.

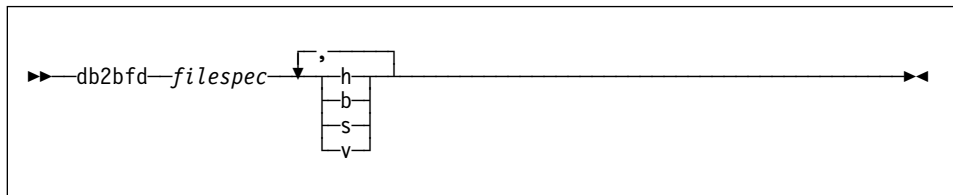
#### Authorization

None

#### Required Connection

None

#### Command Syntax



#### Command Parameters

*filespec*

Name of the bind file whose contents are to be displayed.

**h**

Display help information. When this option is specified, all other options are ignored, and only the help information is displayed.

**b**

Display the bind file header.

**s**

Display the SQL statements.

**v**

Display the host variable declarations.

---

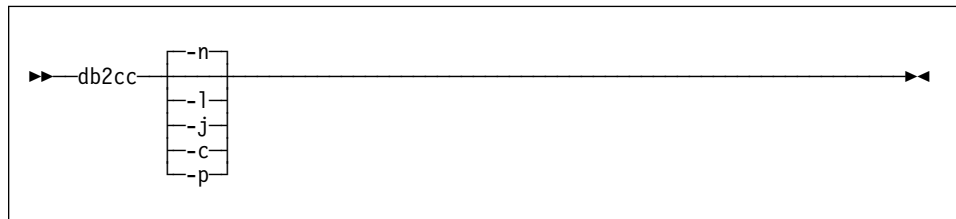
### db2cc - Start Control Center

Starts the Control Center. The Control Center is an easy-to-use graphical interface that displays database objects (such as databases, tables, and packages) and their relationship to one another.

### Authorization

*sysadm*

### Command Syntax



### Command Parameters

- n** Opens the Control Center.
- l** Opens the Journal.
- j** Opens the Script Center.
- c** Opens the Command Center.
- p** Opens the Tools Settings.

### Usage Notes

For general information about the Control Center, see the *Administration Getting Started* book. Detailed information is provided through the online help facility within the Control Center.

## db2cidmg - Remote Database Migration

---

### db2cidmg - Remote Database Migration

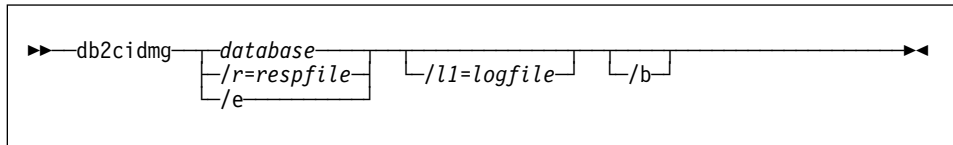
Supports remote unattended migration in the Configuration, Installation, and Distribution (CID) architecture environment.

#### Authorization

One of the following:

*sysadm*  
*dbadm*

#### Command Syntax



#### Command Parameters

*database*

Specifies an alias name for the database which is to be migrated to DB2 Version 3.1. If not specified, a response file or /e must be provided for program invocation. Note that the database alias must be cataloged on the target workstation. However, it can be a local or a remote database.

*/r*

Specifies a response file to be used for CID migration. The response file is an ASCII file containing a list of databases which are to be migrated. If not specified, a database alias or /e must be provided for program invocation.

*/e*

Indicates that every single database cataloged in the system database directory is to be migrated. If /e is not specified, a database alias or a response file must be provided.

*/ll*

Specifies the path name of the file to which error log information from remote workstations can be copied after the migration process is completed. If more than one database is specified in the response file, the log information for each database migration is appended to the end of the file. Regardless of whether /ll is specified or not, a log file with the name DB2CIDMG.LOG is generated and kept in the workstation's file system where the database migration has been performed.

*/b*

Indicates that all packages in the database are to be rebound once migration is complete.



---

## db2ckmig - Database Pre-migration Tool

Verifies that a database can be migrated. For detailed information about using this tool, see one of the *Quick Beginnings* books.

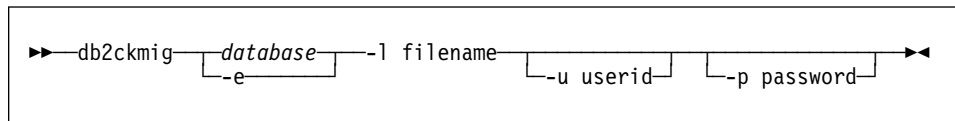
### Authorization

*sysadm*

### Required Connection

None

### Command Syntax



### Command Parameters

*database*

Specifies an alias name of a database to be scanned.

**-e**

Specifies that all local cataloged databases are to be scanned.

**-l**

Specifies a log file to keep a list of errors and warnings generated for the scanned database.

**-u**

Specifies the user ID of the system administrator.

**-p**

Specifies the password of the system administrator's user ID.

### Usage Notes

To verify the state of a database:

1. Logon as the instance owner.
2. Issue the **db2ckmig** command.
3. Check the log file. If it shows errors, refer to Table 1 on page 16 for suggested corrective actions.

**Note:** The log file displays the errors that occur when the **db2ckmig** command is run. Check that the log is empty before continuing with the migration process.

## db2ckmig - Database Pre-migration Tool

<b>Error</b>	<b>Action</b>
A database is in Backup pending state	Perform a backup of the database.
A database is in Roll-forward pending state	Recover the database as required; perform or resume a Roll-forward Database.
Table space ID not in normal state	Recover the database and table space as required; perform or resume a Roll-forward Database.
A database is in Database transaction inconsistent state	Restart the database to return it to a consistent state.
The database contains database objects that have a schema name of SYSCAT, SYSSTAT, or SYSFUN	<p>These schema names are reserved for the database manager. To correct this error, do the following:</p> <ol style="list-style-type: none"><li><b>1</b> Backup the database.</li><li><b>2</b> Export the data from the database object (catalogs or tables).</li><li><b>3</b> Drop the object.</li><li><b>4</b> Re-create the object with the corrected schema name.</li><li><b>5</b> Import or load the data into the object.</li><li><b>6</b> Run <b>db2ckmig</b> against the database again, ensuring that the database passes the <b>db2ckmig</b> check.</li><li><b>7</b> Make a backup copy of the database.</li></ol>

---

### db2cli - DB2 Interactive CLI

Launches the interactive Call Level Interface environment for design and prototyping in CLI. Located in the `sqllib/samples/cli/` subdirectory of the home directory of the database instance owner.

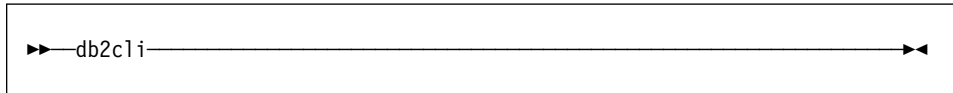
#### Authorization

None

#### Required Connection

None

#### Command Syntax



```
▶ db2cli ◀
```

#### Command Parameters

None

#### Usage Notes

DB2 Interactive CLI consists of a set of commands that can be used to design, prototype, and test CLI function calls. It is a programmers' testing tool provided for the convenience of those who want to use it, and IBM makes no guarantees about its performance. DB2 Interactive CLI is not intended for end users, and so does not have extensive error-checking capabilities.

Two types of commands are supported:

##### *CLI commands*

Commands that correspond to (and have the same name as) each of the function calls that is supported by IBM CLI

##### *Support commands*

Commands that do not have an equivalent CLI function.

Commands can be issued interactively, or from within a file. Similarly, command output can be displayed on the terminal, or written to a file. A useful feature of the CLI command driver is the ability to capture all commands that are entered during a session, and to write them to a file, thus creating a *command script* that can be rerun at a later time.

For more information about this utility, see the file `intcli.doc`, which is also located in the `sqllib/samples/cli/` subdirectory of the home directory of the database instance owner.

## db2cmd - Open DB2 Command Window

---

### db2cmd - Open DB2 Command Window

Opens the CLP-enabled DB2 window, and initializes the DB2 command line environment. Issuing this command is equivalent to clicking on the *DB2 Command Window* icon.

This command is available on Windows NT only.

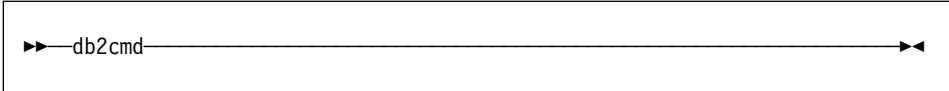
#### Authorization

None

#### Required Connection

None

#### Command Syntax



▶—db2cmd—◀

#### Command Parameters

None

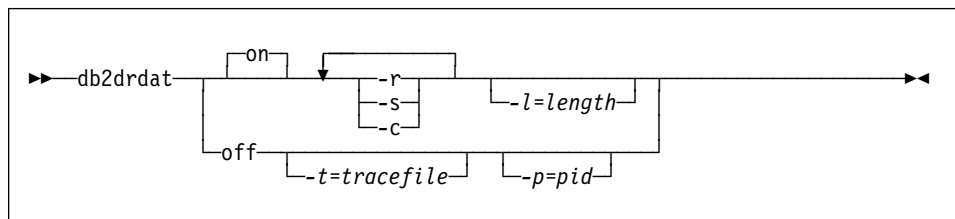
## db2drdat - DRDA Trace

Allows the user to capture the DRDA data stream exchanged between a DRDA Application Requestor (AR) and the DRDA Application Server (AS). Although this tool is most often used for problem determination, by determining how many sends and receives are required to execute an application, it can also be used for performance tuning in a client/server environment.

### Authorization

None

### Command Syntax



### Command Parameters

*on*

Turns on AS trace events (all if none specified).

*off*

Turns off AS trace events.

*-r*

Traces DRDA requests received from the DRDA AR.

*-s*

Traces DRDA replies sent to the DRDA AR.

*-c*

Traces the SQLCA received from the DRDA server on the host system. This is a formatted, easy-to-read version of *not null* SQLCAs.

*-l*

Specifies the size of the buffer used to store the trace information.

*-p*

Traces events only for this process. If *-p* is not specified, all agents with incoming DRDA connections on the server are traced.

**Note:** The *pid* to be traced can be found in the *agent* field returned by "LIST APPLICATIONS" on page 225.

*-t*

Specifies the destination for the trace. If a file name is specified without a complete path, missing information is taken from the current path.

**Note:** If *tracefile* is not specified, messages are directed to `db2drdat.dmp` in the current directory.

## db2drdat - DRDA Trace

### Usage Notes

Do not issue **db2trc** commands while **db2drdat** is active (for information about the **db2trc** command, see the *Troubleshooting Guide*).

**db2drdat** writes the following information to *tracefile*:

1. -r
  - Type of DRDA request
  - Receive buffer.
2. -s
  - Type of DRDA reply/object
  - Send buffer.
3. CPI-C error information
  - Severity
  - Protocol used
  - API used
  - Local LU name
  - Failed CPI-C function
  - CPI-C return code.

The command returns an exit code. A zero value indicates that the command completed successfully, and a nonzero value indicates that the command was not successful.

**Note:** If **db2drdat** sends the output to a file that already exists, the old file will be erased unless the permissions on the file do not allow it to be erased, in which case the operating system will return an error.

## db2empfa - Enable Multi-page File Allocation

---

### db2empfa - Enable Multi-page File Allocation

Enables multi-page file allocation for a database.

#### Scope

This command only affects the node on which it is executed.

#### Authorization

*sysadm*

#### Required Connection

None. This command establishes a database connection.

#### Command Syntax

The diagram shows the command syntax for db2empfa. It consists of a rectangular box containing the text: `db2empfa database-alias`. The text is preceded by a right-pointing arrow and followed by a double-headed arrow, indicating the command and its parameter.

#### Command Parameters

*database-alias*

Specifies the alias of the database for which multi-page file allocation is to be enabled.

#### Usage Notes

This utility:

- Connects to the database partition on a node (where applicable) in exclusive mode
- In all SMS table spaces, allocates empty pages to fill up the last extent in all data and index files which are larger than one extent
- Changes the value of the database configuration parameter *multipage\_alloc* to YES
- Disconnects.

Since **db2empfa** connects to the database partition on a node in exclusive mode, it cannot be run concurrently on the catalog node, or on any other node.

## db2eva - Event Analyzer

---

### db2eva - Event Analyzer

Starts the event analyzer, allowing the user to trace performance data produced by DB2 event monitors that have their data directed to files. See the *System Monitor Guide and Reference* for more information on event monitors.

#### Authorization

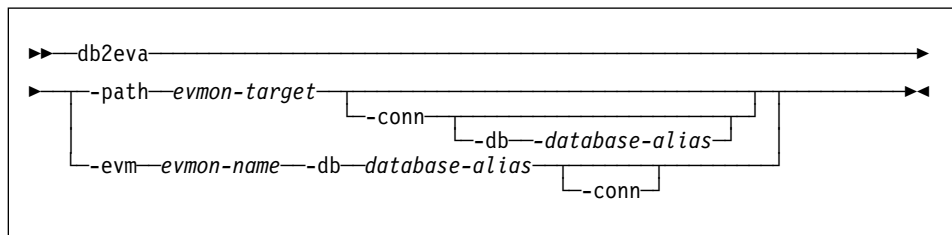
None, unless connecting to the database and selecting from the catalogs (-evm, -db, and -conn); then one of the following is required:

*sysadm*  
*sysctrl*  
*sysmaint*  
*dbadm*

#### Required Connection

None

#### Command Syntax



#### Command Parameters

**-path** *evmon-target*

Specifies the directory containing the event monitor trace files.

**-conn**

Requests that **db2eva** maintain a connection to the database specified with `-db`, or if `-db` is not used, then to the database specified in the event monitor trace header. Maintaining a connection allows the event analyzer to obtain information not contained in the trace files (for example, the text for static SQL). A statement event record contains the package creator, package, and section number; when `-conn` is specified, **db2eva** can retrieve the text from the database system catalog (`sysibm.sysstmt`).

**-db** *database-alias*

Specifies the name of the database defined for the event monitor. If `-path` is specified, the database name in the event monitor trace header is overridden.



### Usage Notes

Although there is no required connection, **db2eva** will attempt to connect to the database if the `-conn`, or the `-evm` and the `-db` options are used. If the user can access the database and has the appropriate authorization, the SQL text for static statements can be displayed. Without the required access or authority, only the text for dynamic statements is available.

There are two methods for reading event monitor traces:

1. Specifying the directory where the trace files are located (using the `-path` option). This allows users to move trace files from a server and analyze them locally. This can be done even if the event monitor has been dropped.
2. Specifying the database and event monitor names allows automatic location of the trace files. The event analyzer connects to the database, and issues a `select target from sysibm.syseventmonitors` to locate the directory where the event monitor writes its trace files. The connection is then released, unless `-conn` was specified. This method cannot be used if the event monitor has been dropped.

**Note:** The event analyzer can be used to analyze the data produced by an active event monitor. However, event monitors buffer their data before writing it to disk; therefore, some information may be missing. Turn off the event monitor, thereby forcing it to flush its buffers.

## db2evmon - Event Monitor Productivity Tool

---

### db2evmon - Event Monitor Productivity Tool

Formats event monitor file and named pipe output, and writes it to standard output. Located in the `misc` subdirectory of the `sql1lib` directory of the instance.

**Note:** This productivity tool is provided *as is*, without any warranty of any kind, including the warranties of merchantability and fitness for a particular purpose, which are expressly disclaimed.

#### Authorization

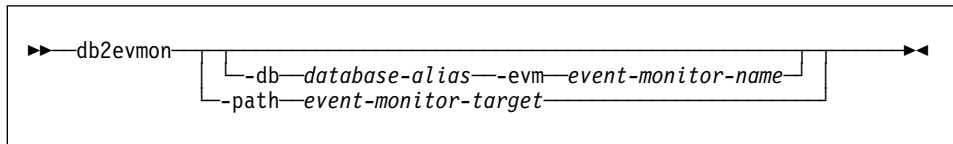
None, unless connecting to the database (`-evm`, `-db`); then, one of the following is required:

`sysadm`  
`sysctrl`  
`sysmaint`  
`dbadm`

#### Required Connection

None

#### Command Syntax



#### Command Parameters

- db** *database-alias*  
Specifies the database whose data is to be displayed.
- evm** *event-monitor-name*  
The one-part name of the event monitor. An ordinary or delimited SQL identifier.
- path** *event-monitor-target*  
Specifies the directory containing the event monitor trace files.

#### Usage Notes

If the data is being written to files, the tool formats the files for display using standard output. In this case, the monitor is turned on first, and any event data in the files is displayed by the tool. To view any data written to files after the tool has been run, reissue **db2evmon**.

If the data is being written to a pipe, the tool formats the output for display using standard output as events occur. In this case, the tool is started *before* the monitor is turned on.

---

### db2exfmt - Explain Table Format Tool

Formats the contents of the explain tables.

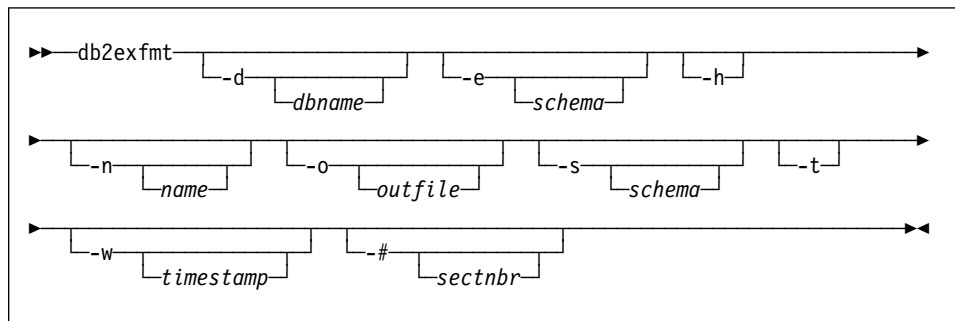
#### Authorization

Read access to the explain tables being formatted.

#### Required Connection

None

#### Command Syntax



#### Command Parameters

- d** *dbname*  
Name of the database containing packages.
- e** *schema*  
Explain table schema.
- h**  
Display help information. When this option is specified, all other options are ignored, and only the help information is displayed.
- n** *name*  
Name of the source of the explain request (SOURCE\_NAME).
- o** *outfile*  
Output file name.
- s** *schema*  
Schema or qualifier of the source of the explain request (SOURCE\_SCHEMA).
- t**  
Direct the output to the terminal.
- w** *timestamp*  
Explain time stamp.
- #** *sectnbr*  
Section number in the source. Use zero for all sections.

## db2exfmt - Explain Table Format Tool

### Usage Notes

Users are prompted for any parameter values that are not supplied, or that are incompletely specified, except in the case of the `-h` option.

If an Explain table schema is not provided, the value of the environment variable **USER** is used as the default. If this variable is not found, the user is prompted for an Explain table schema.

Source name, source schema, and Explain time stamp may be supplied in LIKE predicate form, which allows the percent sign (%) and the underscore (\_) to be used as pattern matching characters to select multiple sources with one invocation. For the latest explained statement, the explain time can be specified as `-1`.

If `-o` is specified without a file name, and `-t` is not specified, the user is prompted for a file name (the default name is `db2exfmt.out`). If neither `-o` nor `-t` is specified, the user is prompted for a file name (the default option is terminal output). If `-o` and `-t` are both specified, the output is directed to the terminal.

For more information about **db2exfmt**, see the *Administration Guide*.

### See Also

"db2expln - DB2 SQL Explain Tool" on page 27.

## db2expln - DB2 SQL Explain Tool

Describes the access plan selection for static SQL statements in packages that are stored in the DB2 common server system catalogs. Given a database name, package name, package creator, and section number, the tool interprets and describes the information in these catalogs. This tool is located in the `misc` subdirectory of the instance `sql1lib` directory.

### Authorization

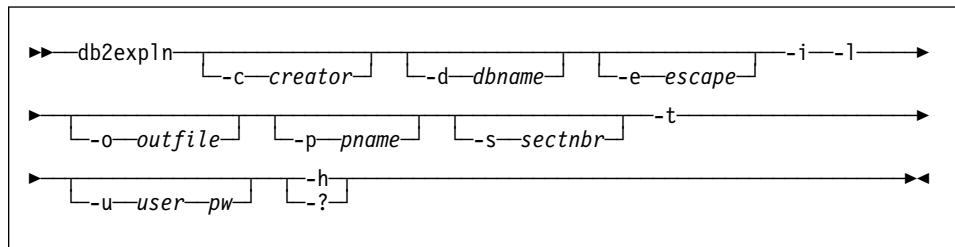
One of the following:

`sysadm`  
`dbadm`

### Required Connection

A database connection is temporarily established by this command during processing.

### Command Syntax



### Command Parameters

- `-c creator`  
User ID of the package creator.
- `-d dbname`  
Alias name of the database that contains the packages to be explained.
- `-e escape`  
Escape character for LIKE predicates.
- `-i`  
Show operator ID numbers.
- `-l`  
Respect case when processing package names.
- `-o outfile`  
Output file name.
- `-p pname`  
The name of the package to be explained.
- `-s sectnbr`  
The section number to be explained within a package. To have all sections in the package explained, specify zero.

## db2expln - DB2 SQL Explain Tool

- t** Request terminal output.
- u *user pw*** Name and password used to connect to a database.
- h/-?** Display help information. When this option is specified, all other options are ignored, and only the help information is displayed.

### Usage Notes

Package name and creator may be supplied in LIKE predicate form, which allows the percent sign (%) and the underscore (\_) to be used as pattern matching characters to select multiple packages with one invocation. To prevent the percent sign and the underscore from being used as pattern matching characters, precede them with the escape character (-e option).

For example, the following command could be issued to explain the package TESTID.CALC%:

```
db2expln -c TESTID -p CALC%.
```

This would also, however, explain any other plans that start with CALC, such as TESTID.CALCSUM or TESTID.CALCIUM. To explain just TESTID.CALC%, use the command:

```
db2expln -c TESTID -e ! -p CALC!%
```

The option -e ! specifies that ! is to be used as the escape character. In the package name, !% is therefore interpreted as the percent character, not the pattern character. For more information about LIKE and escape characters, see the *SQL Reference*.

Users are prompted for any parameter values that are not supplied, or that are incompletely specified, except in the case of the -e, -h, -l, -p, or -w option.

If -o is specified without a file name, and -t is not specified, the user is prompted for a file name (the default name is db2expln.out). If neither -o nor -t is specified, the user is prompted for a file name (the default option is terminal output). If -o and -t are both specified, the output is directed to the terminal.

For more information about **db2expln**, see the *Administration Guide*.

### See Also

“db2exfmt - Explain Table Format Tool” on page 25.

## db2gov - DB2 Governor

Monitors and changes the behavior of applications that run against a database. By default, a daemon is started on every logical node, but the front-end utility can be used to start a single daemon at a specific node to monitor the activity against the database partition at that node.

Each governor daemon collects statistics about the applications running against a database. It then checks these statistics against the rules that were specified in the governor configuration file that applies to that specific database. The governor then acts according to these rules. For example, a rule may indicate that an application is using too much resource. In this situation, the governor may change the application's priority or force it off the database, according to instructions specified in the governor configuration file.

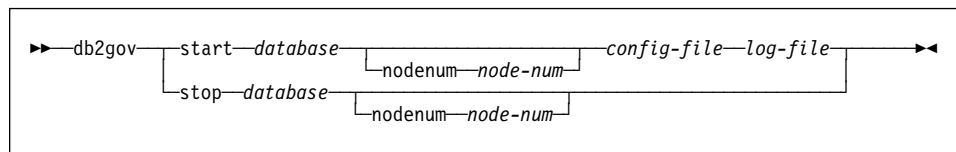
### Authorization

*sysctrl*

### Required Connection

Instance. This command establishes an implicit instance attachment.

### Command Syntax



### Command Parameters

#### **start** *database*

Starts the governor daemon that will monitor the specified database. The database name or alias specified must be identical to that specified in the governor configuration file.

#### **nodenum** *node-num*

Specifies the node on which to start or stop the governor daemon. The number is the same as that specified in the `$HOME/sqllib/db2nodes.cfg` file.

#### *config-file*

Specifies the configuration file to use when monitoring the database. The default location of the configuration file is the `$HOME/sqllib` directory. If the specified file cannot be found in the default directory, the front-end assumes that the name is a fully specified path name, or that the file exists in the current directory.

## db2gov - DB2 Governor

### *log-file*

Specifies the base name of the file to which the governor will write log records. The log file is stored in the `$HOME/sqllib/log` directory. The number of the node on which the governor is running is automatically appended to the log file name (for example, `mylog.0000`, `mylog.0001`, `mylog.0002`).

### **stop database**

Stops the governor daemon that is monitoring the specified database. When the front-end utility stops the governor on all nodes, it reads the `$HOME/sqllib/db2nodes.cfg` file, and sends a command to each database node to call the governor front-end utility with the stop parameter. This stops the daemon at each node.

## Usage Notes

### **Governor Configuration File**

The first task in the governor daemon's execution loop is to check whether its configuration file has changed or has not yet been read. If either condition is true, the daemon reads the file; this permits an administrator to change the behavior of the governor daemon while it is running.

The configuration file must reside in a directory that is mounted across all the database nodes, because the governor daemon on each node must be able to read the same configuration file.

The file consists of rules and comments. Most entries can be specified in uppercase, lowercase, or mixed case characters. The exception is *applname*, which is case sensitive. Each rule in the file must be followed by a semicolon (;). Comments are enclosed by braces ({}).

The following rules are specified only once in a configuration file:

### *account n*

Specifies that the governor is to write account records containing CPU usage statistics for each connection, at intervals of *n* minutes.

**Note:** This option is not available on Windows NT or OS/2.

If a short connect session occurs entirely within the account interval, no log record is written. When log records are written, they contain CPU statistics that reflect CPU usage since the previous log record for the connection. If the governor is stopped and then restarted, CPU usage may be reflected in two log records; these can be identified through the application IDs in the log records. For more information about governor log files, see page 35).

### *dbname*

The name or alias of the database to be monitored.

### *interval*

The interval, in seconds, at which the daemon wakes up. The default value is 120.



The following rule clauses are combined to form a rule. The rule, not the clause, is followed by a semicolon. Clauses can be specified only once in a rule, but can be specified in more than one rule. Clauses must be specified in the order shown. In the description that follows, [ ] indicates an optional clause.

*[desc]*

Specifies a text description of the rule. The description must be enclosed by either single or double quotation marks.

*[time]*

Specifies the time period during which the rule is to be evaluated.

The time period must be specified in the form hh:mm hh:mm. For example: time 8:00 18:00. If this clause does not appear in a rule, the rule is valid 24 hours a day.

*[authid]*

Specifies one or more authorization IDs under which the application is executing. Multiple IDs must be separated by a comma. For example: authid gene, michael, james. If this clause does not appear in a rule, the rule applies to all authorization IDs.

*[applname]*

Specifies the name of the executable (or object file) that makes the connection to the database.

Multiple application names must be separated by a comma. For example: applname db2bp, batch, geneprog. If this clause does not appear in a rule, the rule applies to all application names.

**Notes:**

1. Application names are case sensitive.
2. The database manager truncates all application names to 20 characters. Ensure that the application to be governed is uniquely identified by the first 20 characters of its application name; otherwise, an unintended application may be governed. Application names specified in the governor configuration file are truncated to 20 characters to match their internal representation.

*setlimit*

Specifies one or more limits for the governor to check. The limits can only be -1 or greater than zero. For example: cpu -1 locks 1000 rowsse1 10000). At least one of the following limits must be specified:

*cpu n*

Specifies the number of CPU seconds that can be consumed by an application. If -1 is specified, the governor does not limit the application's CPU usage.

**Note:** This option is not available on Windows NT or OS/2.

*idle n*

Specifies the number of seconds that a connection can be idle before an action is taken. Can be used to force a connection

## db2gov - DB2 Governor

that is not actively working on a query. If -1 is specified, the governor does not limit the connection's idle time.

*locks n*

Specifies the number of locks that an application can hold. If -1 is specified, the governor does not limit the number of locks held by the application.

*rowsread n*

Specifies the number of rows that are read from disk by the database manager on behalf of an application before an action is taken. If -1 is specified, the governor does not limit the number of rows that can be read.

*rowsssel n*

Specifies the number of rows that are returned to the application. This value will only be non-zero at the coordinator node. If -1 is specified, the governor does not limit the number of rows that can be selected.

*uowtime n*

Specifies the number of seconds that can elapse from the time that a unit of work (UOW) first becomes active. If -1 is specified, the elapsed time is not limited.

*[action]*

Specifies the action to take if one or more of the specified limits is exceeded. The following actions can be specified:

*force*

Issue "FORCE APPLICATION" on page 167 to terminate the coordinator agent servicing the application.

*priority n*

Change the priority of agents working for the application. A lower value of *n* assigns a higher priority to the agents.

For this parameter to be effective, the *agentpri* database manager parameter must be set to the default value; otherwise, it overrides the priority clause.

*schedule [class]*

Schedule applications with the goal of optimizing response time while preserving a reasonable degree of fairness. The governor enforces its schedule by setting priorities for the agents working on the applications, using query cost estimates calculated by the DB2 internal query compiler. If the class option is specified, all applications chosen by the rule are scheduled among themselves only. If this option is not specified, the governor uses one or more classes, with scheduling within each class. Prioritization of an application is based on its:

- Lock heat in the class

An application that is holding up many other applications through locking will be given a high priority.

- Age

To ensure fairness, an application that has been in the system for a long time will be given a high priority.

- Estimated remaining running time.

To improve response time, an application that is estimated to be close to finishing will be given a high priority.

**Note:** Applications that are not covered by any schedule rule will run at the highest priority.

The schedule action can:

- Ensure that applications in different groups each get time without all applications splitting time evenly.

For instance, if 12 applications (three short, five medium, and six long) are running at the same time, they may all have poor response times because they are splitting the CPU. The database administrator can set up two groups, medium-length applications and long-length applications. Using priorities, the governor permits all the short applications to run, and ensures that at most three medium and three long applications run simultaneously. To achieve this, the governor configuration file contains one rule for medium-length applications, and another rule for long applications. The following example shows a portion of a governor configuration file that illustrates this point:

```
desc "Group together medium applications in 1 schedule class"
applname medq1, medq2, medq3, medq4, medq5
setlimit cpu -1
action schedule class;
```

```
desc "Group together long applications in 1 schedule class"
applname longq1, longq2, longq3, longq4, longq5, longq6
setlimit cpu -1
action schedule class;
```

- Ensure that each of several user groups (for example, organizational departments) gets equal prioritization.

If one group is running a large number of applications, the administrator can ensure that other groups are still able to obtain reasonable response times for their applications.

For instance, in a case involving three departments (Finance, Inventory, and Planning), all the Finance users could be put into one group, all the Inventory users could be put into a second, and all the Planning users could be put into a third group. The processing power would be split more or less evenly among the three departments.

## db2gov - DB2 Governor

The following example shows a portion of a governor configuration file that illustrates this point:

```
desc "Group together Finance department users"
authid tom, dick, harry, mo, larry, curly
setlimit cpu -1
action schedule class;
```

```
desc "Group together Inventory department users"
authid pat, chris, jack, jill
setlimit cpu -1
action schedule class;
```

```
desc "Group together Planning department users"
authid tara, dianne, henrietta, maureen, linda, candy
setlimit cpu -1
action schedule class;
```

- Let the governor schedule all applications.

If the class option is not included with the action, the governor creates its own classes based on how many applications fall under the schedule action, and puts applications into different classes based on the DB2 query compiler's cost estimate for the query that the application is running. The administrator can choose to have all applications scheduled by not qualifying which applications are chosen. That is, no *applname* or *authid* clauses are supplied, and the *setlimit* clause causes no restrictions.

**Note:** If a limit is exceeded, and the action clause is not specified, the governor reduces the priority of agents working for the application by 10.

Following is an example of a governor configuration file:

```
{ Wake up once a second, the database name is ibmsamp1 }
interval 1; dbname ibmsamp1;
```

```
desc "CPU restrictions apply 24 hours a day to everyone"
setlimit cpu 600 rowsel 1000000;
```

```
desc 'Allow no UOW to run for more than an hour'
setlimit uowtime 3600;
```

```
desc "Slow down a subset of applications"
applname jointA, jointB, jointC, quryA
setlimit cpu 3 locks 1000 rowsel 500;
```

```
desc "Have governor prioritize these 6 long apps in 1 class"
applname longq1, longq2, longq3, longq4, longq5, longq6
setlimit cpu -1
action schedule class;
```

```
desc "Schedule all applications run by the planning dept"
authid planid1, planid2, planid3, planid4, planid5
setlimit cpu -1
action schedule;

desc "Schedule all CPU hogs in one class which will control consumption"
setlimit cpu 3600
action schedule class;

desc 'Slow down the use of db2 CLP by the novice user'
authid novice
applname db2bp
setlimit cpu 5 locks 100 rowsssel 250;

desc "During day hours do not let anyone run for more than 10 seconds"
time 8:30 17:00 setlimit cpu 10 action force;

desc 'Allow users doing performance tuning to run some of
their applications during lunch hour'
time 12:00 13:00 authid ming, geoffrey, john, bill
applname tpcc1, tpcc2, tpcA, tpvG setlimit cpu 600 rowsssel 120000 action force;

desc "Some people should not be limited -- database administrator
and a few others. As this is the last specification in the
file, it will override what came before."
authid gene, hershel, janet setlimit cpu -1 locks -1 rowsssel -1;

desc 'Increase the priority of an important application so it always
completes quickly'
applname V1app setlimit cpu 1 locks 1 rowsssel 1 priority -20;
```

### Governor Log Files

A separate log file exists for each governor daemon. This prevents file-locking bottlenecks that would result from many governor daemons writing to the same file at the same time.

The log files are stored in the `$HOME/sql1lib/log` directory. The base name for the log files is provided when the **db2gov** command is issued. Ensure that the log file name contains the database name, because there will be a log file for each node of each database that is being governed. The number of the node on which the governor is running is automatically appended to the log file name to ensure that the file name for each governor is unique.

Each record in the log file has the following format:

*Date Time NodeNum RecType Message*

The *Date* and *Time* fields are in the form `yyyy-mm-dd-hh.mm.ss`, so that the log files for each node can be merged by sorting on the combination of these two fields.

The *NodeNum* field indicates the number of the node on which the governor is running.

The *RecType* field contains different values, depending on the type of log record being written. The values that can be recorded are:

## db2gov - DB2 Governor

- START, to indicate that the governor was started
- FORCE, to indicate that an application was forced
- NICE, to indicate that the priority of an application was changed
- ERROR, to indicate an error
- WARNING, to indicate a warning
- READCFG, to indicate that the governor read the configuration file
- STOP, to indicate that the governor was stopped
- ACCOUNT, to indicate the rule selection criteria that an application has met. The fields are:
  - authid
  - appl\_id
  - appl\_con\_time
  - written\_usr\_cpu
  - written\_sys\_cpu

Because standard values are written, the log files can be queried for different types of actions. The *Message* field provides other nonstandard information that varies according to the value in the *Rectype* field. For instance, a FORCE or a NICE record indicates that application information exists in the *Message* field, while an ERROR record includes an error message.

Following is an example of a governor log file:

```
1996-12-11 14.54.52 0 START      Database = TQTEST
1996-12-11 14.54.52 0 READCFG   Config = /u/db2instance/sqllib/tqtest.cfg
1996-12-11 14.54.53 0 ERROR     SQLMON Error: SQLCode = -1032
1996-12-11 14.54.54 0 ERROR     SQLMONSZ Error: SQLCode = -1032
```

To query a governor log file, use “db2govlg - DB2 Governor Log Query” on page 37.

### See Also

“db2govlg - DB2 Governor Log Query” on page 37.

---

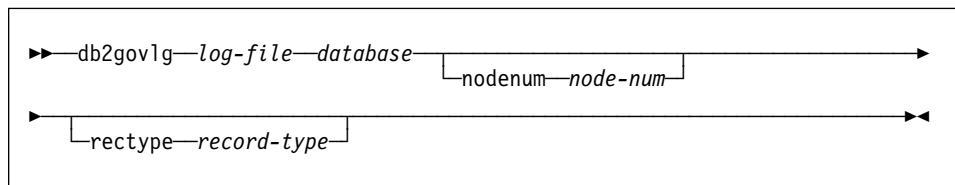
### db2govlg - DB2 Governor Log Query

Extracts records of specified type from the governor log files (see “db2gov - DB2 Governor” on page 29). The DB2 governor monitors and changes the behavior of applications that run against a database.

#### Authorization

None

#### Command Syntax



#### Command Parameters

*log-file*

The base name of one or more log files that are to be queried.

*database*

Name of the database that the governor is monitoring.

**nodenum** *node-num*

Number of the node on which the governor is running.

**rectype** *record-type*

The type of record that is to be queried. Valid record types are:

- START
- FORCE
- NICE
- ERROR
- WARNING
- READCFG
- STOP
- ACCOUNT

#### See Also

“db2gov - DB2 Governor” on page 29.

## db2icrt - Create Instance

---

### db2icrt - Create Instance

Creates DB2 instances.

On UNIX based systems, this utility is located in the DB2DIR/instance directory, where DB2DIR represents /usr/lpp/db2\_05\_00 on AIX, and /opt/IBMcdb2/V5.0 on both Solaris and HP-UX. On OS/2 or the Windows operating system, it is located in the \sql1lib\bin subdirectory.

For a complete description of this command, see one of the *Quick Beginnings* books.



---

### db2idrop - Remove Instance

Removes a DB2 instance that was created by “db2icrt - Create Instance” on page 38. Removes the instance entry from the list of instances.

On UNIX based systems, this utility is located in the `DB2DIR/instance` directory, where `DB2DIR` represents `/usr/lpp/db2_05_00` on AIX, and `/opt/IBMdb2/V5.0` on both Solaris and HP-UX. On OS/2 or the Windows operating system, it is located in the `\sqllib\bin` subdirectory.

For a complete description of this command, see one of the *Quick Beginnings* books.

## db2ilist - List Instances

---

### db2ilist - List Instances

Lists all the instances that are available on a system.

On UNIX based systems, this utility is located in the DB2DIR/instance directory, where DB2DIR represents /usr/lpp/db2\_05\_00 on AIX, and /opt/IBMdb2/V5.0 on both Solaris and HP-UX. On OS/2 or the Windows operating system, it is located in the \sql11ib\bin subdirectory.

For a complete description of this command, see one of the *Quick Beginnings* books.

---

### db2imigr - Migrate Instance

Migrates an existing instance following installation of the database manager.

On UNIX based systems, this utility is located in the `DB2DIR/instance` directory, where `DB2DIR` represents `/usr/lpp/db2_05_00` on AIX, and `/opt/IBMdb2/V5.0` on both Solaris and HP-UX. On OS/2 or the Windows operating system, it is located in the `\sqllib\bin` subdirectory.

For a complete description of this command, see one of the *Quick Beginnings* books.

## db2ipxad - Get IPX/SPX Internetwork Address

---

### db2ipxad - Get IPX/SPX Internetwork Address

Returns the DB2 server's IPX/SPX internetwork address. This command *must* be issued locally from the DB2 server machine. Issuing the command from a remote client is not supported. The internetwork address can be used on a client machine to catalog an IPX/SPX node using "direct addressing". For more information, see one of the *Quick Beginnings* books.

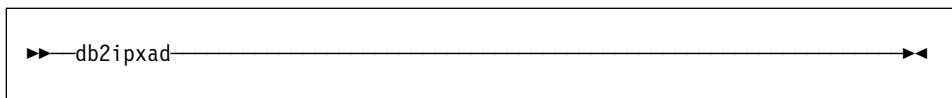
#### Authorization

None

#### Required Connection

None

#### Command Syntax



```
▶—db2ipxad—▶
```

#### Command Parameters

None

#### See Also

"CATALOG IPX/SPX NODE" on page 126.

---

### db2iupdt - Update Instances

Updates a specified DB2 instance to enable acquisition of a new system configuration or access to function associated with the installation or removal of certain product options.

On UNIX based systems, this utility is located in the `DB2DIR/instance` directory, where `DB2DIR` represents `/usr/lpp/db2_05_00` on AIX, and `/opt/IBMdb2/V5.0` on both Solaris and HP-UX. On OS/2 or the Windows operating system, it is located in the `\sql11ib\bin` subdirectory.

For a complete description of this command, see one of the *Quick Beginnings* books.

## db2look - DB2 Statistics Extraction Tool

---

### db2look - DB2 Statistics Extraction Tool

Generates the update statements required to make the catalog statistics of a test database match those of a production database.

It is often advantageous to have a test system contain a subset of the production system's data. However, access plans selected for such a test system are not necessarily the same as those that would be selected for the production system. Both the catalog statistics and the configuration parameters for the test system must be updated to match those of the production system. This tool queries the system catalogs of a database, and outputs table space, table, index, and column information about each table in that database.

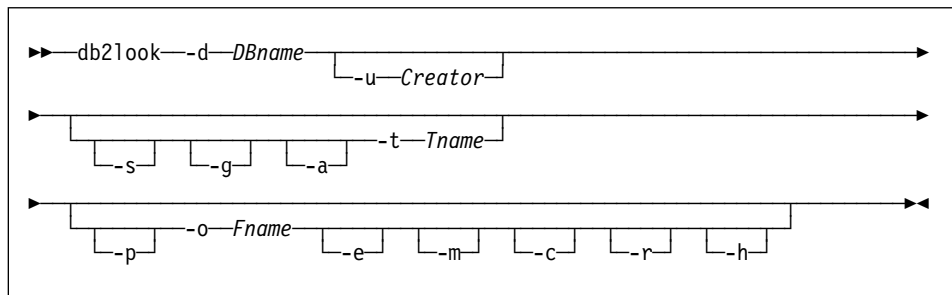
#### Authorization

SELECT privilege on the system catalogs.

#### Required Connection

None. This command establishes a database connection.

#### Command Syntax



#### Command Parameters

**-d** *DBname*

Alias name of the production database that is to be queried.

**-u** *Creator*

Creator ID. If option `-a` is specified, this parameter is ignored. If neither `-u` nor `-a` is specified, the environment variable **USER** is used.

**-s**

Generate a PostScript file.

#### Notes:

1. This option removes all LaTeX and `.tmp` PostScript files.
2. Required non-IBM software: LaTeX, `dvips`.
3. The `psfig.tex` file must be in the LaTeX input path.

## db2look - DB2 Statistics Extraction Tool

- g** Use a graph to show fetch page pairs for indices.
- Notes:**
1. This option generates a *filename.ps* file, as well as the LaTeX file.
  2. Required non-IBM software: Gnuplot.
  3. The *psfig.tex* file must be in the LaTeX input path.
- a** Generate statistics for all creators. If specified, the **-u** parameter is ignored.
- t *Tname*** Table name. Limits the output to a particular table.
- p** Use plain text format.
- o *Fname*** If using LaTeX format, write the output to *filename.tex*. If using plain text format, write the output to *filename.txt*. If this option is not specified, output is written to standard output.
- e** Extract DDL statements to recreate database data objects. This option generates a CLP script containing DDL statements to recreate tables and indexes. The CLP script can then be run against another database to recreate the database. This option can be used in conjunction with the **-m** option. It does not currently support:
- Creation of table spaces
  - Reference to table spaces
  - Creation of triggers
  - Creation of UDFs
  - Reference to UDFs.
- m** Run the program in *mimic* mode. This option generates a CLP script containing all the UPDATE statements required to capture the catalog statistics of the production database. The CLP script can then be run against a smaller test database, and the updated test database can be used to validate access plans for production. The **-p**, **-g**, and **-s** options are ignored in *mimic* mode.
- c** Do not generate COMMIT statements in *mimic* mode. The default action is to generate COMMIT statements. In addition, do not generate CONNECT or CONNECT RESET statements. If option **-m** is not specified, this option is ignored.
- r** Do not include the RUNSTATS command in *mimic* mode. The default action is to issue the RUNSTATS command. If option **-m** is not specified, this option is ignored.
- h** Display help information. When this option is specified, all other options are ignored, and only the help information is displayed.

## db2look - DB2 Statistics Extraction Tool

### Examples

Write the statistics for tables created by USER in database DEPARTMENT in LaTeX format to file output.tex:

```
db2look -d department -o output
```

Write the statistics for all tables in database DEPARTMENT in LaTeX format to file output.tex:

```
db2look -a -d department -o output
```

Write the statistics for all tables in database DEPARTMENT in plain text format to file output.txt:

```
db2look -p -a -d department -o output
```

Write the statistics for all tables in database DEPARTMENT for user JAMES. Use a graph to show page fetch pairs. Save the LaTeX output in file output.tex. Save the PostScript output in file output.ps:

```
db2look -g -s -u james -d department -o output
```

Write the replica CLP commands for table EMPLOYEE in database PAYROLL created by anyone to file employee.mimic:

```
db2look -m -a -d payroll -t employee -o employee.mimic
```



## db2migdr - Local Database Directory Migration

---

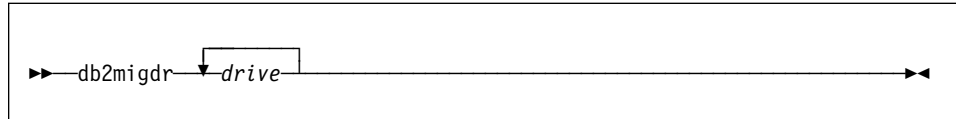
### db2migdr - Local Database Directory Migration

Migrates a down-level local database directory to the current release format, so that databases stored in the local database directory can be accessed for migration.

#### Authorization

*sysadm*

#### Command Syntax



#### Command Parameters

*drive*

Specifies the drive where the local database directory can be found. The local database directory must be named `sqlbdir`, and it must be the first level subdirectory off the drive that is specified. The following example invokes the command to migrate the local database directory from drive `c` and from drive `f`:

```
db2migdr c f
```

## db2rbind - Rebind all Packages

---

### db2rbind - Rebind all Packages

Rebinds all packages in a database.

#### Authorization

One of the following:

*sysadm*  
*dbadm*

#### Required Connection

None

#### Command Syntax

```
▶▶ db2rbind database /l logfile [/u userid /p password] ▶▶
```

#### Command Parameters

*database*

Specifies an alias name for the database whose packages are to be revalidated.

*/l*

Specifies the (optional) path and the (mandatory) file name to be used for recording errors that result from the package revalidation procedure.

*/u*

User ID (optional). This parameter must be specified if a password is specified.

*/p*

Password (optional). This parameter must be specified if a user ID is specified.

#### Usage Notes

This command uses the CLP REBIND command to attempt the revalidation of all packages in a database. Use of **db2rbind** is not mandatory. One may choose to allow package revalidation to occur implicitly when the package is first used, or to selectively revalidate particular packages with either the REBIND or the BIND command.

#### See Also

“BIND” on page 98  
“REBIND” on page 309.

---

### db2sampl - Create Sample Database

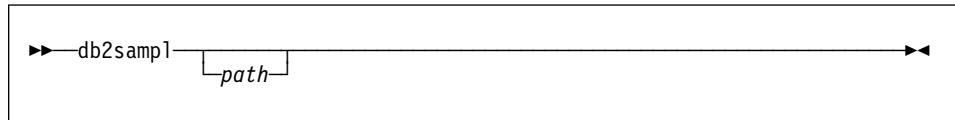
Creates a sample database named SAMPLE. For more information about this database, see the *SQL Reference*.

#### Authorization

One of the following:

*sysadm*  
*sysctrl*

#### Command Syntax



#### Command Parameters

*path*

Specifies the path on which to create the SAMPLE database. The path is a single drive letter for OS/2 and Windows.

If a path is not specified, SAMPLE is created on the default database path (the *dftdbpath* parameter in the database manager configuration file). On UNIX based systems, the default is the HOME directory of the instance owner. On OS/2 or the Windows operating system, it is the root directory (where DB2 is installed).

#### Usage Notes

This command can only be executed from server nodes. SAMPLE cannot be created on nodes that are database clients only.

The SAMPLE database is created with the instance authentication type that is specified by the database manager configuration parameter *authentication* (see “GET DATABASE MANAGER CONFIGURATION” on page 184).

The qualifiers for the tables in SAMPLE are determined by the user ID issuing the command.

If SAMPLE already exists, **db2sampl** creates the tables for the user ID issuing the command, and grants the appropriate privileges.

## db2set - DB2 Profile Registry Command

---

### db2set - DB2 Profile Registry Command

Displays, sets, or removes DB2 profile variables. An external environment registry command that supports local and remote administration, via the DB2 Administration Server, of DB2's environment variables stored in the DB2 profile registry.

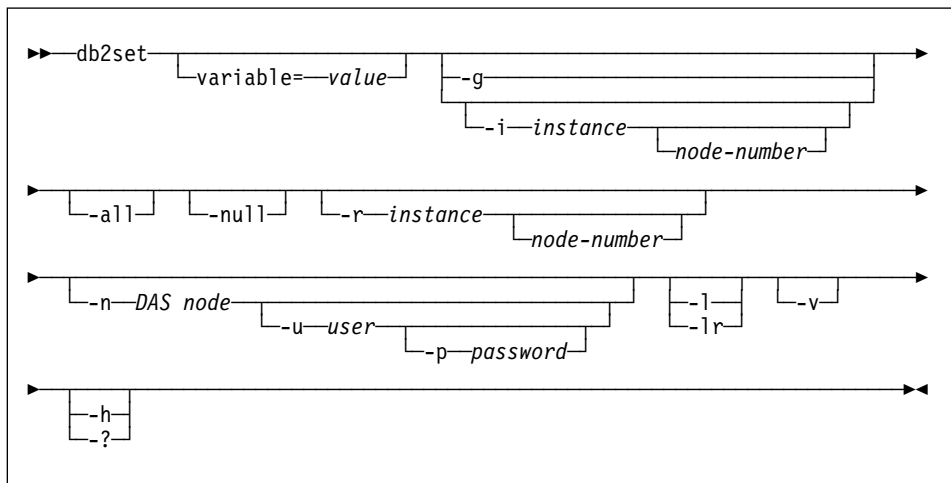
#### Authorization

*sysadm*

#### Required Connection

None

#### Command Syntax



#### Command Parameters

- g** Access the global profile variables.
- i** Specifies the instance profile to use instead of the current or default. The node number is a number listed in the `db2nodes.cfg` file.
- all** Display all occurrences of the local environment variables as defined in:
  - The environment, denoted by [e]
  - The node level registry, denoted by [n]
  - The instance level registry, denoted by [i]
  - The global level registry, denoted by [g].

## db2set - DB2 Profile Registry Command

<b>-null</b>	Set the value of the variable at the specified registry level to null. This avoids having to look up the value in the next registry level, as defined by the search order.
<b>-r</b>	Reset the profile registry for the given instance. The node number is a number listed in the <code>db2nodes.cfg</code> file.
<b>-n</b>	Specifies the remote DB2 administration server node name.
<b>-u</b>	Specifies the user ID to use for the administration server attachment.
<b>-p</b>	Specifies the password to use for the administration server attachment.
<b>-l</b>	List all instance profiles.
<b>-lr</b>	List all supported registry variables.
<b>-v</b>	Verbose mode.
<b>-h/-?</b>	Display help information. When this option is specified, all other options are ignored, and only the help information is displayed.

### Examples

- Display all defined profiles (DB2 instances):  
`db2set -l`
- Display all supported registry variables:  
`db2set -lr`
- Display all defined global variables:  
`db2set -g`
- Display all defined variables for the current instance:  
`db2set`
- Display all defined values for the current instance:  
`db2set -a11`
- Display all defined values for DB2COMM for the current instance:  
`db2set -a11 DB2COMM`
- Reset all defined variables for the instance INST on node 3:  
`db2set -r -i INST 3`
- Unset the variable DB2CHKPTR on the remote instance RMTINST through the DAS node RMTDAS using user ID MYID and password MYPASSWD:  
`db2set -i RMTINST -n RMTDAS -u MYID -p MYPASSWD DB2CHKPTR=`
- Set the variable DB2COMM to be TCPIP,IPXSPX,NETBIOS globally:

## db2set - DB2 Profile Registry Command

```
db2set -g DB2COMM=TCPIP,IPXSPX,NETBIOS
```

- Set the variable DB2COMM to be only TCPIP for instance MYINST:

```
db2set -i MYINST DB2COMM=TCPIP
```

- Set the variable DB2COMM to null at the given instance level:

```
db2set -null DB2COMM
```

### Usage Notes

If no variable name is specified, the values of all defined variables are displayed. If a variable name *is* specified, only the value of that variable is displayed. To delete a variable, specify *variable=*, followed by no value. To modify the value of a variable, specify *variable=*, followed by its new value. To set the value of a variable to NULL, specify *variable -null*. To display all the defined values of a variable, specify *variable -all*. To display all the defined variables in all registries, specify *-all*.

---

### db2split - Data Declustering Tool

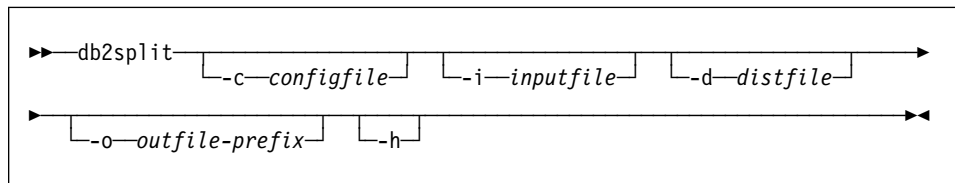
Partitions data across nodes in a multi-node environment.

For detailed information about **db2split**, see the *Administration Guide*.

#### Authorization

None

#### Command Syntax



#### Command Parameters

**-c** *configfile*

Name of the file containing configuration data required by the tool. The configuration file contains such information as the name of the input file, the position and length of the partitioning key, and the name of the log file. When using this option with a file name, the file name is assumed to contain all of the input information for the utility. The default value is `db2split.cfg`.

**-i** *inputfile*

Name of the file containing the data to be partitioned.

**-d** *distfile*

Name of the file containing an input partitioning map. If a customized partitioning map was created, it must be used.

**-o** *outfile-prefix*

The prefix of the output file. The utility appends a 5-character suffix (00000..00999) to the end of the prefix to generate the output file name.

**-h**

Display help information.

#### Usage Notes

The **-i**, **-d**, and **-o** options can be used to override the file names specified in the configuration file.

# db2sql92 - SQL92 Compliant SQL Statement Processor

---

## db2sql92 - SQL92 Compliant SQL Statement Processor

Reads SQL statements from either a flat file or standard input, dynamically describes and prepares the statements, and returns an answer set. Supports concurrent connections to multiple databases. This tool is provided in the `misc` subdirectory of the instance `sql1lib` directory.

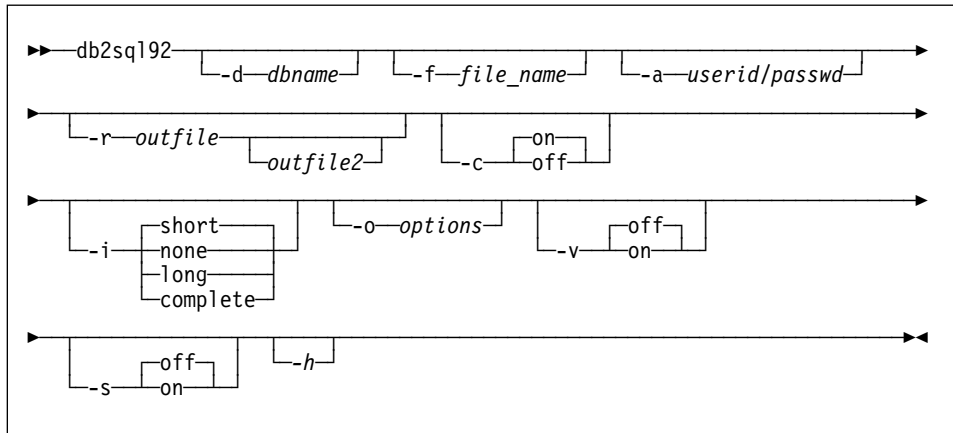
### Authorization

`sysadm`

### Required Connection

None. This command establishes a database connection.

### Command Syntax



### Command Parameters

`-d dbname`

An alias name for the database against which SQL statements are to be applied. The default is the value of the **DB2DBDFT** environment variable.

`-f file_name`

Name of an input file containing SQL statements. The default is standard input.

Identify comment text with two hyphens at the start of each line, that is, `--<comment>`. If it is to be included in the output, mark the comment as follows: `--#COMMENT <comment>`.

A *block* is a number of SQL statements that are treated as one, that is, information is collected for all of those statements at once, instead of one at a time. Identify the beginning of a block of queries as follows: `--#BGBLK`. Identify the end of a block of queries as follows: `--#E0BLK`.



## db2sql92 - SQL92 Compliant SQL Statement Processor

Specify one or more control options as follows: `--#SET <control option> <value>`. Valid control options are:

### *ROWS\_FETCH*

Number of rows to be fetched from the answer set. Valid values are -1 to *n*. The default value is -1 (all rows are to be fetched).

### *ROWS\_OUT*

Number of fetched rows to be sent to output. Valid values are -1 to *n*. The default value is -1 (all fetched rows are to be sent to output).

### *AUTOCOMMIT*

Specifies autocommit on or off. Valid values are ON or OFF. The default value is ON.

### *PAUSE*

Prompts the user to continue.

### *TIMESTAMP*

Generates a time stamp.

### **-a** *userid/passwd*

Name and password used to connect to the database.

### **-r** *outfile*

An output file that will contain the query results. An optional *outfile2* will contain a results summary. The default is standard output.

### **-c**

Automatically commit changes resulting from each SQL statement.

### **-i**

An elapsed time interval (in seconds).

#### *none*

Specifies that time information is not to be collected.

#### *short*

The run time for a query.

#### *long*

Elapsed time at the start of the next query.

#### *complete*

The time to prepare, execute, and fetch, expressed separately.

### **-o** *options*

Control options. Valid options are:

#### **f** *rows\_fetch*

Number of rows to be fetched from the answer set. Valid values are -1 to *n*. The default value is -1 (all rows are to be fetched).

#### **r** *rows\_out*

Number of fetched rows to be sent to output. Valid values are -1 to *n*. The default value is -1 (all fetched rows are to be sent to output).

### **-v**

Verbose. Send information to standard error during query processing. The default value is off.

## db2sql92 - SQL92 Compliant SQL Statement Processor

- s** Summary Table. Provide a summary of elapsed times and CPU times, containing both the arithmetic and the geometric means of all collected values.
- h** Display help information. When this option is specified, all other options are ignored, and only the help information is displayed.

### Usage Notes

The following can be executed from the **db2sql92** command prompt:

- All control options
- SQL statements
- CONNECT statements
- commit work
- help
- quit

This tool supports switching between different databases during a single execution of the program. To do this, issue a CONNECT RESET and then one of the following on the **db2sql92** command prompt (stdin):

```
connect to database  
connect to database USER userid USING passwd
```

SQL statements can be up to 32700 characters in length. Statements must be terminated by a semicolon.

SQL statements are executed with the repeatable read (RR) isolation level.

### See Also

“db2batch - Benchmark Tool” on page 7.

---

### db2start - Start DB2

Starts the current database manager instance background processes on a single node or on all the nodes defined in a multi-node environment. Start DB2 at the server before connecting to a database, precompiling an application, or binding a package to a database.

**db2start** can be executed as a system command or a CLP command. For a complete description of this command, see “START DATABASE MANAGER” on page 361.

## **db2stop - Stop DB2**

---

### **db2stop - Stop DB2**

Stops the current database manager instance.

**db2stop** can be executed as a system command or a CLP command. For a complete description of this command, see “STOP DATABASE MANAGER” on page 365.

---

### db2tbst - Get Tablespace State

Accepts a hexadecimal table space state value, and returns the state. The state value is part of the output from “LIST TABLESPACES” on page 258.

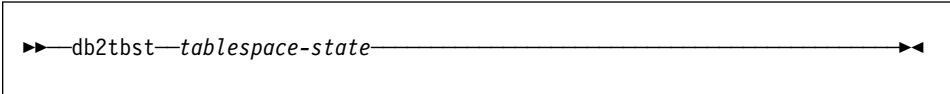
#### Authorization

None

#### Required Connection

None

#### Command Syntax



The diagram shows the command syntax for db2tbst. It consists of a rectangular box containing the text `db2tbst tablespace-state`. A horizontal line with arrowheads at both ends spans the width of the box, starting from the first space after `db2tbst` and ending at the space before `tablespace-state`. This indicates that `tablespace-state` is a required parameter.

#### Command Parameters

*tablespace-state*

A hexadecimal table space state value.

#### Example

The request `db2tbst 0x0000` produces the following output:

```
State = Normal
```

#### See Also

“LIST TABLESPACES” on page 258.

## db2trc - Trace

---

### db2trc - Trace

Activates DB2 trace facility tracing. DB2 uses its trace facility to trace events, dump trace data to a file, and format trace data into a readable form. Only use the trace facility when directed by DB2 Customer Service or by a technical support representative.

### Authorization

On UNIX based systems, one of the following:

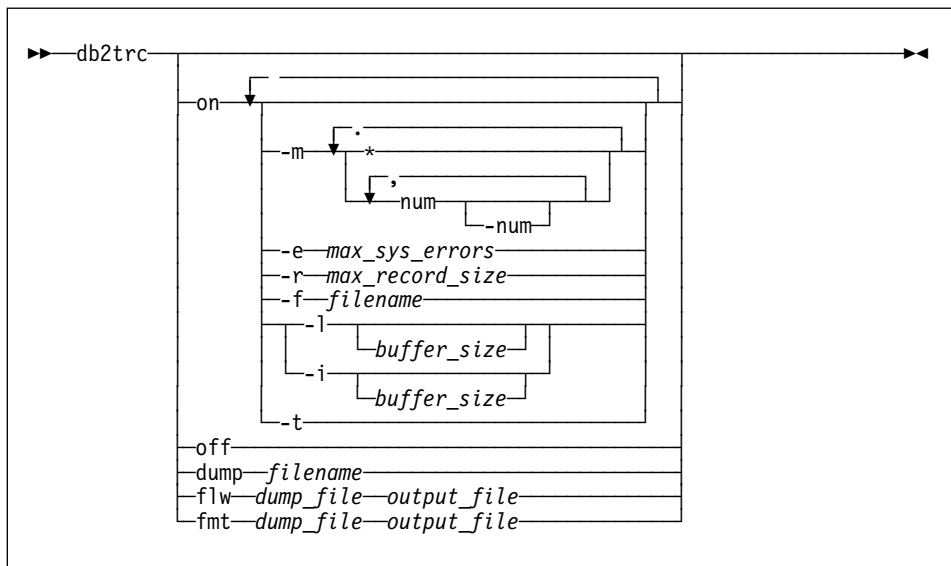
*sysadm*  
*sysctrl*  
*sysmaint*

On OS/2 or the Windows operating system, no authorization is required.

### Required Connection

None

### Command Syntax



### Command Parameters

*on*

Use this parameter to start the DB2 trace facility. See the next section for information about the options for this parameter.

### *dump*

After reproducing the error, dump the trace information to a file. The following command will put the information in the current directory in a file called `db2trc.dmp`:

```
db2trc dump db2trc.dmp
```

Specify a file name with this parameter. The file is saved in the current directory unless the path is explicitly specified.

### *off*

After the trace is dumped to a file, turn the trace off by typing:

```
db2trc off
```

### *flw | fmt*

After the trace is dumped to a binary file, confirm that it is taken by formatting it into an ASCII file. Use either the `flw` option (to sort by process or thread), or the `fmt` option (to list every event chronologically). For either option, specify the name of the dump file and the name of the output file that will be generated. For example:

```
db2trc flw db2trc.dmp db2trc.flw
```

## Starting a DB2 Trace

To use the trace facility, first turn it on using **db2trc on**. Unless instructed otherwise by DB2 Customer Service, use the command without options. The default trace option values are:

- `mask = *` (trace everything)
- `max_sys_errors = -1` (collect all)
- `max_record_size = 16` KB
- Trace destination = `-s` (shared memory)
- Records to retain = `-l` (last)
- `buffer_size = 64` KB if the trace destination is shared memory, or 1MB if the trace destination is a file.

If instructed to specify options to tailor the trace, use the following options as directed by DB2 Customer Service:

- m *mask* Specifies trace record types to focus the search. The *mask* variable consists of four one-byte masks separated by periods. The byte-masks act as a filter to accept or reject the trace record sent by DB2 for each event based on its ID. The four one-byte masks correspond to products, event types, components, and functions, respectively.
- e *max\_sys\_errors* Limits the number of DB2 internal system errors the trace will hold to *max\_sys\_errors*.
- r *max\_record\_size* Limits the size of trace records to *max\_record\_size* bytes. Longer trace records are truncated.
- f *filename* Specifies trace buffer location in shared memory or a file.

## db2trc - Trace

- l [ *buffer\_size*] | -i [ *buffer\_size*]  
'-l' ("l" as in "lucky") specifies that the last trace records are retained (that is, the first records are overwritten when the buffer is full). '-i' specifies that the initial trace records are retained (that is, no more records are written to the buffer once it is full). Use either of these options to specify the buffer size.
- t  
Includes time stamps. Applicable to UNIX environments only, where the logging of time stamps affects performance.

## Examples

For additional information about **db2trc**, including trace examples, see the *Troubleshooting Guide*.

## Usage Notes

The **db2trc** command must be issued several times to turn tracing on, produce a dump file, format the dump file, and turn tracing off again. The parameter list shows the order in which the parameters should be used.



## db2uiddl - Prepare Unique Index Conversion to V5 Semantics

---

### db2uiddl - Prepare Unique Index Conversion to V5 Semantics

Facilitates the management of a staged migration of unique indexes on a user's own schedule. Generates CREATE UNIQUE INDEX statements for unique indexes on user tables. For detailed information about using this tool, see one of the *Quick Beginnings* books.

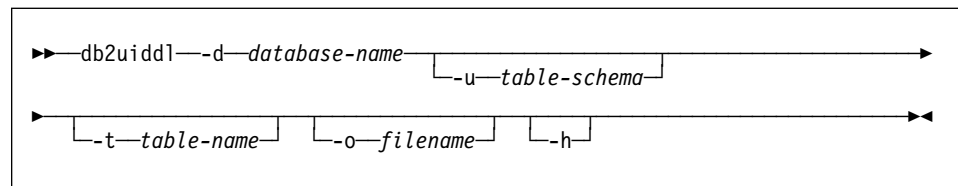
#### Authorization

*sysadm*

#### Required Connection

Database. This command automatically establishes a connection to the specified database.

#### Command Syntax



#### Command Parameters

**-d** *database-name*

The name of the database to be queried.

**-u** *table-schema*

Specifies the schema (creator user ID) of the tables that are to be processed. The default action is to process tables created by all user IDs.

**-t** *table-name*

The name of a table that is to be processed. The default action is to process all tables.

**-o** *filename*

The name of a file to which output is to be written. The default action is to write output to standard output.

**-h**

Display help information. When this option is specified, all other options are ignored, and only the help information is displayed.

#### Usage Notes

This tool can only be used on a database that has been migrated to DB2 Version 5.

**Note:** This tool was not designed to handle certain types of names. If a specific table name or table schema is a delimited identifier containing lower case characters, special characters, or blanks, it is preferable to request processing of *all* tables or schemas. The resulting output can be edited.

## db2untag - Release Container Tag

---

### db2untag - Release Container Tag

Removes the DB2 tag on a table space container. The tag is used to prevent DB2 from reusing a container in more than one table space. Displays information about the container tag, identifying the database with which the container is associated. Useful when it is necessary to release a container last used by a database that has since been deleted. If the tag is left behind, DB2 is prevented from using the resource in future.

**Attention:** This tool should only be used by informed system administrators.

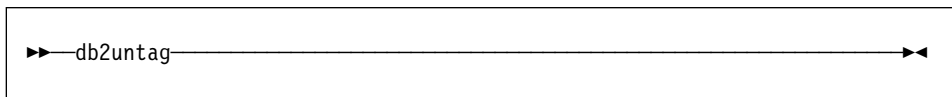
### Authorization

The user needs read/write access to the container for a table space that is owned by the ID that created the database.

### Required Connection

None

### Command Syntax



```
db2untag
```

### Command Parameters

None

### Usage Notes

An SQLCODE -294 (Container in Use error) is sometimes returned from create database or from create or alter table space operations, usually indicating a specification error on the operating system resource name when the container is already in use by another table space. A container can be used by only one table space at a time.

A system or database administrator who finds that the database which last used the container has been deleted, can use the **db2untag** tool if the container's tag was not removed. If the container is to be released, do one of the following:

- For SMS containers, remove the directory and its contents using the appropriate delete commands.
- For DMS containers, either delete the file or device, or let **db2untag** remove the container tag. The tool will leave such a DMS container otherwise unmodified.

**See Also**

“CREATE DATABASE” on page 143.

## db2vexp - Dynamic Visual Explain

---

### db2vexp - Dynamic Visual Explain

Explains a dynamic SQL statement, producing an access plan graph. Creates explain tables if they do not exist.

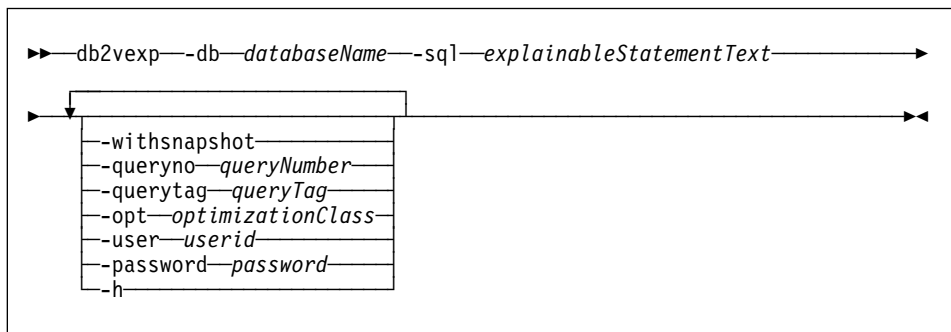
#### Authorization

The authorization rules are those defined for the SQL statement specified in the **db2vexp** command. For example, if a DELETE statement was used as the *explainableStatementText*, then the authorization rules for a DELETE statement apply. The current authorization ID must have INSERT privilege on the explain tables. If explain tables do not exist, the current authorization ID must also have CREATETAB privilege on the database. The explain tables will be created automatically.

#### Required Connection

None

#### Command Syntax



#### Command Parameters

- db** *databaseName*  
The name of the database where a connection will be made.
- sql** *explainableStatementText*  
The statement text to be dynamically explained, enclosed by double quotation marks.
- withsnapshot**  
Generates more explain details.
- queryno** *queryNumber*  
A query number to be associated with the dynamic explain.
- querytag** *queryTag*  
A query tag to be associated with the dynamic explain. It can be a maximum of twenty characters enclosed by double quotation marks.
- opt** *optimizationClass*  
Optimization class. Valid values are 0, 1, 3, 5, 7, and 9. See SET CURRENT QUERY OPTIMIZATION in the *SQL Reference*.

## db2vexp - Dynamic Visual Explain

**-user** *userid*

Specifies the user ID used to connect to the database.

**-password** *password*

Specifies the password used to connect to the database.

**-h**

Displays help information. When this option is specified, all other options are ignored and only the help is displayed. On Windows platforms, */?* can also be used to specify the help option.

### Usage Notes

A slash (*/*) can be used instead of a hyphen (-) to prefix a keyword.

## db2vexp - Dynamic Visual Explain

---

### Chapter 2. Command Line Processor (CLP)

This chapter explains how to invoke and use the command line processor, and describes CLP options.

---

#### Command Line Processor Invocation and Options

The **db2** command starts the command line processor. The CLP is used to execute database utilities, SQL statements and online help. It offers a variety of command options, and can be started in:

- Interactive input mode, characterized by the **db2 =>** input prompt
- Command mode, where each command must be prefixed by **db2**
- Batch mode, which uses the **-f** file input option.

**Note:** On Windows NT, “db2cmd - Open DB2 Command Window” on page 18 opens the CLP-enabled DB2 window, and initializes the DB2 command line environment. Issuing this command is equivalent to clicking on the *DB2 Command Window* icon.

“QUIT” on page 308 stops the command line processor. “TERMINATE” on page 368 also stops the command line processor, but removes the associated back-end process and frees any memory that is being used. TERMINATE is recommended if the database has been stopped, or if database configuration parameters have been changed.

**Note:** Existing connections should be reset before terminating the CLP.

The shell command (!), allows operating system commands to be executed from the interactive or the batch mode on UNIX based systems, and on OS/2 or the Windows operating system (!!s on UNIX, and !dir on OS/2 or the Windows operating system, for example).

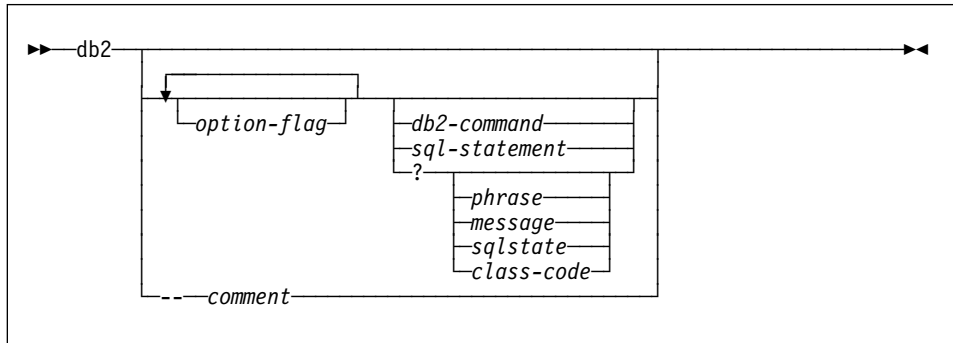
**Note:** Shell command support is not available on Windows 3.1.

#### Authorization

None

# Command Line Processor Invocation and Options

## Command Syntax



## Command Parameters

### *option-flag*

For a summary of valid CLP option flags, see Table 2 on page 71.

### *db2-command*

Specifies a DB2 command. For a description of DB2 commands, see Chapter 3, "CLP Commands" on page 83.

### *sql-statement*

Specifies an SQL statement.

?

Requests CLP general help.

### ? *phrase*

Requests the help text associated with a specified command or topic. If the database manager cannot find the requested information, it displays the general help screen.

? *options* requests a description and the current settings of the CLP options. ? *help* requests information about reading the online help syntax diagrams.

### ? *message*

Requests help for a message specified by a valid SQLCODE (? sql10007n, for example).

### ? *sqlstate*

Requests help for a message specified by a valid SQLSTATE.

### ? *class-code*

Requests help for a message specified by a valid class-code.

### -- *comment*

Input that begins with the comment characters -- is treated as a comment by the command line processor.

**Note:** In each case, a blank space must separate the question mark (?) from the variable name.



# Command Line Processor Invocation and Options

## Options

The CLP command options can be specified by setting the command line processor **DB2OPTIONS** environment variable (which must be in uppercase), or with command line flags.

Users can set options for an entire session using **DB2OPTIONS**.

View the current settings for the option flags and the value of **DB2OPTIONS** using “LIST COMMAND OPTIONS” on page 229. Change an option setting from the interactive input mode or a command file using “UPDATE COMMAND OPTIONS” on page 377.

The command line processor sets options in the following order:

1. Sets up default options.
2. Reads **DB2OPTIONS** to override the defaults.
3. Reads the command line to override **DB2OPTIONS**.
4. Accepts input from UPDATE COMMAND OPTIONS as a final interactive override.

Table 2 summarizes the CLP option flags. These options can be specified in almost any sequence and combination. To turn an option on, prefix the corresponding option letter with a minus sign (-). To turn an option off, either prefix the option letter with a minus sign and follow the option letter with another minus sign, or prefix the option letter with a plus sign (+). For example, -c turns the auto-commit option on, and either -c- or +c turns it off. These option letters are not case sensitive, that is, -a and -A are equivalent.

Option Flag	Description	Default Setting
-a	This option tells the command line processor to display SQLCA data.	OFF
-c	This option tells the command line processor to automatically commit SQL statements.	ON
-e{c s}	This option tells the command line processor to display SQLCODE or SQLSTATE. These options are mutually exclusive.	OFF
-filename	This option tells the command line processor to read command input from a file instead of from standard input.	OFF
-lfilename	This option tells the command line processor to log commands in a history file.	OFF
-o	This option tells the command line processor to display output data and messages to standard output.	ON
-p	This option tells the command line processor to display a command line processor prompt when in interactive input mode.	ON

## Command Line Processor Invocation and Options

Option Flag	Description	Default Setting
-rfilename	This option tells the command line processor to write the report generated by a command to a file.	OFF
-s	This option tells the command line processor to stop execution if errors occur while executing commands in a batch file or in interactive mode.	OFF
-t	This option tells the command line processor to use a semicolon (;) as the statement termination character.	OFF
-tdx	This option tells the command line processor to define and to use x as the statement termination character.	OFF
-v	This option tells the command line processor to echo command text to standard output.	OFF
-w	This option tells the command line processor to display SQL statement warning messages.	ON
-zfilename	This option tells the command line processor to redirect all output to a file. It is similar to the -r option, but includes any messages or error codes with the output.	OFF

### Example

The AIX command:

```
export DB2OPTIONS='+a -c +ec -o -p'
```

sets the following default settings for the session:

```
Display SQLCA - off
Auto Commit - on
Display SQLCODE - off
Display Output - on
Display Prompt - on
```

The following is a detailed description of these options:

#### Show SQLCA Data Option (-a):

Displays SQLCA data to standard output after executing a DB2 command or an SQL statement. The SQLCA data is displayed instead of an error or success message.

The default setting for this command option is OFF (+a or -a-).

The -o and the -r options affect the -a option; see the option descriptions for details.

#### Auto-commit Option (-c):

This option specifies whether each command or statement is to be treated independently. If set ON (-c), each command or statement is automatically

## Command Line Processor Invocation and Options

committed or rolled back. If the command or statement is successful, it and all successful commands and statements that were issued before it with autocommit OFF (+c or -c-) are committed. If, however, the command or statement fails, it and all successful commands and statements that were issued before it with autocommit OFF are rolled back. If set OFF (+c or -c-), COMMIT or ROLLBACK must be issued explicitly, or one of these actions will occur when the next command with autocommit ON (-c) is issued.

The default setting for this command option is ON.

The auto-commit option does not affect any other command line processor option.

**Example:** Consider the following scenario:

1. db2 create database test
2. db2 connect to test
3. db2 +c "create table a (c1 int)"
4. db2 select c2 from a

The SQL statement in step 4 fails because there is no column named C2 in table A. Since that statement was issued with auto-commit ON (default), it rolls back not only the statement in step 4, but also the one in step 3, because the latter was issued with auto-commit OFF. The command:

```
db2 list tables
```

then returns an empty list.

### Display SQLCODE/SQLSTATE Option (-e):

The -e{c|s} option tells the command line processor to display the SQLCODE (-ec) or the SQLSTATE (-es) to standard output.

The default setting for this command option is OFF (+e or -e-).

The -o and the -r options affect the -e option; see the option descriptions for details.

The display SQLCODE/SQLSTATE option does not affect any other command line processor option.

**Example:** To retrieve SQLCODE from the command line processor running on AIX, enter:

```
sqlcode='db2 -ec +o db2-command'
```

### Read from Input File Option (-f):

The -f*filename* option tells the command line processor to read input from a specified file, instead of from standard input. *Filename* is an absolute or relative file name which may include the directory path to the file. If the directory path is not specified, the current directory is used.

When other options are combined with option -f, option -f must be specified last. For example:

```
db2 -tvf filename
```

**Note:** This option cannot be changed from within the interactive mode.

## Command Line Processor Invocation and Options

The default setting for this command option is OFF (+f or -f-).

Commands are processed until “QUIT” on page 308 or “TERMINATE” on page 368 is issued, or an end-of-file is encountered.

If both this option and a database command are specified, the command line processor does not process any commands, and an error message is returned.

Input file lines which begin with the comment characters -- are treated as comments by the command line processor. Comment characters must be the first non-blank characters on a line.

If the -f*filename* option is specified, the -p option is ignored.

The read from input file option does not affect any other command line processor option.

### Log Commands in History File Option (-l):

The -l*filename* option tells the command line processor to log commands to a specified file. This history file contains records of the commands executed and their completion status. *Filename* is an absolute or relative file name which may include the directory path to the file. If the directory path is not specified, the current directory is used. If the specified file or default file already exists, the new log entry is appended to that file.

When other options are combined with option -l, option -l must be specified last. For example:

```
db2 -tv1 filename
```

The default setting for this command option is OFF (+l or -l-).

The log commands in history file option does not affect any other command line processor option.

### Display Output Option (-o):

The -o option tells the command line processor to send output data and messages to standard output.

The default setting for this command option is ON.

The interactive mode start-up information is not affected by this option. Output data consists of report output from the execution of the user-specified command, and SQLCA data (if requested).

The following options may be affected by the +o option:

- -r*filename*: Interactive mode start-up information is not saved.
- -e: SQLCODE or SQLSTATE is displayed on standard output even if +o is specified.
- -a: No effect if +o is specified. If -a, +o and -r*filename* are specified, SQLCA information is written to a file.

If both -o and -e options are specified, the data and either the SQLCODE or the SQLSTATE are displayed on the screen.

## Command Line Processor Invocation and Options

If both `-o` and `-v` options are specified, the data is displayed, and the text of each command issued is echoed to the screen.

The display output option does not affect any other command line processor option.

### Display DB2 Interactive Prompt Option (`-p`):

The `-p` option tells the command line processor to display the command line processor prompt when the user is in interactive mode.

The default setting for this command option is ON.

Turning the prompt off is useful when commands are being piped to the command line processor. For example, a file containing CLP commands could be executed by issuing:

```
db2 +p < myfile.clp
```

The `-p` option is ignored if the `-filename` option is specified.

The display DB2 interactive prompt option does not affect any other command line processor option.

### Save to Report File Option (`-r`):

The `-filename` option causes any output data generated by a command to be written to a specified file, and is useful for capturing a report that would otherwise scroll off the screen. Messages or error codes are not written to the file. *Filename* is an absolute or relative file name which may include the directory path to the file. If the directory path is not specified, the current directory is used. New report entries are appended to the file.

The default setting for this command option is OFF (`+r` or `-r-`).

If the `-a` option is specified, SQLCA data is written to the file.

The `-r` option does not affect the `-e` option. If the `-e` option is specified, SQLCODE or SQLSTATE is written to standard output, not to a file.

If `-filename` is set in **DB2OPTIONS**, the user can set the `+r` (or `-r-`) option from the command line to prevent output data for a particular command invocation from being written to the file.

The save to report file option does not affect any other command line processor option.

### Stop Execution on Command Error Option (`-s`):

When commands are issued in interactive mode, or from an input file, and syntax or command errors occur, the `-s` option causes the command line processor to stop execution and to write error messages to standard output.

The default setting for this command option is OFF (`+s` or `-s-`). This setting causes the command line processor to display error messages, continue execution of the remaining commands, and to stop execution only if a system error occurs (return code 8).

## Command Line Processor Invocation and Options

The following table summarizes this behavior:

Return Code	-s Option Set	+s Option Set
0 (success)	execution continues	execution continues
1 (0 rows selected)	execution continues	execution continues
2 (warning)	execution continues	execution continues
4 (DB2 or SQL error)	execution stops	execution continues
8 (System error)	execution stops	execution stops

### Statement Termination Character Option (-t):

The `-t` option tells the command line processor to use a semicolon (;) as the statement termination character, and disables the backslash (\) line continuation character.

**Note:** This option cannot be changed from within the interactive mode.

The default setting for this command option is OFF (+t or -t-).

To define a termination character, use `-td` followed by the chosen termination character. For example, `-tdx` sets `x` as the statement termination character.

The termination character cannot be used to concatenate multiple statements from the command line, since only the last non-blank character on each input line is checked for a termination symbol.

The statement termination character option does not affect any other command line processor option.

### Verbose Output Option (-v):

The `-v` option causes the command line processor to echo (to standard output) the command text entered by the user prior to displaying the output, and any messages from that command. "ECHO" on page 160 is exempt from this option.

The default setting for this command option is OFF (+v or -v-).

The `-v` option has no effect if `+o` (or `-o-`) is specified.

The verbose output option does not affect any other command line processor option.

### Show Warning Messages Option (-w):

The `-w` option tells the command line processor to show SQL statement warning messages.

The default setting for this command option is ON.

### Save all Output to File Option (-z):

The `-zfilename` option causes all output generated by a command to be written to a specified file, and is useful for capturing a report that would otherwise scroll off the screen. It is similar to the `-r` option; in this case, however, messages, error codes, and other informational output are also

## Command Line Processor Invocation and Options

written to the file. *Filename* is an absolute or relative file name which may include the directory path to the file. If the directory path is not specified, the current directory is used. New report entries are appended to the file.

The default setting for this command option is OFF (+z or -z-).

If the -a option is specified, SQLCA data is written to the file.

The -z option does not affect the -e option. If the -e option is specified, SQLCODE or SQLSTATE is written to standard output, not to a file.

If -z*filename* is set in **DB2OPTIONS**, the user can set the +z (or -z-) option from the command line to prevent output data for a particular command invocation from being written to the file.

The save all output to file option does not affect any other command line processor option.

### Return Codes

When the command line processor finishes processing a command or an SQL statement, it returns an exit (or return) code. These codes are transparent to users executing CLP functions from the command line, but they can be retrieved when those functions are executed from a shell script.

For example, the following Bourne shell script executes the GET DATABASE MANAGER CONFIGURATION command, then inspects the CLP return code:

```
db2 get database manager configuration
if ["$?" = "0"]
then echo "OK!"
fi
```

The return code can be one of the following:

<b>Code</b>	<b>Description</b>
<b>0</b>	DB2 command or SQL statement executed successfully
<b>1</b>	SELECT or FETCH statement returned no rows
<b>2</b>	DB2 command or SQL statement warning
<b>4</b>	DB2 command or SQL statement error
<b>8</b>	Command line processor system error

The command line processor does not provide a return code while a user is executing statements from interactive mode, or while input is being read from a file (using the -f option).

A return code is available only after the user quits interactive mode, or when processing of an input file ends. In these cases, the return code is the logical OR of the distinct codes returned from the individual commands or statements executed to that point.

For example, if a user in interactive mode issues commands resulting in return codes of 0, 1, and 2, a return code of 3 will be returned after the user quits interactive mode. The individual codes 0, 1, and 2 are not returned. Return code 3 tells the user that

## Using the Command Line Processor

during interactive mode processing, one or more commands returned a 1, and one or more commands returned a 2.

A return code of 4 results from a negative SQLCODE returned by a DB2 command or an SQL statement. A return code of 8 results only if the command line processor encounters a system error.

If commands are issued from an input file or in interactive mode, and the command line processor experiences a system error (return code 8), command execution is halted immediately. If one or more DB2 commands or SQL statements end in error (return code 4), command execution stops if the `-s` (Stop Execution on Command Error) option is set; otherwise, execution continues.

---

## Using the Command Line Processor

The command line processor operates as follows:

- The CLP command (in either case) is typed at the command prompt.
- The command is sent to the command shell by pressing the ENTER key.
- Output is automatically directed to the standard output device.
- Piping and redirection are supported.
- The user is notified of successful and unsuccessful completion.
- Following execution of the command, control returns to the operating system command prompt, and the user may enter more commands.

Before accessing a database, the user must perform preliminary tasks, such as starting DB2 with “START DATABASE MANAGER” on page 361. The user must also connect to a database before it can be queried. Connect to a database by doing one of the following:

- Issue the SQL `CONNECT TO database` statement
- Establish an implicit connection to the default database defined by the environment variable **DB2DBDFT**.

For information about working with tables within a database, see the *SQL Reference*.

If a command exceeds the character limit allowed at the command prompt, a backslash (`\`) can be used as the line continuation character. When the command line processor encounters the line continuation character, it reads the next line and concatenates the characters contained on both lines. Alternatively, the `-t` option can be used to set a line termination character (see page 76). In this case, the line continuation character is invalid, and all statements and commands must end with the line termination character.

The command line processor recognizes a string called NULL as a null string. Fields that have been set previously to some value can later be set to null. For example,

```
db2 update database manager configuration using tm_database NULL
```



## Using the Command Line Processor

sets the *tm\_database* field to null. This operation is case sensitive. A lowercase null is not interpreted as a null string, but rather as a string containing the letters null.

### Using the Command Line Processor in Command Files

CLP requests to the database manager can be imbedded in a shell script command file. The following example shows how to enter the CREATE TABLE statement in a shell script command file:

```
db2 "create table mytable (name VARCHAR(20), color CHAR(10))"
```

For more information about commands and command files, see the appropriate operating system manual.

### Command Line Processor Design

The command line processor consists of two processes: the front-end process (the DB2 command), which acts as the user interface, and the back-end process (db2bp), which maintains a database connection.

#### Maintaining Database Connections

Each time that **db2** is invoked, a new front-end process is started. The back-end process is started by the first **db2** invocation, and can be explicitly terminated with "TERMINATE" on page 368. All front-end processes with the same parent are serviced by a single back-end process, and therefore share a single database connection.

For example, the following **db2** calls from the same operating system command prompt result in separate front-end processes sharing a single back-end process, which holds a database connection throughout:

```
db2 'connect to sample',  
db2 'select * from org',  
. foo (where foo is a shell script containing DB2 commands), and  
db2 -tf myfile.clp.
```

The following invocations from the same operating system prompt result in separate database connections because each has a distinct parent process, and therefore a distinct back-end process:

```
foo  
. foo &  
foo &  
sh foo
```

#### Communication between Front-end and Back-end Processes

The front-end process and back-end processes communicate through three message queues: a request queue, an input queue, and an output queue.

## Using the Command Line Processor

### Environment Variables

The following environment variables offer a means of configuring communication between the two processes:

Variable	Minimum	Maximum	Default
DB2BQTIME	1 second	5294967295	1 second
DB2BQTRY	0 tries	5294967295	60 tries
DB2RQTIME	1 second	5294967295	5 seconds
DB2IQTIME	1 second	5294967295	5 seconds

**DB2BQTIME** When the command line processor is invoked, the front-end process checks if the back-end process is already active. If it is active, the front-end process re-establishes a connection to it. If it is not active, the front-end process activates it. The front-end process then idles for the duration specified by the **DB2BQTIME** variable, and checks again. The front-end process continues to check for the number of times specified by the **DB2BQTRY** variable, after which, if the back-end process is still not active, it times out and returns an error message.

**DB2BQTRY** works in conjunction with the **DB2BQTIME** variable, and specifies the number of times the front-end process tries to determine whether the back-end process is active.

The values of **DB2BQTIME** and **DB2BQTRY** can be increased during peak periods to optimize query time.

**DB2RQTIME** Once the back-end process has been started, it waits on its request queue for a request from the front-end. It also waits on the request queue between requests initiated from the command prompt.

The **DB2RQTIME** variable specifies the length of time the back-end process waits for a request from the front-end process. At the end of this time, if no request is present on the request queue, the back-end process checks whether the parent of the front-end process still exists, and terminates itself if it does not exist. Otherwise, it continues to wait on the request queue.

**DB2IQTIME** When the back-end process receives a request from the front-end process, it sends an acknowledgement to the front-end process indicating that it is ready to receive input via the input queue. The back-end process then waits on its input queue. It also waits on the input queue while a batch file (specified with the `-f` option) is executing, and while the user is in interactive mode.

The **DB2IQTIME** variable specifies the length of time the back-end process waits on the input queue for the front-end process to pass the commands. After this time has elapsed, the back-end process checks whether the front-end process is active, and returns to wait on the request queue if the

## Using the Command Line Processor

front-end process no longer exists. Otherwise, the back-end process continues to wait for input from the front-end process.

To view the values of these environment variables, use “LIST COMMAND OPTIONS” on page 229.

The back-end environment variables inherit the values set by the front-end process at the time the back-end process is initiated. However, if the front-end environment variables are changed, the back-end process will not inherit these changes. The back-end process must first be terminated, and then restarted (by issuing the **db2** command) to inherit the changed values.

An example of when the back-end process must be terminated is provided by the following scenario:

1. User A logs on, issues some CLP commands, and then logs off without issuing TERMINATE.
2. User B logs on using the same window.
3. When user B issues certain CLP commands, they fail with message DB21016 (system error).

The back-end process started by user A is still active when user B starts using the CLP, because the parent of user B's front-end process (the operating system window from which the commands are issued) is still active. The back-end process attempts to service the new commands issued by user B; however, user B's front-end process does not have enough authority to use the message queues of the back-end process, because it needs the authority of user A, who created that back-end process. A CLP session must end with a TERMINATE command before a user starts a new CLP session using the same operating system window. This creates a fresh back-end process for each new user, preventing authority problems, and setting the correct values of environment variables (such as **DB2INSTANCE**) in the new user's back-end process.

### Usage Notes

Commands can be entered either in uppercase or in lowercase from the command prompt. However, parameters that are case sensitive to DB2 must be entered in the exact case desired. For example, the *comment-string* in the WITH clause of the CHANGE DATABASE COMMENT command is a case sensitive parameter.

Delimited identifiers are allowed in SQL statements. For more detailed information on the use of delimited identifiers in SQL statements, see the *SQL Reference*.

Special characters, or metacharacters (such as \$ & \* ( ) ; < > ? \ ' ") are allowed within CLP commands. If they are used outside the CLP interactive mode, or the CLP batch input mode, these characters are interpreted by the operating system shell. Quotation marks or an escape character are required if the shell is not to take any special action.

For example, when executed inside an AIX Korn shell environment,

```
db2 select * from org where division > 'Eastern'
```

## Using the Command Line Processor

is interpreted as "select <the names of all files> from org where division". The result, an SQL syntax error, is redirected to the file Eastern. The following syntax produces the correct output:

```
db2 "select * from org where division > 'Eastern'"
```

Special characters vary from platform to platform. In the AIX Korn shell, the above example could be rewritten using an escape character (\), such as \\*, \>, or \'. In the OS/2 shell, \\* or \' results in a syntax error.

Most operating system environments allow input and output to be redirected. For example, if a connection to the SAMPLE database has been made, the following request queries the STAFF table, and sends the output to a file named staflist.txt in the mydata directory:

```
db2 "select * from staff" > mydata/staflist.txt
```

For environments such as Windows 3.1, where output redirection is not supported, CLP options can be used. For example, the request can be rewritten as

```
db2 -r mydata\staflist.txt "select * from staff"
```

```
db2 -z mydata\staflist.txt "select * from staff"
```

For more information on CLP options for Windows, see the *Installing and Using DB2 Clients for Windows* book.

The command line processor is not a programming language. For example, it does not support host variables, and the statement,

```
db2 connect to :HostVar in share mode
```

is syntactically incorrect, because :HostVar is not a valid database name.

The command line processor represents SQL NULL values as hyphens (-). If the column is numeric, the hyphen is placed at the right of the column. If the column is not numeric, the hyphen is at the left. For information about using the command line processor with DB2 Connect and host databases, see the *DB2 Connect User's Guide*.

## Chapter 3. CLP Commands

This chapter describes the DB2 commands in alphabetical order. These commands are used to control the system interactively.

**Note:** Slashes (/) in directory paths are specific to UNIX based systems, and are equivalent to back slashes (\) in directory paths on OS/2 and Windows operating systems.

For information about how the command descriptions are organized, see “How the Command Descriptions are Organized” on page 1.

---

### DB2 CLP Commands

The following table lists the CLP commands grouped by functional category:

<i>Table 5 (Page 1 of 4). DB2 CLP Commands</i>
<b>CLP Session Control</b>
“LIST COMMAND OPTIONS” on page 229
“UPDATE COMMAND OPTIONS” on page 377
“CHANGE ISOLATION LEVEL” on page 141
“SET RUNTIME DEGREE” on page 356
“TERMINATE” on page 368
“QUIT” on page 308
<b>Database Manager Control</b>
“START DATABASE MANAGER” on page 361
“STOP DATABASE MANAGER” on page 365
“GET DATABASE MANAGER CONFIGURATION” on page 184
“RESET DATABASE MANAGER CONFIGURATION” on page 331
“UPDATE DATABASE MANAGER CONFIGURATION” on page 381
“GET ADMIN CONFIGURATION” on page 169
“RESET ADMIN CONFIGURATION” on page 327
“UPDATE ADMIN CONFIGURATION” on page 375
<b>Database Control</b>
“RESTART DATABASE” on page 335
“CREATE DATABASE” on page 143
“DROP DATABASE” on page 157
“MIGRATE DATABASE” on page 282
“LIST INDOUBT TRANSACTIONS” on page 241
“ACTIVATE DATABASE” on page 87

## DB2 CLP Commands

<i>Table 5 (Page 2 of 4). DB2 CLP Commands</i>
"DEACTIVATE DATABASE" on page 149
"LIST DRDA INDOUBT TRANSACTIONS" on page 239
<b>Database Directory Management</b>
"CATALOG DATABASE" on page 118
"UNCATALOG DATABASE" on page 369
"CATALOG DCS DATABASE" on page 121
"UNCATALOG DCS DATABASE" on page 371
"CHANGE DATABASE COMMENT" on page 139
"LIST DATABASE DIRECTORY" on page 231
"LIST DCS DIRECTORY" on page 237
"CATALOG GLOBAL DATABASE" on page 124
"CATALOG ODBC DATA SOURCE" on page 135
"LIST ODBC DATA SOURCES" on page 251
"UNCATALOG ODBC DATA SOURCE" on page 374
<b>Client/Server Directory Management</b>
"CATALOG LOCAL NODE" on page 129
"CATALOG NAMED PIPE NODE" on page 131
"CATALOG APPC NODE" on page 111
"CATALOG APPCLU NODE" on page 114
"CATALOG APPN NODE" on page 116
"CATALOG IPX/SPX NODE" on page 126
"CATALOG NETBIOS NODE" on page 133
"CATALOG TCP/IP NODE" on page 136
"UNCATALOG NODE" on page 372
"LIST NODE DIRECTORY" on page 245
<b>Network Support</b>
"REGISTER" on page 315
"DEREGISTER" on page 151
<b>Database Configuration</b>
"GET DATABASE CONFIGURATION" on page 175
"RESET DATABASE CONFIGURATION" on page 329
"UPDATE DATABASE CONFIGURATION" on page 379
<b>Backup/Recovery</b>
"BACKUP DATABASE" on page 93
"RESTORE DATABASE" on page 337
"ROLLFORWARD DATABASE" on page 344

<i>Table 5 (Page 3 of 4). DB2 CLP Commands</i>
"LIST BACKUP/HISTORY" on page 227
"PRUNE HISTORY" on page 303
"UPDATE RECOVERY HISTORY FILE" on page 385
<b>Operational Utilities</b>
"FORCE APPLICATION" on page 167
"LIST PACKAGES/TABLES" on page 253
"REORGCHK" on page 320
"REORGANIZE TABLE" on page 317
"RUNSTATS" on page 350
<b>Database Monitoring</b>
"GET MONITOR SWITCHES" on page 197
"UPDATE MONITOR SWITCHES" on page 383
"GET DATABASE MANAGER MONITOR SWITCHES" on page 194
"GET SNAPSHOT" on page 199
"RESET MONITOR" on page 333
"LIST ACTIVE DATABASES" on page 224
"LIST APPLICATIONS" on page 225
"LIST DCS APPLICATIONS" on page 235
<b>Data Utilities</b>
"EXPORT" on page 161
"IMPORT" on page 212
"LOAD" on page 262
"LOAD QUERY" on page 280
<b>Application Preparation</b>
"PRECOMPILE PROGRAM" on page 284
"BIND" on page 98
"REBIND" on page 309
<b>Remote Server Utilities</b>
"ATTACH" on page 91
"DETACH" on page 156
<b>Table Space Management</b>
"LIST TABLESPACE CONTAINERS" on page 256
"SET TABLESPACE CONTAINERS" on page 358
"LIST TABLESPACES" on page 258
"QUIESCE TABLESPACES FOR TABLE" on page 305
<b>Node Management</b>

## DB2 CLP Commands

<i>Table 5 (Page 4 of 4). DB2 CLP Commands</i>
"ADD NODE" on page 89
"DROP NODE VERIFY" on page 159
"LIST NODES" on page 250
<b>Nodegroup Management</b>
"LIST NODEGROUPS" on page 248
"REDISTRIBUTE NODEGROUP" on page 312
<b>Additional Commands</b>
"DESCRIBE" on page 152
"ECHO" on page 160
"GET AUTHORIZATIONS" on page 172
"GET CONNECTION STATE" on page 174
"GET INSTANCE" on page 196
"HELP" on page 211
"INVOKE STORED PROCEDURE" on page 222
"QUERY CLIENT" on page 304
"SET CLIENT" on page 353
"INITIALIZE TAPE" on page 221
"REWIND TAPE" on page 343
"SET TAPE POSITION" on page 360



## ACTIVATE DATABASE

Activates the specified database and starts up all necessary database services, so that the database is available for connection and use by any application.

### Scope

This command activates the specified database on all nodes within the system. If one or more of these nodes encounters an error during activation of the database, a warning is returned. The database remains activated on all nodes on which the command has succeeded.

### Authorization

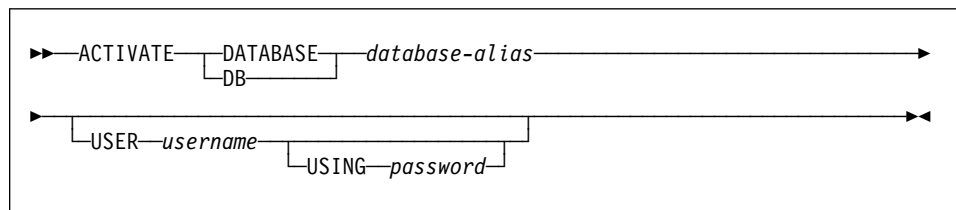
One of the following:

*sysadm*  
*sysctrl*  
*sysmaint*

### Required Connection

None

### Command Syntax



### Command Parameters

*database-alias*

Specifies the alias of the database to be started.

**USER** *username*

Specifies the user starting the database.

**USING** *password*

Specifies the password for the user name.

### Usage Notes

If a database has not been started, and a **CONNECT TO** (or an implicit connect) is issued in an application, the application must wait while the database manager starts the required database, before it can do any work with that database. However, once the database is started, other applications can simply connect and use it without spending time on its start up.

## ACTIVATE DATABASE

Database administrators can use ACTIVATE DATABASE to start up selected databases. This eliminates any application time spent on database initialization.

Databases initialized by ACTIVATE DATABASE can be shut down by “DEACTIVATE DATABASE” on page 149, or by “STOP DATABASE MANAGER” on page 365.

If a database was started by a CONNECT TO (or an implicit connect) and subsequently an ACTIVATE DATABASE is issued for that same database, then DEACTIVATE DATABASE must be used to shut down that database. If ACTIVATE DATABASE was not used to start the database, the database will shut down when the last application disconnects.

ACTIVATE DATABASE behaves in a similar manner to a CONNECT TO (or an implicit connect) when working with a database requiring a restart (for example, database in an inconsistent state). The database will be restarted before it can be initialized by ACTIVATE DATABASE. Restart will only be performed if the database is configured to have AUTORESTART ON.

**Note:** The application issuing the ACTIVATE DATABASE command cannot have an active database connection to any database.

### See Also

“DEACTIVATE DATABASE” on page 149

“STOP DATABASE MANAGER” on page 365.

## ADD NODE

Adds a new node to the parallel database system. This command creates database partitions for all databases currently defined in the MPP server on the new node. The user can specify the source node for any temporary table spaces to be created with the databases, or specify that no temporary table spaces are to be created. The command must be issued from the node that is being added, and can only be issued on an MPP server.

### Scope

This command only affects the node on which it is executed.

### Authorization

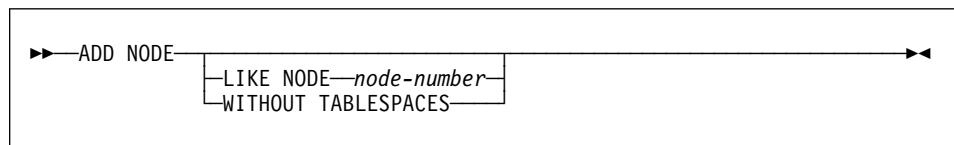
One of the following:

*sysadm*  
*sysctrl*

### Required Connection

None

### Command Syntax



### Command Parameters

**LIKE NODE** *node-number*

Specifies that the containers for the temporary table spaces will be the same as the containers on the specified *node-number* for each database in the instance. The node specified must be a node that is already in the `db2nodes.cfg` file.

**WITHOUT TABLESPACES**

Specifies that containers for the temporary table spaces are not created for any of the databases. The `ALTER TABLESPACE` statement must be used to add temporary table space containers to each database before the database can be used.

**Note:** If no option is specified, containers for the temporary table spaces will be the same as the containers on the catalog node for each database. The catalog node may be a different node for each database in the MPP system.

## ADD NODE

### Usage Notes

Before adding a new node, ensure that there is sufficient storage for the containers that must be created for all existing databases on the system.

The add node operation creates an empty database partition on the new node for every database that exists in the instance. The configuration parameters for the new database partitions are set to the default value.

If an add node operation fails while creating a database partition locally, it enters a clean-up phase, in which it locally drops all databases that have been created. This means that the database partitions are removed only from the node being added (that is, the local node). Existing database partitions remain unaffected on all other nodes. If this fails, no further clean up is done, and an error is returned.

The database partitions on the new node cannot be used to contain user data until after the ALTER NODEGROUP statement has been used to add the node to a nodegroup. For details, see the *SQL Reference*.

This command will fail if a create database or a drop database operation is in progress. The command can be reissued once the operation has completed.

If temporary table spaces are to be created with the database partitions, ADD NODE may have to communicate with another node in the MPP system in order to retrieve the table space definitions. The *start\_stop\_time* database manager configuration parameter is used to specify the time, in minutes, by which the other node must respond with the table space definitions. If this time is exceeded, the command fails. Increase the value of *start\_stop\_time*, and reissue the command.

### See Also

“START DATABASE MANAGER” on page 361.

---

**ATTACH**

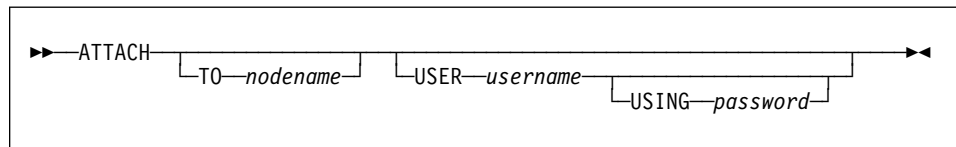
Enables an application to specify the instance at which instance-level commands (CREATE DATABASE and FORCE APPLICATION, for example) are to be executed. This instance may be the current instance, another instance on the same workstation, or an instance on a remote workstation.

**Authorization**

None

**Required Connection**

None. This command establishes an instance attachment.

**Command Syntax****Command Parameters**

**TO** *nodename*

Alias of the instance to which the user wants to attach. This instance must have a matching entry in the local node directory. The only exception to this is the local instance (as specified by the **DB2INSTANCE** environment variable) which may be specified as the object of an attach, but which cannot be used as a node name in the node directory.

**USER** *username*

Specifies the authentication identifier.

**USING** *password*

Specifies the password for the user name.

**Example**

Catalog two remote nodes:

```

db2 catalog tcpip node node1 remote freedom server server1
db2 catalog tcpip node node2 remote flash server server1

```

Attach to the first node, force all users, and then detach:

```

db2 attach to node1
db2 force application all
db2 detach

```

## ATTACH

Attach to the second node, and see who is on:

```
db2 attach to node2
db2 list applications
```

After the command returns agent IDs 1, 2 and 3, force 1 and 3, and then detach:

```
db2 force application (1, 3)
db2 detach
```

Attach to the current instance (not necessary, will be implicit), force all users, then detach (AIX only):

```
db2 attach to $DB2INSTANCE
db2 force application all
db2 detach
```

### Usage Notes

If *nodename* is omitted from the command, information about the current state of attachment is returned.

If ATTACH has not been executed, instance-level commands are executed against the current instance, specified by the **DB2INSTANCE** environment variable.

### See Also

“DETACH” on page 156.

## BACKUP DATABASE

Creates a backup copy of a database or a table space.

### Scope

This command only affects the node on which it is executed.

### Authorization

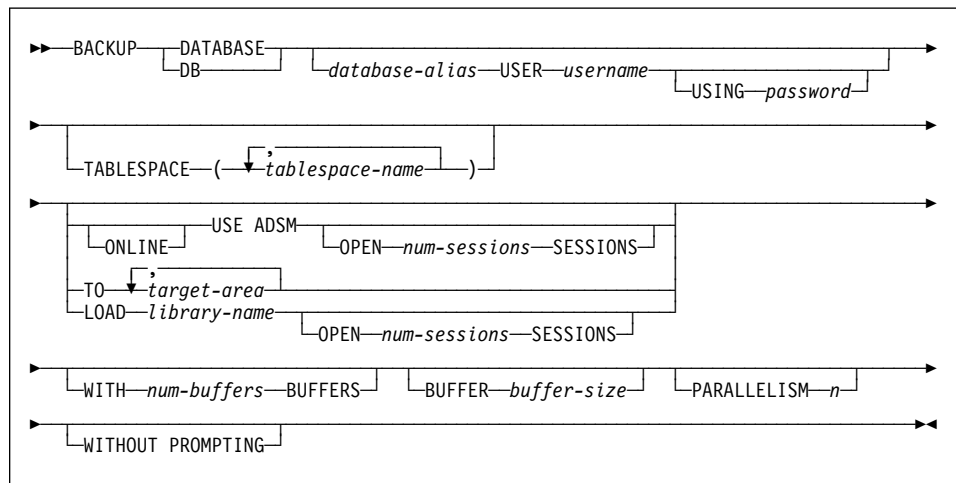
One of the following:

*sysadm*  
*sysctrl*  
*sysmaint*

### Required Connection

Database. This command automatically establishes a connection to the specified database.

### Command Syntax



### Command Parameters

**DATABASE** *database-alias*

Specifies the alias of the database to back up.

**USER** *username*

Identifies the user name under which to back up the database.

**USING** *password*

The password used to authenticate the user name. If the password is omitted, the user is prompted to enter it.

## BACKUP DATABASE

**TABLESPACE** *tablespace-name*

A list of names used to specify the table spaces to be backed up.

**ONLINE**

Specifies online backup. The default is offline backup.

**USE ADSM**

Specifies that the backup is to use ADSM managed output.

**OPEN** *num-sessions* **SESSIONS**

The number of I/O sessions to be used with ADSM or another product.

**TO** *target-area*

A list of directory or tape device names. The full path on which the directory resides must be specified. The target must reside on the database server. This parameter may be repeated to specify the target directories and devices that the backup image will span. If more than one target is specified (*target1*, *target2*, and *target3*, for example), *target1* will be opened first. The media header and special files (including the configuration file, table space table, and history file) are placed in *target1*. All remaining targets are opened, and are then used in parallel during the backup.

Use of tape devices or floppy disks may generate messages and prompts for user input. Valid response options are:

- c** Continue. Continue using the device that generated the warning message (for example, when a new tape has been mounted)
- d** Device terminate. Stop using *only* the device that generated the warning message (for example, when there are no more tapes)
- t** Terminate. Abort the backup or restore utility.

Tape is not supported on OS/2. On OS/2, 0 or 0: can be specified to cause the backup operation to call the user exit program (see the *Administration Guide*). This option is not valid on any other platform.

**LOAD** *library-name*

The name of the shared library (DLL on OS/2 or the Windows operating system) containing the vendor backup and restore I/O functions to be used. It can contain the full path. If the full path is not given, it will default to the path on which the user exit program resides.

**WITH** *num-buffers* **BUFFERS**

The number of buffers to be used.

**BUFFER** *buffer-size*

The size, in pages, of the buffer used when building the backup image. The minimum value for this parameter is 16 pages; the default value is 1024 pages. If a *buffer-size* of 0 is specified, the value of the database manager configuration parameter *backbufsz* is used.

**PARALLELISM** *n*

Specifies the number of buffer manipulators to be spawned during the restore process. The default value is 1.



## WITHOUT PROMPTING

Specifies that the backup will run unattended, and that any actions which normally require user intervention will return an error message.

## Examples

```
db2 backup database sample use adsm open 2 sessions with 4 buffers
```

```
db2 backup database payroll tablespace syscatspace, userspace1 to  
/dev/rmt0, /dev/rmt1 with 8 buffers without prompting
```

## Usage Notes

### Database Level Backup

**Note:** Backup each database on a regular basis. If a database becomes damaged or corrupted, it can be returned to the state of the backed-up copy (see “RESTORE DATABASE” on page 337).

If a successfully restored database was enabled for roll-forward recovery at the time of the backup operation, it can be returned to the state it was in prior to the occurrence of damage (see “ROLLFORWARD DATABASE” on page 344).

The backup can be directed to fixed disk, diskette, tape, ADSM utility, or to other vendor products enabled for DB2.

On UNIX based systems, a backup file name consists of the concatenation of several units of information separated by periods:

### **dbname.type.instance.nodexxxx.catnxxxx.yyyymmddhhmmss.seq**

<b>dbname</b>	1 to 8 character database alias.
<b>type</b>	Type of backup taken (0 for full database, or 3 for table space level backup).
<b>instance</b>	1 to 8 character database instance name.
<b>nodexxxx</b>	The number of the node. In non-partitioned database systems, this is always zero (NODE0000). In a partitioned database system, it is NODExxxx, where xxxx is the number assigned to the node in the db2nodes.cfg file.
<b>catnxxxx</b>	The number of the catalog node for the database. In non-partitioned database systems, this is always zero (CATN0000). In a partitioned database system, it is CATNxxxx, where xxxx is the number assigned to the node in the db2nodes.cfg file.
<b>yyymmdd</b>	Date (year month day).
<b>hhmmss</b>	Time (hour minute second).
<b>seq</b>	A file extension consisting of a 3-digit sequence number.

## BACKUP DATABASE

In addition to fixed disk, tape, ADSM and other vendors, the backup may be directed to diskettes. Since there is no general tape support on OS/2 or the Windows operating system, each type of tape device requires a unique device driver.

To back up to the FAT file system on OS/2 or the Windows operating system, users must conform to the 8.3 naming restriction. The file is placed in a 5-level subdirectory tree as follows:

**dbname.type\db2instance.nodexxx\catnxxx\yyyymmdd\hhmmss.seq**

<b>dbname</b>	1 to 8 character database alias.
<b>type</b>	Type of backup taken. 0 for full database, 3 for tablespace level backup, 4 for copy from a table load, and 5 for table unload.
<b>db2instance</b>	1 to 8 character database instance name.
<b>nodexxx</b>	The number of the node. In non-partitioned database systems, this is always zero (NODE000). In a partitioned database system, it is NODExxx, where xxx is the number assigned to the node in the db2nodes.cfg file.
<b>catnxxx</b>	The number of the catalog node for the database. In non-partitioned database systems, this is always zero (CATN000). In a partitioned database system, it is CATNxxx, where xxx is the number assigned to the node in the db2nodes.cfg file.
<b>yyyymmdd</b>	Date (year month day).
<b>hhmmss</b>	Time (hour minute second).
<b>seq</b>	A file extension consisting of a 3-digit sequence number.

Online backups are permitted only if roll-forward recovery is enabled. The utility can perform an online backup while it is being accessed and modified by other applications.

To perform an offline backup, the utility must be able to connect to the database in exclusive mode. BACKUP fails if any application, including the calling application, is already connected to the database. If the connection is successful, BACKUP locks out other applications until the backup is completed.

Execute BACKUP DATABASE offline when the database is not currently needed.

An offline backup operation will fail if the database is not in a consistent state. If the database is inconsistent, it must be restarted to be brought back to a consistent state through crash recovery before BACKUP is executed (see "RESTART DATABASE" on page 335, and a description of the *autorestart* configuration parameter in the *Administration Guide*). If the database is in a partially restored state after a system failure during restoration, the restore operation must be successfully rerun before a backup can be executed. If the database has been restored and a roll-forward operation is needed, the database must be rolled forward to a consistent state before it can be backed up.

If a database is changed from roll-forward disabled to roll-forward enabled state, either the *logretain* or the *userexit* database configuration options must be enabled before a backup of the database can be made (see “GET DATABASE CONFIGURATION” on page 175).

### Table Space Level Backup

A table space level backup contains one or more table spaces for a database, specified when BACKUP is executed. A table space level backup can be taken online or offline.

Table space level backup can be used to recover from problems that only affect specific table spaces. While this recovery is taking place, all other table spaces are available for processing.

To ensure that restored table spaces are synchronized with the rest of the database, the table spaces must be rolled forward to the end of the log (or to the point where the table spaces were last used). For this reason, table space level backup and restore can only be performed if roll-forward recovery is enabled. If roll-forward recovery is disabled at any time after a table space level backup is executed, it will not be possible to restore from the backup, and then to roll the table space forward to the current point in time. In this case, all table space level backups taken prior to that time are no longer restorable. The restore operation will fail if the user tries to restore from such a backup. In cases where it cannot be determined that the backup is not valid (if, for instance, the database has been restored and rolled forward, thus creating a new log sequence), the restore may be successful, and the broken restore set will be detected during roll-forward recovery.

The user may choose to separate data, index, long field (LONG), and large objects (LOB) into different table spaces. Long field and LOB data for the same table must reside in the same table space.

Each component of a table may be backed up and restored with the table space in which it resides, independently of the other components of the table.

It is not necessary to back up table spaces for temporary tables. If a list of table spaces to be backed up contains such a table space, BACKUP fails.

Table space level backup and restore cannot be run concurrently.

### See Also

“MIGRATE DATABASE” on page 282

“RESTORE DATABASE” on page 337

“ROLLFORWARD DATABASE” on page 344.

## BIND

---

### BIND

Invokes the bind utility, which prepares SQL statements stored in the bind file generated by the precompiler, and creates a package that is stored in the database.

### Scope

This command can be issued from any node in `db2nodes.cfg`. It updates the database catalogs on the catalog node. Its effects are visible to all nodes.

### Authorization

One of the following:

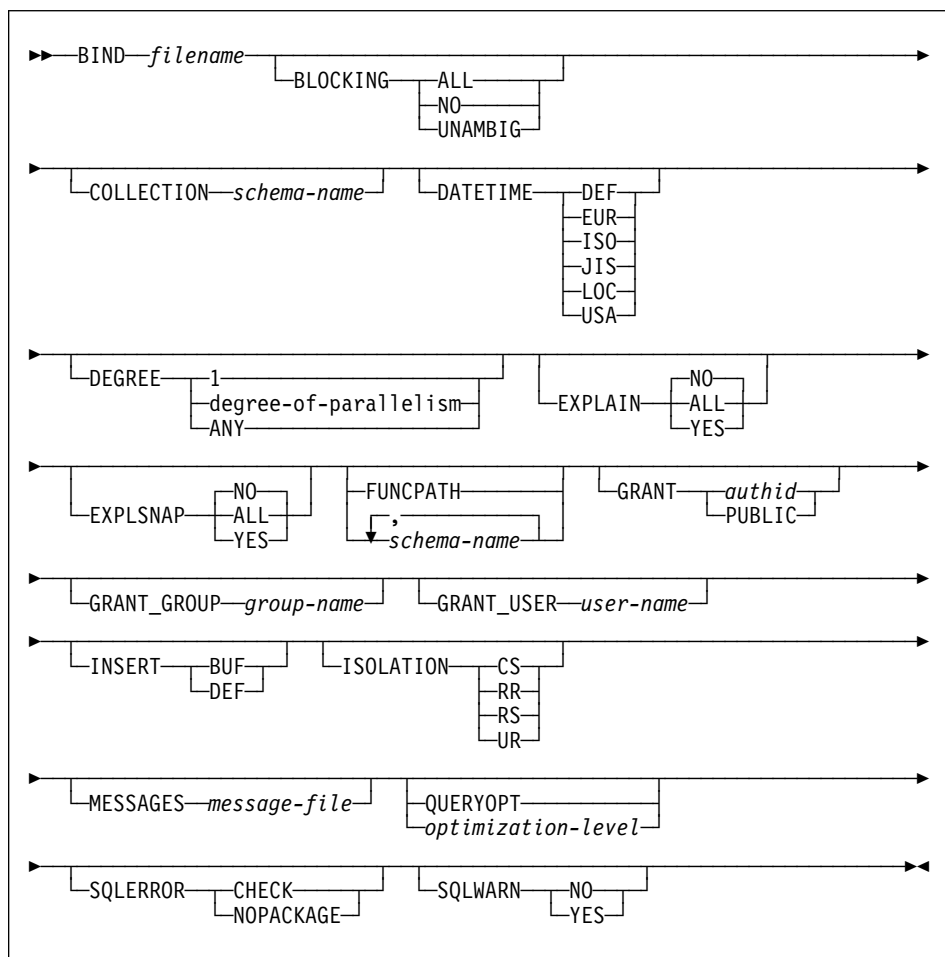
- *sysadm* or *dbadm* authority
- BINDADD privilege if a package does not exist and one of:
  - IMPLICIT\_SCHEMA authority on the database if the schema name of the package does not exist
  - CREATEIN privilege on the schema if the schema name of the package exists
- ALTERIN privilege on the schema if the package exists
- BIND privilege on the package if it exists.

The user also needs all privileges required to compile any static SQL statements in the application. Privileges granted to groups are not used for authorization checking of static statements. If the user has *sysadm* authority, but not explicit privileges to complete the bind, the database manager grants explicit *dbadm* authority automatically.

### Required Connection

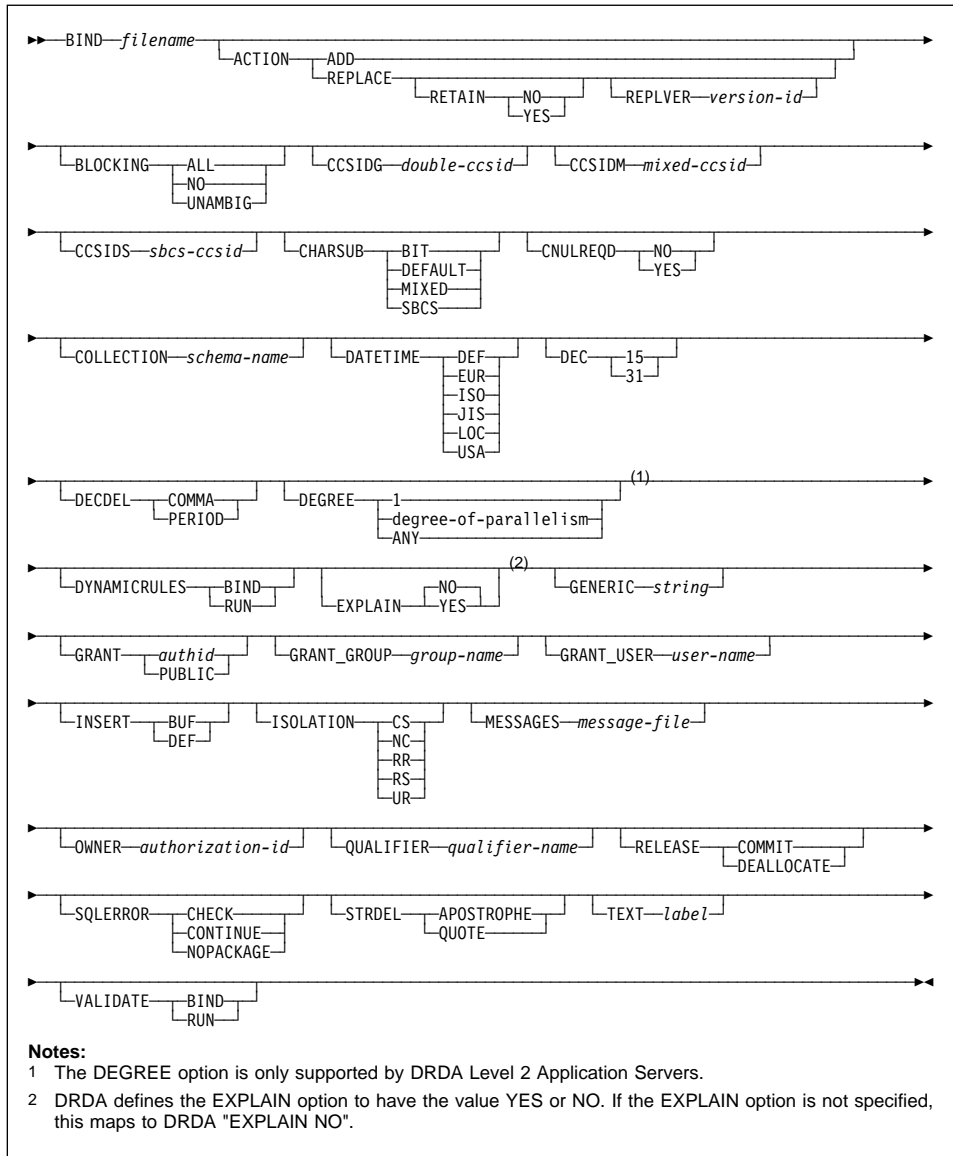
Database. If implicit connect is enabled, a connection to the default database is established.

**Command Syntax**  
For DB2



# BIND

## For DRDA



## Command Parameters

*filename*

Specifies the name of the bind file that was generated when the application program was precompiled, or a list file containing the names of several bind files. Bind files have the extension .bnd. The full path name can be specified.

If a list file is specified, the @ character must be the first character of the list file name. The list file can contain several lines of bind file names. Bind files listed on the same line must be separated by plus (+) characters, but a + cannot appear in front of the first file listed on each line, or after the last bind file listed. For example,

```
/u/smith/sql1lib/bnd/@a11.lst
```

is a list file that contains the following bind files:

```
mybind1.bnd+mybind.bnd2+mybind3.bnd+
mybind4.bnd+mybind5.bnd+
mybind6.bnd+
mybind7.bnd
```

## ACTION

Indicates whether the package can be added or replaced. This DRDA precompile/bind option is not supported by DB2.

### ADD

Indicates that the named package does not exist, and that a new package is to be created. If the package already exists, execution stops, and a diagnostic error message is returned.

### REPLACE

Indicates that the old package is to be replaced by a new one with the same location, collection, and package name.

### RETAIN

Indicates whether EXECUTE authorities are to be preserved when a package is replaced. If ownership of the package changes, the new owner grants the BIND and EXECUTE authority to the previous package owner.

### NO

Does not preserve EXECUTE authorities when a package is replaced.

### YES

Preserves EXECUTE authorities when a package is replaced.

### REPLVER *version-id*

Replaces a specific version of a package. The version identifier specifies which version of the package is to be replaced. Maximum length is 254 characters.

## BLOCKING

For information about row blocking, see the *Administration Guide* or the *Embedded SQL Programming Guide*.

## BIND

### *ALL*

Specifies to block for:

- Read-only cursors
- Cursors not specified as FOR UPDATE OF

Ambiguous cursors are treated as read-only.

### *NO*

Specifies not to block any cursors. Ambiguous cursors are treated as updateable.

### *UNAMBIG*

Specifies to block for:

- Read-only cursors
- Cursors not specified as FOR UPDATE OF

Ambiguous cursors are treated as updateable.

### **CCSIDG** *double-ccsid*

An integer specifying the coded character set identifier (CCSID) to be used for double byte characters in character column definitions (without a specific CCSID clause) in CREATE and ALTER TABLE SQL statements. This DRDA precompile/bind option is not supported by DB2. The DRDA server will use a system defined default value if this option is not specified.

### **CCSIDM** *mixed-ccsid*

An integer specifying the coded character set identifier (CCSID) to be used for mixed byte characters in character column definitions (without a specific CCSID clause) in CREATE and ALTER TABLE SQL statements. This DRDA precompile/bind option is not supported by DB2. The DRDA server will use a system defined default value if this option is not specified.

### **CCSIDS** *sbc-ccsid*

An integer specifying the coded character set identifier (CCSID) to be used for single byte characters in character column definitions (without a specific CCSID clause) in CREATE and ALTER TABLE SQL statements. This DRDA precompile/bind option is not supported by DB2. The DRDA server will use a system defined default value if this option is not specified.

### **CHARSUB**

Designates the default character sub-type that is to be used for column definitions in CREATE and ALTER TABLE SQL statements. This DRDA precompile/bind option is not supported by DB2.

#### *BIT*

Use the FOR BIT DATA SQL character sub-type in all new character columns for which an explicit sub-type is not specified.

#### *DEFAULT*

Use the target system defined default in all new character columns for which an explicit sub-type is not specified.



*MIXED*

Use the FOR MIXED DATA SQL character sub-type in all new character columns for which an explicit sub-type is not specified.

*SBCS*

Use the FOR SBCS DATA SQL character sub-type in all new character columns for which an explicit sub-type is not specified.

**CNULREQD**

This option is related to the **langlevel** precompile option, which is not supported by DRDA. It is valid only if the bind file is created from a C or a C++ application. This DRDA bind option is not supported by DB2.

*NO*

The application was coded on the basis of the **langlevel** SAA1 precompile option with respect to the null terminator in C string host variables.

*YES*

The application was coded on the basis of the **langlevel** MIA precompile option with respect to the null terminator in C string host variables.

**COLLECTION** *schema-name*

Specifies an 8-character collection identifier for the package. If not specified, the authorization identifier for the user processing the package is used.

**DATETIME**

Specifies the date and time format to be used. For more information about date and time formats, see the *SQL Reference*.

*DEF*

Use a date and time format associated with the country code of the database.

*EUR*

Use the IBM standard for Europe date and time format.

*ISO*

Use the date and time format of the International Standards Organization.

*JIS*

Use the date and time format of the Japanese Industrial Standard.

*LOC*

Use the date and time format in local form associated with the country code of the database.

*USA*

Use the IBM standard for U.S. date and time format.

**DEC**

Specifies the maximum precision to be used in decimal arithmetic operations. This DRDA precompile/bind option is not supported by DB2.

## BIND

The DRDA server will use a system defined default value if this option is not specified.

*15*

15-digit precision is used in decimal arithmetic operations.

*31*

31-digit precision is used in decimal arithmetic operations.

## DECDEL

Designates whether a period (.) or a comma (,) will be used as the decimal point indicator in decimal and floating point literals. This DRDA precompile/bind option is not supported by DB2. The DRDA server will use a system defined default value if this option is not specified.

*COMMA*

Use a comma (,) as the decimal point indicator.

*PERIOD*

Use a period (.) as the decimal point indicator.

## DEGREE

Specifies the degree of parallelism for the execution of static SQL statements in an SMP system. This option does not affect CREATE INDEX parallelism.

*1*

The execution of the statement will not use parallelism.

*degree-of-parallelism*

Specifies the degree of parallelism with which the statement can be executed, a value between 2 and 32 767 (inclusive).

*ANY*

Specifies that the execution of the statement can involve parallelism using a degree determined by the database manager.

## DYNAMICRULES

Specifies which authorization identifier to use when dynamic SQL in a package is executed. This DRDA precompile/bind option is not supported by DB2.

*BIND*

Indicates that the authorization identifier used for the execution of dynamic SQL is the package owner.

*RUN*

Indicates that the authorization identifier used for the execution of dynamic SQL is the *authid* of the person executing the package.

## EXPLAIN

Stores information in the Explain tables about the access plans chosen for each SQL statement in the package. DRDA does not support the ALL value for this option.

- NO* Explain information will not be captured.
- YES* Explain tables will be populated with information about the chosen access plan.
- ALL* Explain information for each eligible static SQL statement will be placed in the Explain tables. In addition, Explain information will be gathered for eligible dynamic SQL statements at run time, even if the CURRENT EXPLAIN SNAPSHOT register is set to NO. For more information about special registers, see the *SQL Reference*.  
**Note:** This value for EXPLAIN is not supported by DRDA.

**EXPLSNAP**

Stores Explain Snapshot information in the Explain tables. This DB2 precompile/bind option is not supported by DRDA.

- NO* An Explain Snapshot will not be captured.
- YES* An Explain Snapshot for each eligible static SQL statement will be placed in the Explain tables.
- ALL* An Explain Snapshot for each eligible static SQL statement will be placed in the Explain tables. In addition, Explain Snapshot information will be gathered for eligible dynamic SQL statements at run time, even if the CURRENT EXPLAIN SNAPSHOT register is set to NO. For more information about special registers, see the *SQL Reference*.

**FUNCPATH**

Specifies the function path to be used in resolving user-defined distinct types and functions in static SQL. If this option is not specified, the default function path is "SYSIBM","SYSFUN",USER where USER is the value of the USER special register. This DB2 precompile/bind option is not supported by DRDA.

*schema-name*

A short SQL identifier, either ordinary or delimited, which identifies a schema that exists at the application server. No validation that the schema exists is made at precompile or at bind time. The same schema cannot appear more than once in the function path. The number of schemas that can be specified is limited by the length of the resulting function path, which cannot exceed 254 bytes. The schema SYSIBM does not need to be explicitly specified; it is implicitly assumed to be the first schema if it is not included in the function path. For more information, see the *SQL Reference*.

## BIND

### GENERIC *string*

Provides a means of passing new bind options to a target DRDA database. Supports the binding of new options that are defined in the target database, but that are not known to the local command. Do not use this option to pass bind options that *are* defined in “BIND” on page 98 or “PRECOMPILE PROGRAM” on page 284. This option can substantially improve dynamic SQL performance. The syntax is as follows:

```
generic "option1 value1 option2 value2 ..."
```

Each option and value must be separated by one or more blank spaces. For example, if the target DRDA database is DB2 MVS Version 5, one could use:

```
generic "keepdynamic yes"
```

to bind the new **keepdynamic** YES option, which is not defined locally on the PRECOMPILE PROGRAM or the BIND command.

The maximum length of the string is 1023 bytes. This DRDA bind option is currently only supported by DB2 MVS Version 5; it is not supported by DB2.

### GRANT

*authid*

Grants EXECUTE and BIND privileges to a specified user name or group ID.

*PUBLIC*

Grants EXECUTE and BIND privileges to PUBLIC.

### GRANT\_GROUP *group-name*

Grants EXECUTE and BIND privileges to a specified group ID.

### GRANT\_USER *user-name*

Grants EXECUTE and BIND privileges to a specified user name.

### INSERT

Allows a program being precompiled or bound from a DB2 V2.1 client to a DATABASE 2 Parallel Edition server to request that data inserts be buffered to increase performance.

*BUF*

Specifies that inserts from an application should be buffered.

*DEF*

Specifies that inserts from an application should not be buffered.

### ISOLATION

Determines how far a program bound to this package can be isolated from the effect of other executing programs. For more information about isolation levels, see the *SQL Reference*.

*CS*

Specifies Cursor Stability as the isolation level.

<i>NC</i>	No Commit. Specifies that commitment control is not to be used. This isolation level is not supported by DB2.
<i>RR</i>	Specifies Repeatable Read as the isolation level.
<i>RS</i>	Specifies Read Stability as the isolation level. Read Stability ensures that the execution of SQL statements in the package is isolated from other application processes for rows read and changed by the application.
<i>UR</i>	Specifies Uncommitted Read as the isolation level.

**MESSAGES** *message-file*

Specifies the destination for warning, error, and completion status messages. A message file is created whether the bind is successful or not. If a message file name is not specified, the messages are written to standard output. If the complete path to the file is not specified, the current directory is used. If the name of an existing file is specified, the contents of the file are overwritten.

**OWNER** *authorization-id*

Designates an 8-character authorization identifier for the package owner. The owner must have the privileges required to execute the SQL statements in the package. The default is the primary authorization ID of the precompile/bind process if this option has not been explicitly specified. This DRDA precompile/bind option is not supported by DB2.

**QUALIFIER** *qualifier-name*

Provides an 18-character implicit qualifier for unqualified table names, views, indexes, and aliases contained in the package. The default is the owner's authorization ID, whether or not **owner** is explicitly specified. This DRDA precompile/bind option is not supported by DB2.

**QUERYOPT** *optimization-level*

Indicates the desired level of optimization for all static SQL statements contained in the package. The default value is 5. For the complete range of optimization levels available, see the SET CURRENT QUERY OPTIMIZATION statement in the *SQL Reference*. This DB2 precompile/bind option is not supported by DRDA.

**RELEASE**

Indicates whether resources are released at each COMMIT point, or when the application terminates. This DRDA precompile/bind option is not supported by DB2.

*COMMIT*

Release resources at each COMMIT point. Used for dynamic SQL statements.

*DEALLOCATE*

Release resources only when the application terminates.

# BIND

## SQLERROR

Indicates whether to create a package or a bind file if an error is encountered.

### *CHECK*

Specifies that the target system performs all syntax and semantic checks on the SQL statements being bound. A package will not be created as part of this process. If, while creating a package, an existing package with the same name and version is encountered, the existing package is neither dropped nor replaced if **action replace** was specified.

### *CONTINUE*

A package or a bind file is created even when SQL errors are encountered. This option is not supported by DB2.

### *NOPACKAGE*

A package or a bind file is not created if an error is encountered.

## SQLWARN

Indicates whether warnings will be returned from the compilation of dynamic SQL statements (via PREPARE or EXECUTE IMMEDIATE), or from describe processing (via PREPARE...INTO or DESCRIBE). This DB2 precompile/bind option is not supported by DRDA.

### *NO*

Warnings will not be returned from the SQL compiler.

### *YES*

Warnings will be returned from the SQL compiler.

**Note:** SQLCODE +238 is an exception. It is returned regardless of the **sqlwarn** option value.

## STRDEL

Designates whether an apostrophe (') or double quotation marks (") will be used as the string delimiter within SQL statements. This DRDA precompile/bind option is not supported by DB2. The DRDA server will use a system defined default value if this option is not specified.

### *APOSTROPHE*

Use an apostrophe (') as the string delimiter.

### *QUOTE*

Use double quotation marks (") as the string delimiter.

## TEXT *label*

The description of a package. Maximum length is 255 characters. The default value is blanks. This DRDA precompile/bind option is not supported by DB2.

## VALIDATE

Determines when the database manager checks for authorization errors and object not found errors. The package owner authorization ID is used for validity checking. This DRDA precompile/bind option is not supported by DB2.

## *BIND*

Validation is performed at precompile/bind time. If all objects do not exist, or all authority is not held, error messages are produced. If **sqlerror continue** is specified, a package/bind file is produced despite the error message, but the statements in error are not executable.

## *RUN*

Validation is attempted at bind time. If all objects exist, and all authority is held, no further checking is performed at execution time.

If all objects do not exist, or all authority is not held at precompile/bind time, warning messages are produced, and the package is successfully bound, regardless of the **sqlerror continue** option setting. However, authority checking and existence checking for SQL statements that failed these checks during the precompile/bind process may be redone at execution time.

## Example

The following example binds myapp.bnd (the bind file generated when the myapp.sqc program was precompiled) to the database to which a connection has been established:

```
db2 bind myapp.bnd
```

Any messages resulting from the bind process are sent to standard output.

## Usage Notes

Binding can be done as part of the precompile process for an application program source file, or as a separate step at a later time. Use BIND when binding is performed as a separate process.

The name used to create the package is stored in the bind file, and is based on the source file name from which it was generated (existing paths or extensions are discarded). For example, a precompiled source file called myapp.sq1 generates a default bind file called myapp.bnd and a default package name of MYAPP. However, the bind file name and the package name can be overridden at precompile time by using the **bindfile** and the **package** options.

Binding a package with a schema name that does not already exist will result in the implicit creation of that schema. The schema owner is SYSIBM. The CREATEIN privilege on the schema is granted to PUBLIC.

BIND executes under the transaction that the user has started. After performing the bind, BIND issues a COMMIT (if bind is successful) or a ROLLBACK (if bind is unsuccessful) operation to terminate the current transaction and start another one.

## **BIND**

Binding halts if a fatal error or more than 100 errors occur. If a fatal error occurs during binding, BIND stops binding, attempts to close all files, and discards the package.

Binding application programs has prerequisite requirements and restrictions beyond the scope of this manual. For more detailed information about binding application programs to databases, see the *Embedded SQL Programming Guide*.

### **See Also**

“PRECOMPILE PROGRAM” on page 284.



## CATALOG APPC NODE

Adds an APPC node entry to the node directory. The Advanced Program-to-Program Communications protocol is used to access the remote node.

### Authorization

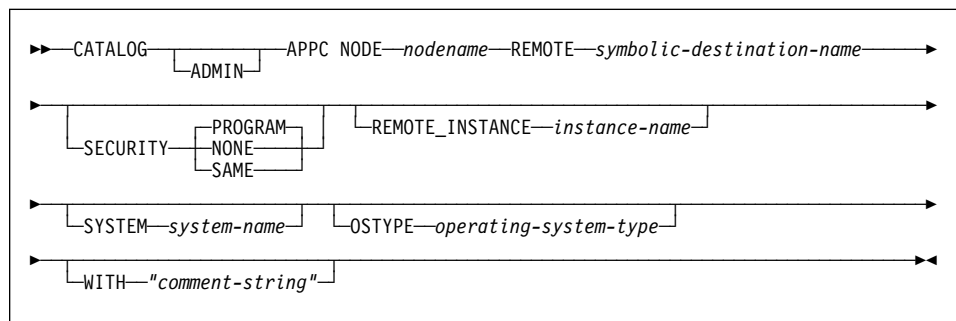
One of the following:

*sysadm*  
*sysctrl*

### Required Connection

None

### Command Syntax



### Command Parameters

#### ADMIN

Specifies administration server nodes.

#### NODE *nodename*

A local alias for the node to be cataloged. This is an arbitrary name on the user's workstation, used to identify the node. It should be a meaningful name to make it easier to remember. The name must conform to database manager naming conventions (see Appendix B, "Naming Conventions" on page 397).

#### REMOTE *symbolic-destination-name*

Specifies the symbolic destination name of the remote partner node. The name corresponds to an entry in the CPI Communications side information table that contains the necessary information for the client to set up an APPC connection to the server (partner LU name, mode name, partner TP name). Maximum length is 8 characters.

## CATALOG APPC NODE

### SECURITY

#### *PROGRAM*

Specifies that both a user name and a password are to be included in the allocation request sent to the partner LU.

#### *NONE*

Specifies that no security information is to be included in the allocation request sent to the partner LU.

#### *SAME*

Specifies that a user name is to be included in the allocation request sent to the partner LU, together with an indicator that the user name has been "already verified". The partner must be configured to accept "already verified" security.

### **REMOTE\_INSTANCE** *instance-name*

Specifies the real name of the instance to which an attachment is being made on the remote server machine.

### **SYSTEM** *system-name*

Specifies a name that is used to identify the server machine.

### **OSTYPE** *operating-system-type*

Specifies the operating system type of the server machine. Valid values are: OS2, AIX, WIN95, NT, HPUX, SUN, MVS, OS400, VM, VSE, SNI, SCO, and SGI.

### **WITH** *"comment-string"*

Describes the node entry in the node directory. Any comment that helps to describe the node can be entered. Maximum length is 30 characters. A carriage return or a line feed character is not permitted. The comment text must be enclosed by single or double quotation marks.

## Example

```
db2 catalog appc node db2appc1 remote db2inst1 security program
with "A remote APPC node"
```

## Usage Notes

The database manager creates the node directory when the first node is cataloged (that is, when the first CATALOG...NODE command is issued). On a client using DATABASE 2 Client Application Enabler for OS/2 or DATABASE 2 Client Application Enabler for Windows, it stores and maintains the node directory in the instance subdirectory where the Client Application Enabler is installed. On a client using DATABASE 2 Client Application Enabler for AIX, it creates the node directory in the DB2 installation directory.

List the contents of the local node directory using "LIST NODE DIRECTORY" on page 245.

**Note:** If directory caching is enabled (see the configuration parameter *dir\_cache* in "GET DATABASE MANAGER CONFIGURATION" on page 184), database, node, and DCS directory files are cached in memory. An application's directory

## CATALOG APPC NODE

cache is created during its first directory lookup. Since the cache is only refreshed when the application modifies any of the directory files, directory changes made by other applications may not be effective until the application has restarted.

To refresh the CLP's directory cache, use "TERMINATE" on page 368. To refresh DB2's shared cache, stop (**db2stop**) and then restart (**db2start**) the database manager. To refresh the directory cache for another application, stop and then restart that application.

## CATALOG APPCLU NODE

---

### CATALOG APPCLU NODE

Writes information to the node directory about a remote workstation that uses APPC as its communication protocol. DB2 uses this information to establish the connection between an application and a remote database cataloged on this node.

This command is available on OS/2 only.

#### Authorization

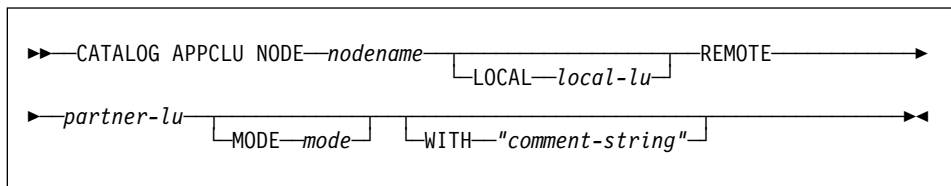
One of the following:

*sysadm*  
*sysctrl*

#### Required Connection

None

#### Command Syntax



#### Command Parameters

##### **NODE** *nodename*

Specifies the name of the remote workstation to catalog. This is the same name that is entered for the node name parameter when cataloging a database residing on that workstation. The name must conform to DB2 naming conventions (see Appendix B, "Naming Conventions" on page 397).

##### **LOCAL** *local-lu*

Specifies the alias of the SNA local logical unit used for the connection. It must be a string containing 1 to 8 non-blank characters. The alias must be entered exactly as it appears (using mixed case characters) in the corresponding SNA definition (from the Communication Manager configuration).

##### **REMOTE** *partner-lu*

Specifies the alias of the SNA remote logical unit used for the connection. It must be a string containing 1 to 8 non-blank characters. The alias must be entered exactly as it appears (using mixed case characters) in the corresponding SNA definition (from the Communication Manager configuration).

## CATALOG APPCLU NODE

**MODE** *mode*

Specifies the SNA transmission mode used for the connection. The name must conform to SNA naming conventions.

If a value is not entered, DB2 stores a character string of eight blanks as the mode type.

**WITH** *"comment-string"*

Describes the node entry in the node directory. Any comment that helps to describe the node can be entered. Maximum length is 30 characters. A carriage return or a line feed character is not permitted. The comment text must be enclosed by single or double quotation marks.

### Example

The following example shows how to catalog REMNODE, a remote workstation that contains a database:

```
db2 catalog appclu node remnode local LU1 remote LU2
mode SQLMOD01 with "Remote node comment"
```

### Usage Notes

This command is identical to the Version 1 CATALOG APPC NODE command, but is different from the Version 2 CATALOG APPC NODE command.

**Note:** Version 1 script files containing the CATALOG APPC NODE command must be modified. Change the keyword APPC to APPCLU to make this command perform as it did in Version 1.

"ATTACH" on page 91 cannot be used with an APPCLU node to attach to a server that has TPNAME other than x'07F6C4C2'.

APPCLU nodes only support security PROGRAM (security SAME and NONE are not supported).

## CATALOG APPN NODE

---

### CATALOG APPN NODE

Writes information to the node directory about a remote workstation that uses APPN as its communication protocol. DB2 uses this information to establish the connection between an application and a remote database cataloged on this node.

This command is available on OS/2 only.

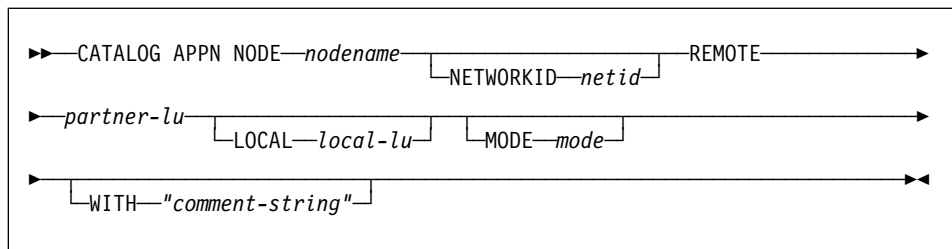
#### Authorization

*sysadm*

#### Required Connection

None

#### Command Syntax



#### Command Parameters

**NODE** *nodename*

Specifies the name of the remote workstation to catalog. This is the same name that is entered for the node name parameter when a database residing on that workstation is cataloged (using "CATALOG DATABASE" on page 118). The name must conform to DB2 naming conventions (see Appendix B, "Naming Conventions" on page 397).

**NETWORKID** *netid*

Specifies the ID of the SNA network where the remote LU resides. This network ID is a string of one to eight characters that follows naming conventions for SNA.

**REMOTE** *partner-lu*

Specifies the SNA partner logical unit used for the connection. Enter the LU name of the remote node. The name must be entered exactly as it appears (using mixed case characters) in the corresponding SNA definition (from the Communication Manager configuration). The name must follow SNA naming conventions.

**LOCAL** *local-lu*

Specifies the alias of the SNA local logical unit used for the connection. It must be a string containing 1 to 8 non-blank characters. The alias must be entered exactly as it appears (using mixed case characters) in the

## CATALOG APPN NODE

corresponding SNA definition (from the Communication Manager configuration).

**MODE** *mode*

Specifies the SNA transmission mode used for the connection. The name must conform to SNA naming conventions.

If a value is not entered, DB2 stores a character string of eight blanks as the mode type.

**WITH** *"comment-string"*

Describes the node entry in the node directory. Any comment that helps to describe the node can be entered. Maximum length is 30 characters. A carriage return or a line feed character is not permitted. The comment text must be enclosed by single or double quotation marks.

### Example

The following example catalogs an APPN node:

```
db2 catalog appn node remnode remote rlu with "Catalog APPN NODE"
```

### Usage Notes

This command is identical to the Version 1 command. (It was removed in Version 2.)

"ATTACH" on page 91 cannot be used with an APPN node to attach to a server that has TPNAME other than x'07F6C4C2'.

APPN nodes only support security PROGRAM (security SAME and NONE are not supported).

# CATALOG DATABASE

---

## CATALOG DATABASE

Stores database location information in the system database directory. The database can be located either on the local workstation or on a remote node.

### Authorization

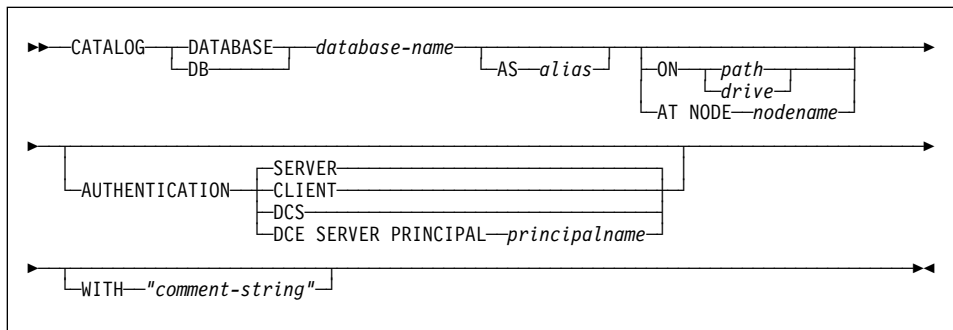
One of the following:

*sysadm*  
*sysctrl*

### Required Connection

None. Directory operations affect the local directory only.

### Command Syntax



### Command Parameters

**DATABASE** *database-name*

Specifies the name of the database to catalog.

**AS** *alias*

Specifies an alias as an alternate name for the database being cataloged. If an alias is not specified, the database manager uses *database-name* as the alias.

**ON** *path/drive*

On UNIX based systems, specifies the path on which the database being cataloged resides. On OS/2 or the Windows operating system, specifies the letter of the drive on which the database being cataloged resides.

**AT NODE** *nodename*

Specifies the name of the node where the database being cataloged resides. This name should match the name of an entry in the node directory. If the node name specified does not exist in the node directory, a warning is returned, but the database is cataloged in the system database directory. The node name should be cataloged in the node directory if a connection to the cataloged database is desired.



## AUTHENTICATION

**Note:** Required only if a DB2 Version 2 or greater client is communicating with a DB2 Version 1 server.

The authentication value is stored for remote databases (it appears in the output from "LIST DATABASE DIRECTORY" on page 231) but it is not stored for local databases.

Specifying an authentication type can result in a performance benefit. For more information about authentication types, see the *Administration Guide*.

### *SERVER*

Specifies that authentication takes place on the node containing the target database.

### *CLIENT*

Specifies that authentication takes place on the node where the application is invoked.

### *DCS*

Specifies that authentication takes place on the node containing the target database, except when using DDCS, when it specifies that authentication takes place at the DRDA AS.

### *DCE*

Specifies that authentication takes place using DCE Security Services. When authentication is DCE, and an APPC connection is used for access, only SECURITY=NONE is supported.

### *SERVER PRINCIPAL principalname*

Fully qualified DCE principal name for the target server. This value is also recorded in the keytab file at the target server.

### **WITH** *"comment-string"*

Describes the database or the database entry in the system database directory. The maximum length of a comment string is 30 characters. A carriage return or a line feed character is not permitted. The comment text must be enclosed by double quotation marks.

## Example

```
db2 catalog database sample on /databases/sample
with "Sample Database"
```

## Usage Notes

Use CATALOG DATABASE to catalog databases located on local or remote nodes, recatalog databases that were uncataloged previously, or maintain multiple aliases for one database (regardless of database location).

DB2 automatically catalogs databases when they are created. It catalogs an entry for the database in the local database directory and another entry in the system database directory. If the database is created from a remote client (or a client which is executing

## CATALOG DATABASE

from a different instance on the same machine), an entry is also made in the system database directory at the client instance.

If neither path nor node name is specified, the database is assumed to be local, and the location of the database is assumed to be that specified in the database manager configuration parameter *dftdbpath*.

Databases on the same node as the database manager instance are cataloged as *indirect* entries. Databases on other nodes are cataloged as *remote* entries.

CATALOG DATABASE automatically creates a system database directory if one does not exist. The system database directory is stored on the path that contains the database manager instance that is being used, and is maintained outside of the database.

List the contents of the system database directory using “LIST DATABASE DIRECTORY” on page 231.

**Note:** If directory caching is enabled (see the configuration parameter *dir\_cache* in “GET DATABASE MANAGER CONFIGURATION” on page 184), database, node, and DCS directory files are cached in memory. An application's directory cache is created during its first directory lookup. Since the cache is only refreshed when the application modifies any of the directory files, directory changes made by other applications may not be effective until the application has restarted.

To refresh the CLP's directory cache, use “TERMINATE” on page 368. To refresh DB2's shared cache, stop (**db2stop**) and then restart (**db2start**) the database manager. To refresh the directory cache for another application, stop and then restart that application.

### See Also

“UNCATALOG DATABASE” on page 369.

## CATALOG DCS DATABASE

Stores information about a remote database in the Database Connection Services (DCS) directory. Such databases are accessed through an Application Requester (AR), such as Distributed Database Connection Services (DDCS). Having a DCS directory entry with a database name matching a database name in the system database directory invokes the specified AR to forward SQL requests to the remote server where the database resides. For more information about DDCS and DCS directory entries, see the *DB2 Connect User's Guide*.

### Authorization

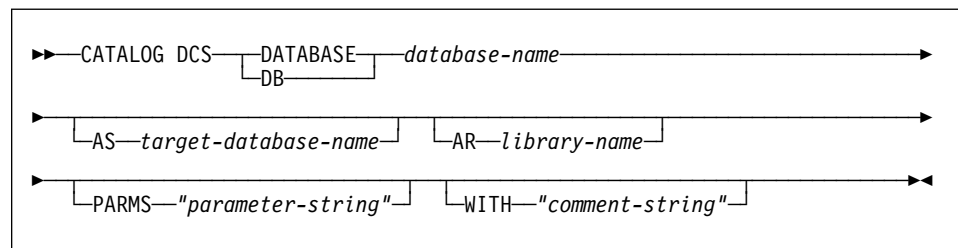
One of the following:

*sysadm*  
*sysctrl*

### Required Connection

None

### Command Syntax



### Command Parameters

**DATABASE** *database-name*

Specifies the alias of the target database to catalog. This alias must match the database name entered during the remote cataloging of this database in the system database directory.

**AS** *target-database-name*

Specifies the name of the target host database to catalog.

**AR** *library-name*

Specifies the name of the Application Requester library that is loaded and used to access a remote database listed in the DCS directory.

**Note:** If using the DDCS AR, do not specify a library name. The default value will cause DDCS to be invoked.

If not using DDCS, specify the library name of the AR, and place that library on the same path as the database manager libraries. On OS/2 or

## CATALOG DCS DATABASE

the Windows operating system, the path is drive:\sql11ib\d11. On UNIX based systems, the path is \$HOME/sql11ib/lib of the instance owner.

**PARMS** *"parameter-string"*

Specifies a parameter string that is to be passed to the AR when it is invoked. For an explanation of what format DDCS expects for this string, see the *DB2 Connect User's Guide*.

The parameter string must be enclosed by double quotation marks.

**WITH** *"comment-string"*

Describes the DCS directory entry. Any comment that helps to describe the database cataloged in this directory can be entered. Maximum length is 30 characters. A carriage return or a line feed character is not permitted. The comment text must be enclosed by double quotation marks.

### Example

The following example catalogs information about the DB1 database, which is a DB2 for MVS host database, into the DCS directory:

```
db2 catalog dcs database db1 as dsn_db_1
with "DB2/MVS location name DSN_DB_1"
```

### Usage Notes

The DB2 Connect program provides connections to DRDA Application Servers such as:

- DATABASE 2 (DB2) for MVS databases on System/370 and System/390 architecture host computers
- Structured Query Language/Data System (SQL/DS) databases on System/370 and System/390 architecture host computers
- OS/400 databases on Application System/400 (AS/400) host computers.

The database manager creates a Database Connection Services directory if one does not exist. This directory is stored on the path that contains the database manager instance that is being used. The DCS directory is maintained outside of the database.

The database must also be cataloged as a remote database in the system database directory.

List the contents of the DCS directory using "LIST DCS DIRECTORY" on page 237.

**Note:** If directory caching is enabled (see the configuration parameter *dir\_cache* in "GET DATABASE MANAGER CONFIGURATION" on page 184), database, node, and DCS directory files are cached in memory. An application's directory cache is created during its first directory lookup. Since the cache is only refreshed when the application modifies any of the directory files, directory changes made by other applications may not be effective until the application has restarted.

To refresh the CLP's directory cache, use "TERMINATE" on page 368. To refresh DB2's shared cache, stop (**db2stop**) and then restart (**db2start**) the database manager.

## CATALOG DCS DATABASE

To refresh the directory cache for another application, stop and then restart that application.

### See Also

“UNCATALOG DCS DATABASE” on page 371.

# CATALOG GLOBAL DATABASE

---

## CATALOG GLOBAL DATABASE

Creates a system database directory entry of the DCE type. This entry is used to define a local alias of the fully qualified DCE directory object name of the target database. The information about that database is stored centrally in the DCE directory.

### Authorization

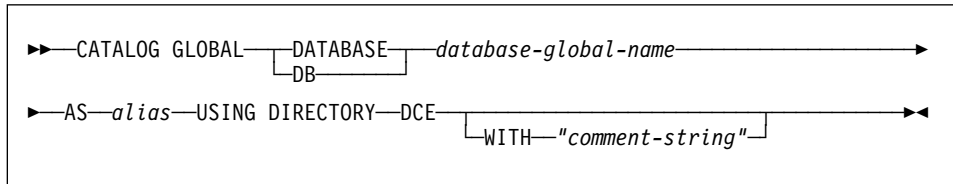
One of the following:

*sysadm*  
*sysctrl*

### Required Connection

None

### Command Syntax



### Command Parameters

**DATABASE** *database-global-name*

Specifies the fully qualified name that uniquely identifies the database in the DCE name space.

**AS** *alias*

Specifies an alternate name for the database being cataloged.

**USING DIRECTORY** *DCE*

Specifies the global directory service being used.

**WITH** *"comment-string"*

Describes the DCE type entry in the system database directory. Any comment that helps to describe the database cataloged in this directory can be entered. Maximum length is 30 characters. A carriage return or a line feed character is not permitted. The comment text must be enclosed by double quotation marks.

### Example

```
db2 catalog global database ../../cell11/subsys/database/DB3
as dbtest using directory dce
```

### Usage Notes

The maximum length of *database-global-name* is 255 bytes. The name must begin with either */.../* or *././*.

**Note:** If directory caching is enabled (see the configuration parameter *dir\_cache* in “GET DATABASE MANAGER CONFIGURATION” on page 184), database, node, and DCS directory files are cached in memory. An application's directory cache is created during its first directory lookup. Since the cache is only refreshed when the application modifies any of the directory files, directory changes made by other applications may not be effective until the application has restarted.

To refresh the CLP's directory cache, use “TERMINATE” on page 368. To refresh DB2's shared cache, stop (**db2stop**) and then restart (**db2start**) the database manager. To refresh the directory cache for another application, stop and then restart that application.

# CATALOG IPX/SPX NODE

---

## CATALOG IPX/SPX NODE

Adds an Internetwork Packet Exchange/Sequenced Packet Exchange (IPX/SPX) node entry to the node directory. The Novell NetWare IPX/SPX communications protocol is used to access the remote node.

This command is available on OS/2, Windows NT, and Windows 95 only.

### Authorization

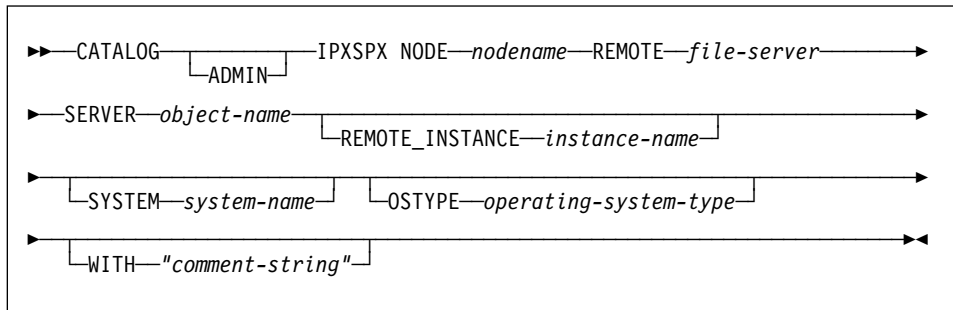
One of the following:

*sysadm*  
*sysctrl*

### Required Connection

None

### Command Syntax



### Command Parameters

#### ADMIN

Specifies that an IPX/SPX administration server node is to be cataloged.

#### NODE *nodename*

A local alias for the node to be cataloged. This is an arbitrary name on the user's workstation, used to identify the node. It should be a meaningful name to make it easier to remember. The name must conform to database manager naming conventions (see Appendix B, "Naming Conventions" on page 397).

#### REMOTE *file-server*

Name of the NetWare file server where the internetwork address of the server database manager instance is registered. The internetwork address is stored in the bindery at the NetWare file server, and is accessed using *object-name*.

**Note:** The following characters are not valid: / \ : ; , \* ?



## CATALOG IPX/SPX NODE

### **SERVER** *object-name*

Name of the database manager instance stored in the bindery of the NetWare file server. Each server database manager instance registered at one NetWare file server must be represented by a unique *object-name*. It is recommended that each database manager instance on the network be represented by a unique *object-name*.

**Note:** The following characters are not valid: / \ ; , \* ?

When cataloging the IPX/SPX client to use *file server* addressing, specify the file server and object name as defined above. When cataloging the IPX/SPX client to use *direct* addressing, specify *file-server* as \*, and specify the server's IPX/SPX internetwork address in the *object-name* parameter. Use "db2ipxad - Get IPX/SPX Internetwork Address" on page 42 to retrieve the server's IPX/SPX internetwork address. For more information about the addressing methods, see one of the *Quick Beginnings* books.

### **REMOTE\_INSTANCE** *instance-name*

Specifies the name of the server instance to which an attachment is being made.

### **SYSTEM** *system-name*

Specifies the DB2 system name that is used to identify the server machine.

### **OSTYPE** *operating-system-type*

Specifies the operating system type of the server machine. Valid values are: OS2, AIX, WIN95, NT, HPUX, SUN, MVS, OS400, VM, VSE, SNI, SCO, and SGI.

### **WITH** *"comment-string"*

Describes the node entry in the node directory. Any comment that helps to describe the node can be entered. Maximum length is 30 characters. A carriage return or a line feed character is not permitted. The comment text must be enclosed by single or double quotation marks.

## Examples

```
db2 catalog ipxspx node db2ipx1 remote netwsrv server db2inst1
with "A remote IPX/SPX node"
```

```
db2 catalog ipxspx node db2ipx2 remote * server 09212700.400011527745.879E
with "IPX/SPX node using direct addr"
```

## Usage Notes

The database manager creates the node directory when the first node is cataloged (that is, when the first CATALOG...NODE command is issued). On a client using DATABASE 2 Client Application Enabler for OS/2 or DATABASE 2 Client Application Enabler for Windows, it stores and maintains the node directory in the instance subdirectory where the Client Application Enabler is installed. On a client using DATABASE 2 Client Application Enabler for AIX, it creates the node directory in the DB2 installation directory.

## CATALOG IPX/SPX NODE

List the contents of the local node directory using “LIST NODE DIRECTORY” on page 245.

**Note:** If directory caching is enabled (see the configuration parameter *dir\_cache* in “GET DATABASE MANAGER CONFIGURATION” on page 184), database, node, and DCS directory files are cached in memory. An application's directory cache is created during its first directory lookup. Since the cache is only refreshed when the application modifies any of the directory files, directory changes made by other applications may not be effective until the application has restarted.

To refresh the CLP's directory cache, use “TERMINATE” on page 368. To refresh DB2's shared cache, stop (**db2stop**) and then restart (**db2start**) the database manager. To refresh the directory cache for another application, stop and then restart that application.

## CATALOG LOCAL NODE

Creates a local alias for an instance that resides on the same machine. A local node should be cataloged when there is more than one instance on the same workstation to be accessed from the user's client. Interprocess Communications (IPC) is used to access the local node.

### Authorization

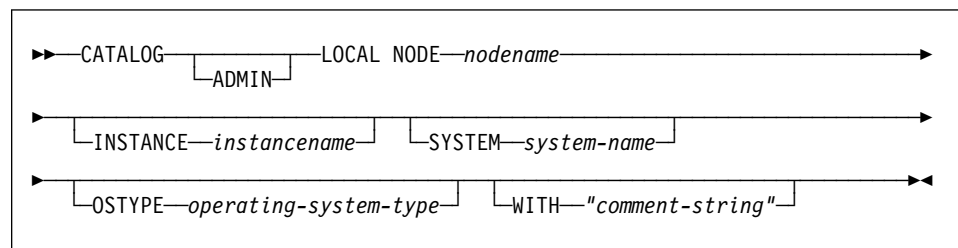
One of the following:

*sysadm*  
*sysctrl*

### Required Connection

None

### Command Syntax



### Command Parameters

#### ADMIN

Specifies that a local administration server node is to be cataloged.

#### NODE *nodename*

A local alias for the node to be cataloged. This is an arbitrary name on the user's workstation, used to identify the node. It should be a meaningful name to make it easier to remember. The name must conform to database manager naming conventions (see Appendix B, "Naming Conventions" on page 397).

#### INSTANCE *instancename*

Name of the local instance to be accessed.

#### SYSTEM *system-name*

Specifies the DB2 system name that is used to identify the server machine.

#### OSTYPE *operating-system-type*

Specifies the operating system type of the server machine. Valid values are: OS2, AIX, WIN95, NT, HPUX, SUN, MVS, OS400, VM, VSE, SNI, SCO, and SGI.

## CATALOG LOCAL NODE

### WITH *"comment-string"*

Describes the node entry in the node directory. Any comment that helps to describe the node can be entered. Maximum length is 30 characters. A carriage return or a line feed character is not permitted. The comment text must be enclosed by single or double quotation marks.

### Example

Workstation A has two server instances, `inst1` and `inst2`. To create databases at both instances from a single CLP session, issue the following sequence of commands (assume the **DB2INSTANCE** environment variable is set to `inst1`):

1. Create a local database at `inst1`:  

```
db2 create database mydb1
```
2. Catalog another server instance on this workstation:  

```
db2 catalog local node mynode2 instance inst2
```
3. Create a database at `mynode2`:  

```
db2 attach to mynode2  
db2 create database mydb2
```

### Usage Notes

**Note:** If directory caching is enabled (see the configuration parameter *dir\_cache* in "GET DATABASE MANAGER CONFIGURATION" on page 184), database, node, and DCS directory files are cached in memory. An application's directory cache is created during its first directory lookup. Since the cache is only refreshed when the application modifies any of the directory files, directory changes made by other applications may not be effective until the application has restarted.

To refresh the CLP's directory cache, use "TERMINATE" on page 368. To refresh DB2's shared cache, stop (**db2stop**) and then restart (**db2start**) the database manager. To refresh the directory cache for another application, stop and then restart that application.

---

## CATALOG NAMED PIPE NODE

Adds a named pipe node entry to the node directory. The named pipe is used to access the remote node.

This command is available on Windows NT only.

### Authorization

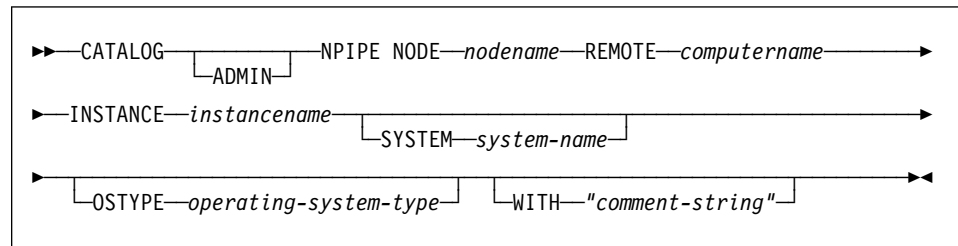
One of the following:

*sysadm*  
*sysctrl*

### Required Connection

None

### Command Syntax



### Command Parameters

#### ADMIN

Specifies that an NPIPE administration server node is to be cataloged.

#### NODE *nodename*

A local alias for the node to be cataloged. This is an arbitrary name on the user's workstation, used to identify the node. It should be a meaningful name to make it easier to remember. The name must conform to database manager naming conventions (see Appendix B, "Naming Conventions" on page 397).

#### REMOTE *computername*

The computer name of the node on which the target database resides. Maximum length is 15 characters.

#### INSTANCE *instancename*

Name of the server instance on which the target database resides. Identical to the name of the remote named pipe, which is used to communicate with the remote node.

## CATALOG NAMED PIPE NODE

**SYSTEM** *system-name*

Specifies the DB2 system name that is used to identify the server machine.

**OSTYPE** *operating-system-type*

Specifies the operating system type of the server machine. Valid values are: OS2, AIX, WIN95, NT, HPUX, SUN, MVS, OS400, VM, VSE, SNI, SCO, and SGI.

**WITH** *"comment-string"*

Describes the node entry in the node directory. Any comment that helps to describe the node can be entered. Maximum length is 30 characters. A carriage return or a line feed character is not permitted. The comment text must be enclosed by single or double quotation marks.

### Example

```
db2 catalog npipe node db2np1 remote nphost instance db2inst1
with "A remote named pipe node."
```

### Usage Notes

The database manager creates the node directory when the first node is cataloged (that is, when the first CATALOG...NODE command is issued). On a client using DATABASE 2 Client Application Enabler for OS/2 or DATABASE 2 Client Application Enabler for Windows, it stores and maintains the node directory in the instance subdirectory where the Client Application Enabler is installed. On a client using DATABASE 2 Client Application Enabler for AIX, it creates the node directory in the DB2 installation directory.

List the contents of the local node directory using "LIST NODE DIRECTORY" on page 245.

**Note:** If directory caching is enabled (see the configuration parameter *dir\_cache* in "GET DATABASE MANAGER CONFIGURATION" on page 184), database, node, and DCS directory files are cached in memory. An application's directory cache is created during its first directory lookup. Since the cache is only refreshed when the application modifies any of the directory files, directory changes made by other applications may not be effective until the application has restarted.

To refresh the CLP's directory cache, use "TERMINATE" on page 368. To refresh DB2's shared cache, stop (**db2stop**) and then restart (**db2start**) the database manager. To refresh the directory cache for another application, stop and then restart that application.

## CATALOG NETBIOS NODE

Adds a NetBIOS node entry to the node directory. The NetBIOS communications protocol is used to access the remote node.

This command is available on OS/2, Windows NT, and Windows 95 only.

### Authorization

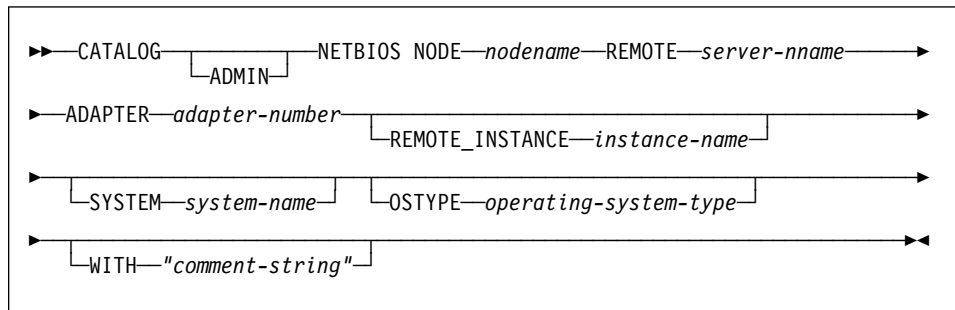
One of the following:

*sysadm*  
*sysctrl*

### Required Connection

None. Directory operations affect the local directory only.

### Command Syntax



### Command Parameters

#### ADMIN

Specifies administration server nodes.

#### NODE *nodename*

A local alias for the node to be cataloged. This is an arbitrary name on the user's workstation, used to identify the node. It should be a meaningful name to make it easier to remember. The name must conform to database manager naming conventions (see Appendix B, "Naming Conventions" on page 397).

#### REMOTE *server-nname*

The name of the remote workstation where the target database resides. This name must conform to the naming conventions for the database manager. This is the workstation name (*nname*) found in the database manager configuration file of the server workstation.

## CATALOG NETBIOS NODE

**ADAPTER** *adapter-number*

Specifies the local, logical, outgoing LAN adapter number. The default value is zero.

**REMOTE\_INSTANCE** *instance-name*

Specifies the real name of the instance to which an attachment is being made on the remote server machine.

**SYSTEM** *system-name*

Specifies a name that is used to identify the server machine.

**OSTYPE** *operating-system-type*

Specifies the operating system type of the server machine. Valid values are: OS2, AIX, WIN95, NT, HPUX, SUN, MVS, OS400, VM, VSE, SNI, SCO, and SGI.

**WITH** *"comment-string"*

Describes the node entry in the node directory. Any comment that helps to describe the node can be entered. Maximum length is 30 characters. A carriage return or a line feed character is not permitted. The comment text must be enclosed by single or double quotation marks.

### Example

```
db2 catalog netbios node db2netb1 remote db2inst1 adapter 0
with "A remote NetBIOS node"
```

### Usage Notes

The database manager creates the node directory when the first node is cataloged (that is, when the first CATALOG...NODE command is issued). On a client using DATABASE 2 Client Application Enabler for OS/2 or DATABASE 2 Client Application Enabler for Windows, it stores and maintains the node directory in the instance subdirectory where the Client Application Enabler is installed. On a client using DATABASE 2 Client Application Enabler for AIX, it creates the node directory in the DB2 installation directory.

List the contents of the local node directory using "LIST NODE DIRECTORY" on page 245.

**Note:** If directory caching is enabled (see the configuration parameter *dir\_cache* in "GET DATABASE MANAGER CONFIGURATION" on page 184), database, node, and DCS directory files are cached in memory. An application's directory cache is created during its first directory lookup. Since the cache is only refreshed when the application modifies any of the directory files, directory changes made by other applications may not be effective until the application has restarted.

To refresh the CLP's directory cache, use "TERMINATE" on page 368. To refresh DB2's shared cache, stop (**db2stop**) and then restart (**db2start**) the database manager. To refresh the directory cache for another application, stop and then restart that application.



## CATALOG ODBC DATA SOURCE

Catalogs a user or system ODBC data source.

A *data source*, in ODBC (Open Database Connectivity) terminology, is a user-defined name for a specific database or file system. That name is used to access the database or file system through ODBC APIs. On Windows NT and Windows 95, either user or system data sources can be cataloged. A user data source is only visible to the user who cataloged it, whereas a system data source is visible to and can be used by all other users.

This command is available on Windows NT, Windows 95, and Windows 3.1 only.

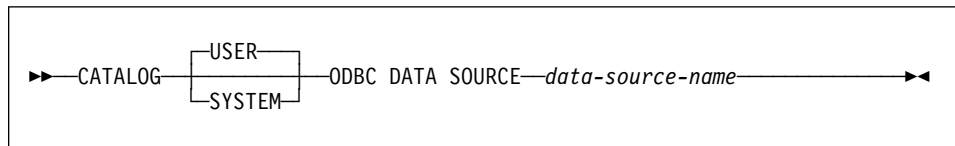
### Authorization

None to catalog a user ODBC data source; *sysadm* to catalog a system ODBC data source.

### Required Connection

None

### Command Syntax



### Command Parameters

#### USER

Catalog a user data source. This is the default if no keyword is specified.

#### SYSTEM

Catalog a system data source.

#### ODBC DATA SOURCE *data-source-name*

Specifies the name of the data source to be cataloged. Maximum length is 32 characters.

### See Also

“LIST ODBC DATA SOURCES” on page 251

“UNCATALOG ODBC DATA SOURCE” on page 374.

# CATALOG TCP/IP NODE

---

## CATALOG TCP/IP NODE

Adds a Transmission Control Protocol/Internet Protocol (TCP/IP) node entry to the node directory. The TCP/IP communications protocol is used to access the remote node.

### Authorization

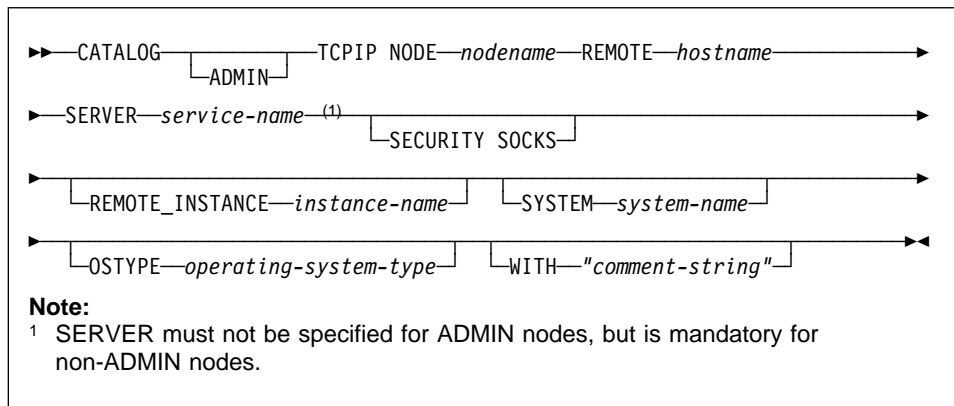
One of the following:

*sysadm*  
*sysctrl*

### Required Connection

None. Directory operations affect the local directory only.

### Command Syntax



### Command Parameters

#### ADMIN

Specifies that a TCP/IP administration server node is to be cataloged.

#### NODE *nodename*

A local alias for the node to be cataloged. This is an arbitrary name on the user's workstation, used to identify the node. It should be a meaningful name to make it easier to remember. The name must conform to database manager naming conventions (see Appendix B, "Naming Conventions" on page 397).

#### REMOTE *hostname*

The host name of the node where the target database resides. The host name is the name of the node that is known to the TCP/IP network. Maximum length is 255 characters.

### **SERVER** *service-name*

Specifies the service name or the port number of the server database manager instance.

The CATALOG TCPIP NODE command is run on a client.

- If a service name is specified, the *services* file on the client is used to map the service name to a port number. A service name is specified in the database manager configuration file, and the *services* file on the server is used to map this service name to a port number. The port number on the client and the server must match.

**Note:** A port number, instead of a service name, can be specified in the database manager configuration file on the server, but this is not recommended.

- If a port number is specified, it must match the port number associated with the service name specified in the server's database manager configuration file. No service name needs to be specified in the local TCP/IP *services* file.

The value of *service-name* is used as a key to search the local *services* file for the associated port number. If a matching entry is not found, and *service-name* is numeric, the value is interpreted as the port number.

Maximum length is 14 characters. This parameter is case sensitive.

**Note:** This parameter must not be specified for ADMIN nodes. The value on ADMIN nodes is always 523.

### **SECURITY SOCKS**

Specifies that the node will be SOCKS-enabled.

The following environment variables are mandatory and *must* be set to enable SOCKS:

#### **SOCKS\_NS**

The Domain Name Server for resolving the host address of the SOCKS server. This should be an IP address.

#### **SOCKS\_SERVER**

The fully qualified host name or the IP address of the SOCKS server. If the SOCKSified DB2 client is unable to resolve the fully qualified host name, it assumes that an IP address has been entered.

One of the following conditions should be true:

- The SOCKS server should be reachable via the domain name server
- It should be listed in the *hosts* file. The location of this file is described in the TCP/IP documentation.
- It should be in an IP address format.

If these environment variables are set after a **db2start** has been issued, it is necessary to issue a TERMINATE command.

## CATALOG TCP/IP NODE

**REMOTE\_INSTANCE** *instance-name*

Specifies the name of the server instance to which an attachment is being made.

**SYSTEM** *system-name*

Specifies the DB2 system name that is used to identify the server machine.

**OSTYPE** *operating-system-type*

Specifies the operating system type of the server machine. Valid values are: OS2, AIX, WIN95, NT, HPUX, SUN, MVS, OS400, VM, VSE, SNI, SCO, and SGI.

**WITH** *"comment-string"*

Describes the node entry in the node directory. Any comment that helps to describe the node can be entered. Maximum length is 30 characters. A carriage return or a line feed character is not permitted. The comment text must be enclosed by single or double quotation marks.

### Examples

```
db2 catalog tcpip node db2tcp1 remote tcphost server db2inst1
with "A remote TCP/IP node"
```

```
db2 catalog tcpip node db2tcp2 remote 9.21.15.235 server db2inst2
with "TCP/IP node using IP address"
```

### Usage Notes

The database manager creates the node directory when the first node is cataloged (that is, when the first CATALOG...NODE command is issued). On a client using DATABASE 2 Client Application Enabler for OS/2 or DATABASE 2 Client Application Enabler for Windows, it stores and maintains the node directory in the instance subdirectory where the Client Application Enabler is installed. On a client using DATABASE 2 Client Application Enabler for AIX, it creates the node directory in the DB2 installation directory.

List the contents of the local node directory using "LIST NODE DIRECTORY" on page 245.

**Note:** If directory caching is enabled (see the configuration parameter *dir\_cache* in "GET DATABASE MANAGER CONFIGURATION" on page 184), database, node, and DCS directory files are cached in memory. An application's directory cache is created during its first directory lookup. Since the cache is only refreshed when the application modifies any of the directory files, directory changes made by other applications may not be effective until the application has restarted.

To refresh the CLP's directory cache, use "TERMINATE" on page 368. To refresh DB2's shared cache, stop (**db2stop**) and then restart (**db2start**) the database manager. To refresh the directory cache for another application, stop and then restart that application.

---

## CHANGE DATABASE COMMENT

Changes a database comment in the system database directory or the local database directory. New comment text can be substituted for text currently associated with a comment.

### Scope

This command only affects the node on which it is executed.

### Authorization

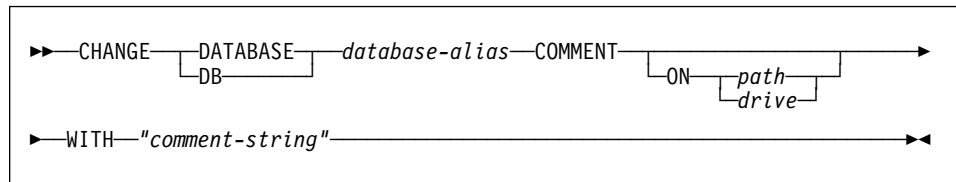
One of the following:

*sysadm*  
*sysctrl*

### Required Connection

None

### Command Syntax



### Command Parameters

#### **DATABASE** *database-alias*

Specifies the alias of the database whose comment is to be changed. To change the comment in the system database directory, specify the alias for the database. To change the comment in the local database directory, specify the path where the database resides (with the *path* parameter), and enter the name (not the alias) of the database.

#### **ON** *path/drive*

On UNIX based systems, specifies the path on which the database resides, and changes the comment in the local database directory. If a path is not specified, the database comment for the entry in the system database directory is changed. On OS/2 or the Windows operating system, specifies the letter of the drive on which the database resides.

#### **WITH** *"comment-string"*

Describes the entry in the system database directory or the local database directory. Any comment that helps to describe the cataloged database can be entered. The maximum length of a comment string is 30 characters. A carriage return or a line feed character is not permitted. The comment text must be enclosed by double quotation marks.

## CHANGE DATABASE COMMENT

### Example

The following example changes the text in the system database directory comment for the SAMPLE database from "Test 2 - Holding" to "Test 2 - Add employee inf rows":

```
db2 change database sample comment
with "Test 2 - Add employee inf rows"
```

### Usage Notes

New comment text replaces existing text. To append information, enter the old comment text, followed by the new text.

Only the comment for an entry associated with the database alias is modified. Other entries with the same database name, but with different aliases, are not affected.

If the path is specified, the database alias must be cataloged in the local database directory. If the path is not specified, the database alias must be cataloged in the system database directory.

### See Also

"CREATE DATABASE" on page 143.

---

## CHANGE ISOLATION LEVEL

Changes the way that DB2 isolates data from other processes while a database is being accessed.

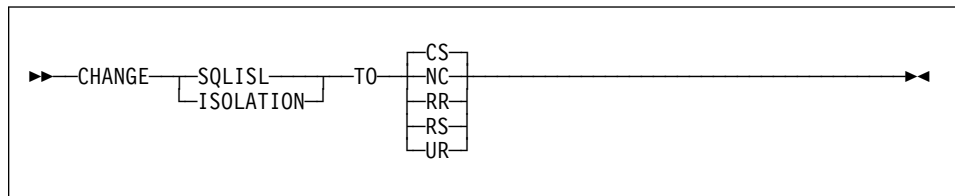
### Authorization

None

### Required Connection

None

### Command Syntax



### Command Parameters

#### TO

*CS*

Specifies cursor stability as the isolation level.

*NC*

Specifies no commit as the isolation level. Not supported by DB2.

*RR*

Specifies repeatable read as the isolation level.

*RS*

Specifies read stability as the isolation level.

*UR*

Specifies uncommitted read as the isolation level.

### Usage Notes

DB2 uses isolation levels to maintain data integrity in a database. The isolation level defines the degree to which an application process is isolated (shielded) from changes made by other concurrently executing application processes.

If a selected isolation level is not supported by a database, it is automatically escalated to a supported level at connect time.

Isolation level changes are not permitted while connected to a database with a type 1 connection (see "SET CLIENT" on page 353). Changes are permitted using a type 2 connection, but should be made with caution, because the changes will apply to every connection made from the same command line processor back-end process. The user

## CHANGE ISOLATION LEVEL

assumes responsibility for remembering which isolation level applies to which connected database.

In the following example, a user is in DB2 interactive mode following creation of the SAMPLE database:

```
update command options using c off
catalog db sample as sample2

set client connect 2

connect to sample
connect to sample2

change isolation to cs
set connection sample
declare c1 cursor for select * from org
open c1
fetch c1 for 3 rows

change isolation to rr
fetch c1 for 2 rows
```

An SQL0514N error occurs because c1 is not in a prepared state for this isolation level.

```
change isolation to cs
set connection sample2
fetch c1 for 2 rows
```

An SQL0514N error occurs because c1 is not in a prepared state for this database.

```
declare c1 cursor for select division from org
```

A DB21029E error occurs because cursor c1 has already been declared and opened.

```
set connection sample
fetch c1 for 2 rows
```

This works because the original database (SAMPLE) was used with the original isolation level (CS).

For more information about isolation levels, see the *SQL Reference*.

### See Also

“QUERY CLIENT” on page 304.



---

### CREATE DATABASE

Initializes a new database with an optional user-defined collating sequence, creates the three initial table spaces, creates the system tables, and allocates the recovery log.

This command is not valid on a client.

#### Scope

In a multi-node environment, this command affects all nodes that are listed in the `db2nodes.cfg` file.

The node from which this command is issued becomes the catalog node for the new database.

#### Authorization

One of the following:

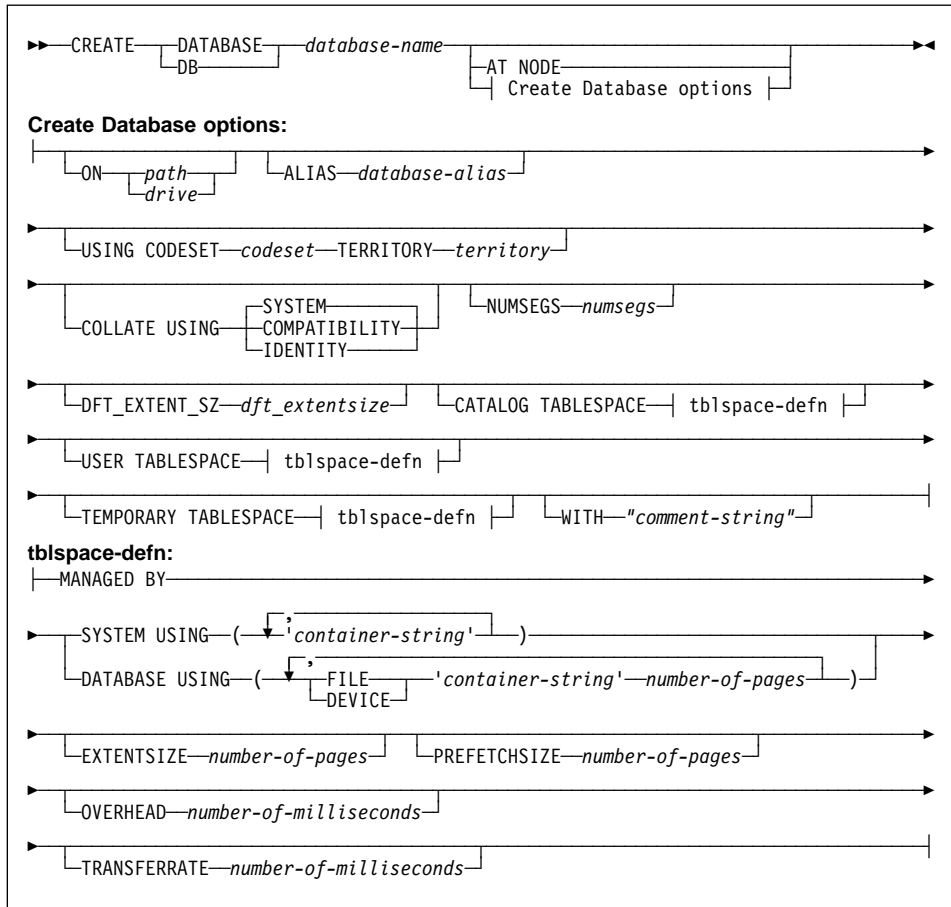
*sysadm*  
*sysctrl*

#### Required Connection

Instance. To create a database at another (remote) node, it is necessary to first attach to that node. A database connection is temporarily established by this command during processing.

# CREATE DATABASE

## Command Syntax



### Notes:

1. The code set and territory values specified must be a valid combination. For a list of valid combinations, see one of the *Quick Beginnings* books.
2. For details on the **tblspace-defn** parameters, see the CREATE TABLESPACE statement in the *SQL Reference*. The table space definitions specified on CREATE DATABASE apply to all nodes on which the database is being created. They cannot be specified separately for each node. If the table space definitions are to be created differently on particular nodes, the CREATE TABLESPACE statement must be used.

## Command Parameters

### DATABASE *database-name*

A name to be assigned to the new database. This must be a unique name that differentiates the database from any other database in either the local

## CREATE DATABASE

database directory or the system database directory. The name must conform to naming conventions for databases.

### **AT NODE**

Specifies that the database is to be created only on the node that issues the command. This parameter is not intended for general use. For example, it should be used with “RESTORE DATABASE” on page 337 if the database partition at a node was damaged and must be re-created. Improper use of this parameter can cause inconsistencies in the system, so it should only be used with caution.

**Note:** If this parameter is used to recreate a database partition that was dropped (because it was damaged), the database at this node will be in the restore-pending state. After recreating the database partition, the database must immediately be restored on this node.

### **ON path/drive**

On UNIX based systems, specifies the path on which to create the database. If a path is not specified, the database is created on the default database path specified in the database manager configuration file (*dftdbpath* parameter). Maximum length is 205 characters. On OS/2 or the Windows operating system, specifies the letter of the drive on which to create the database.

**Note:** For MPP systems, a database should not be created in an NFS-mounted directory. If a path is not specified, ensure that the *dftdbpath* database manager configuration parameter is not set to an NFS-mounted path (for example, on UNIX based systems, it should not specify the \$HOME directory of the instance owner). The path specified for this command in an MPP system cannot be a relative path.

### **ALIAS database-alias**

An alias for the database in the system database directory. If no alias is provided, the specified database name is used.

### **USING CODESET codeset**

Specifies the code set to be used for data entered into this database.

### **TERRITORY territory**

Specifies the territory to be used for data entered into this database.

### **COLLATE USING**

Identifies the type of collating sequence to be used for the database. Once the database has been created, the collating sequence cannot be changed.

#### *COMPATIBILITY*

The DB2 Version 2 collating sequence. Some collation tables have been enhanced. This option specifies that the previous version of these tables is to be used.

#### *IDENTITY*

Identity collating sequence, in which strings are compared byte for byte.

#### *SYSTEM*

Collating sequence based on the current territory.

## CREATE DATABASE

For information about how the database collating sequence is used, see the *SQL Reference*.

### **NUMSEGS** *numsegs*

Specifies the number of segment directories that will be created and used to store DAT, IDX, LF, LB, and LBA files for any default SMS table spaces. This parameter does not affect DMS table spaces, any SMS table spaces with explicit creation characteristics (created when the database is created), or any SMS table spaces explicitly created after the database is created.

### **DFT\_EXTENT\_SZ** *dft\_extentsize*

Specifies the default extent size of table spaces in the database.

### **CATALOG TABLESPACE** *tblspace-defn*

Specifies the definition of the table space which will hold the catalog tables, SYSCATSPACE. If not specified, SYSCATSPACE will be created as a System Managed Space (SMS) table space with *numsegs* number of directories as containers, and with an extent size of *dft\_extentsize*. For example, the following containers would be created if *numsegs* were specified to be 5:

```
/u/smith/smith/NODE0000/SQL00001/SQLT0000.0  
/u/smith/smith/NODE0000/SQL00001/SQLT0000.1  
/u/smith/smith/NODE0000/SQL00001/SQLT0000.2  
/u/smith/smith/NODE0000/SQL00001/SQLT0000.3  
/u/smith/smith/NODE0000/SQL00001/SQLT0000.4
```

In an MPP system, the catalog table space is only created on the catalog node (the node on which the CREATE DATABASE command is issued).

### **USER TABLESPACE** *tblspace-defn*

Specifies the definition of the initial user table space, USERSPACE1. If not specified, USERSPACE1 will be created as an SMS table space with *numsegs* number of directories as containers, and with an extent size of *dft\_extentsize*. For example, the following containers would be created if *numsegs* were specified to be 5:

```
/u/smith/smith/NODE0000/SQL00001/SQLT0001.0  
/u/smith/smith/NODE0000/SQL00001/SQLT0001.1  
/u/smith/smith/NODE0000/SQL00001/SQLT0001.2  
/u/smith/smith/NODE0000/SQL00001/SQLT0001.3  
/u/smith/smith/NODE0000/SQL00001/SQLT0001.4
```

### **TEMPORARY TABLESPACE** *tblspace-defn*

Specifies the definition of the initial temporary table space, TEMPSPACE1. If not specified, TEMPSPACE1 will be created as an SMS table space with *numsegs* number of directories as containers, and with an extent size of *dft\_extentsize*. For example, the following containers would be created if *numsegs* were specified to be 5:

```
/u/smith/smith/NODE0000/SQL00001/SQLT0002.0  
/u/smith/smith/NODE0000/SQL00001/SQLT0002.1  
/u/smith/smith/NODE0000/SQL00001/SQLT0002.2  
/u/smith/smith/NODE0000/SQL00001/SQLT0002.3  
/u/smith/smith/NODE0000/SQL00001/SQLT0002.4
```

### WITH "comment-string"

Describes the database entry in the database directory. Any comment that helps to describe the database can be entered. Maximum length is 30 characters. A carriage return or a line feed character is not permitted. The comment text must be enclosed by single or double quotation marks.

## Usage Notes

### CREATE DATABASE:

- Creates a database in the specified subdirectory. In an MPP system, creates the database on all nodes listed in `db2nodes.cfg`, and creates a `$DB2INSTANCE/NODExxxx` directory under the specified subdirectory at each node. In a non-MPP system, creates a `$DB2INSTANCE/NODE0000` directory under the specified subdirectory.
- Creates the system catalog tables and recovery log.
- Catalogs the database in the following database directories:
  - server's local database directory on the path indicated by *path* or, if the path is not specified, the default database path defined in the database manager system configuration file. A local database directory resides on each file system that contains a database.
  - server's system database directory for the attached instance. The resulting directory entry will contain the database name and a database alias.

If the command was issued from a remote client, the client's system database directory is also updated with the database name and an alias.

Creates a system or a local database directory if neither exists. If specified, the comment and code set values are placed in both directories.

- Stores the specified code set, territory, and collating sequence. A flag is set in the database configuration file if the collating sequence consists of unique weights, or if it is the identity sequence.
- Creates the schemata called SYSCAT, SYSFUN, SYSIBM, and SYSSTAT with SYSIBM as the owner. The server node on which this command is issued becomes the catalog node for the new database. Two nodegroups are created automatically: IBMDEFAULTGROUP and IBMCATGROUP. For more information, see the *SQL Reference*.
- Binds the previously defined database manager bind files to the database (these are listed in `db2ubind.lst`). If one or more of these files do not bind successfully, CREATE DATABASE returns a warning in the SQLCA, and provides information about the binds that failed. If a bind fails, the user can take corrective action and manually bind the failing file. The database is created in any case. A schema called NULLID is implicitly created when performing the binds with CREATEIN privilege granted to PUBLIC.
- Creates SYSCATSPACE, TEMPSPACE1, and USERSPACE1 table spaces. The SYSCATSPACE table space is only created on the catalog node.

## CREATE DATABASE

- Grants the following:
  - DBADM authority, and CONNECT, CREATETAB, BINDADD, CREATE\_NOT\_FENCED, and IMPLICIT\_SCHEMA privileges to the database creator
  - CONNECT, CREATETAB, BINDADD, and IMPLICIT\_SCHEMA privileges to PUBLIC
  - SELECT privilege on each system catalog to PUBLIC
  - BIND and EXECUTE privilege to PUBLIC for each successfully bound utility.

With *dbadm* authority, one can grant these privileges to (and revoke them from) other users or PUBLIC. If another administrator with *sysadm* or *dbadm* authority over the database revokes these privileges, the database creator nevertheless retains them.

In an MPP environment, the database manager creates a subdirectory, `$DB2INSTANCE/NODExxxx`, under the specified or default path on all nodes. The *xxxx* is the node number as defined in the `db2nodes.cfg` file (that is, node 0 becomes `NODE0000`). Subdirectories `SQL00001` through `SQLnnnnn` will reside on this path. This ensures that the database objects associated with different nodes are stored in different directories (even if the subdirectory `$DB2INSTANCE` under the specified or default path is shared by all nodes).

CREATE DATABASE will fail if the application is already connected to a database.

Use CATALOG DATABASE to define different alias names for the new database.

### See Also

“BIND” on page 98

“CATALOG DATABASE” on page 118

“DROP DATABASE” on page 157.

## DEACTIVATE DATABASE

Stops the specified database.

### Scope

In an MPP system, this command deactivates the specified database on all nodes in the system. If one or more of these nodes encounters an error, a warning is returned. The database will be successfully deactivated on some nodes, but may remain activated on the nodes encountering the error.

### Authorization

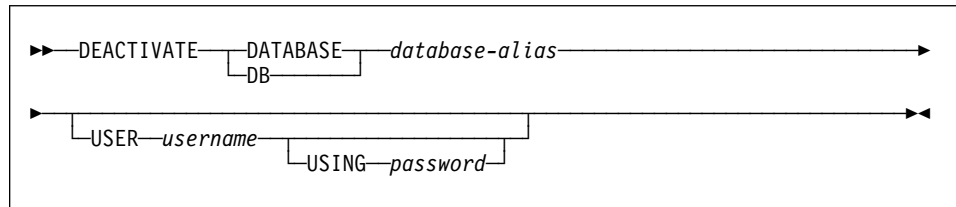
One of the following:

*sysadm*  
*sysctrl*  
*sysmaint*

### Required Connection

None

### Command Syntax



### Command Parameters

**DATABASE** *database-alias*

Specifies the alias of the database to be stopped.

**USER** *username*

Specifies the user starting the database.

**USING** *password*

Specifies the password for the user ID.

### Usage Notes

Databases initialized by `ACTIVATE DATABASE` can be shut down by `DEACTIVATE DATABASE` or by `db2stop`. If a database was initialized by `ACTIVATE DATABASE`, the last application disconnecting from the database will not shut down the database, and `DEACTIVATE DATABASE` must be used. (In this case, `db2stop` will also shut down the database.)

**Note:** The application issuing the `DEACTIVATE DATABASE` command cannot have an active database connection to any database.

## DEACTIVATE DATABASE

### See Also

“ACTIVATE DATABASE” on page 87

“STOP DATABASE MANAGER” on page 365.



---

**DEREGISTER**

Deregisters the DB2 server from the network server. The DB2 server's network address is removed from the network server.

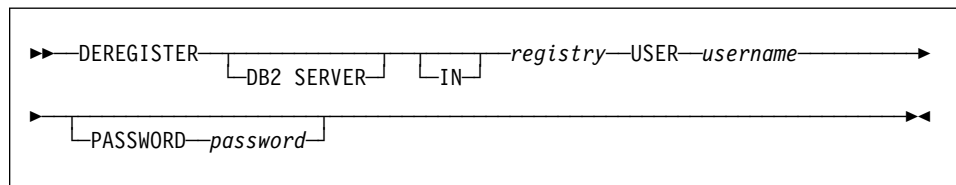
This command is not available on the Windows operating system.

**Authorization**

None

**Required Connection**

None

**Command Syntax****Command Parameters**

**IN** *registry*

Indicates where on the network server to deregister the DB2 server. In this release, the only supported value is NWBINDERY (NetWare bindery).

**USER** *username*

User ID to log into the network server. The user ID must have SUPERVISOR or Workgroup Manager security equivalence.

**PASSWORD** *password*

Password used to log into the network server. The password must have SUPERVISOR or Workgroup Manager security equivalence.

**Usage Notes**

This command *must* be issued locally from the DB2 server. It is not supported remotely.

This command is only relevant when using file server addressing to connect a client and server.

**See Also**

“REGISTER” on page 315.

# DESCRIBE

---

## DESCRIBE

This command:

- Displays the SQLDA information about a SELECT statement
- Displays columns of a table or a view
- Displays indexes of a table or a view

### Authorization

To display the SQLDA information about a SELECT statement, one of the privileges or authorities listed below for each table or view referenced in the SELECT statement is required.

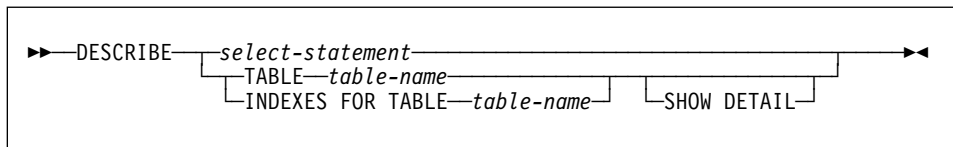
To display the columns or indexes of a table or a view, one of the privileges or authorities listed below for the system catalogs SYSCAT.COLUMNS (DESCRIBE TABLE) and SYSCAT.INDEXES (DESCRIBE INDEXES FOR TABLE) is required:

SELECT privilege  
CONTROL privilege  
*sysadm* or *dbadm* authority

### Required Connection

Database. If implicit connect is enabled, a connection to the default database is established.

### Command Syntax



### Command Parameters

*select-statement*

Identifies the statement about which information is wanted. The SELECT statement is automatically prepared by CLP.

**TABLE** *table-name*

Specifies the table or view to be described. The fully qualified name or alias in the form *schema.table-name* must be used. The *schema* is the user name under which the table or view was created.

The DESCRIBE TABLE command lists the following information about each column:

- Column name
- Type schema
- Type name

## DESCRIBE

- Length
- Scale
- Nulls (yes/no)

### INDEXES FOR TABLE *table-name*

Specifies the table or view for which indexes need to be described. The fully qualified name or alias in the form *schema.table-name* must be used. The *schema* is the user name under which the table or view was created.

The DESCRIBE INDEXES FOR TABLE command lists the following information about each index of the table or view:

- Index schema
- Index name
- Unique rule
- Column count

### SHOW DETAIL

For the DESCRIBE TABLE command, specifies that output include the following additional information:

- Whether a CHARACTER, VARCHAR or LONG VARCHAR column was defined as FOR BIT DATA
- Column number
- Partitioning key sequence
- Code page
- Default

For the DESCRIBE INDEXES FOR TABLE command, specifies that output include the following additional information:

- Column names

# DESCRIBE

## Examples

### Describing a SELECT Statement

The following example shows how to describe a SELECT statement:

```
db2 "describe select * from staff"
```

SQLDA Information					
sqldaid :	SQLDA	sqldabc:	896	sqln:	20
Column Information					
sqltype		sqllen	sqlname.data		sqlname.length
-----		-----	-----		-----
500	SMALLINT	2	ID		2
449	VARCHAR	9	NAME		4
501	SMALLINT	2	DEPT		4
453	CHARACTER	5	JOB		3
501	SMALLINT	2	YEARS		5
485	DECIMAL	7, 2	SALARY		6
485	DECIMAL	7, 2	COMM		4

### Describing a Table

The following example shows how to describe a table:

```
db2 describe table user1.department
```

Table: USER1.DEPARTMENT					
Column name	Type schema	Type name	Length	Scale	Nulls
-----			-----		-----
AREA	SYSIBM	SMALLINT	2	0	No
DEPT	SYSIBM	CHARACTER	3	0	No
DEPTNAME	SYSIBM	CHARACTER	20	0	Yes

**Describing a Table Index**

The following example shows how to describe a table index:

```
db2 describe indexes for table user1.department
```

Table: USER1.DEPARTMENT			
Index schema	Index name	Unique rule	Number of columns
USER1	IDX1	U	2

## DETACH

---

### DETACH

Removes the logical DBMS instance attachment, and terminates the physical communication connection if there are no other logical connections using this layer.

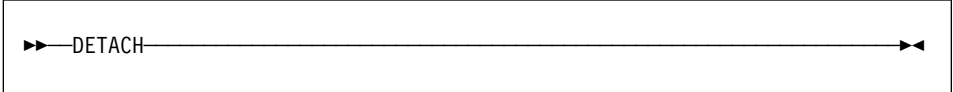
#### Authorization

None

#### Required Connection

None. Removes an existing instance attachment.

#### Command Syntax



A diagram showing the command syntax for DETACH. It consists of a horizontal line with a double arrowhead pointing left at the start and a double arrowhead pointing right at the end. The word "DETACH" is positioned in the middle of the line, between the two arrowheads.

#### Command Parameters

None

#### See Also

“ATTACH” on page 91.

## DROP DATABASE

Deletes the database contents and all log files for the database, uncatalogs the database, and deletes the database subdirectory.

### Scope

By default, this command affects all nodes that are listed in the `db2nodes.cfg` file.

### Authorization

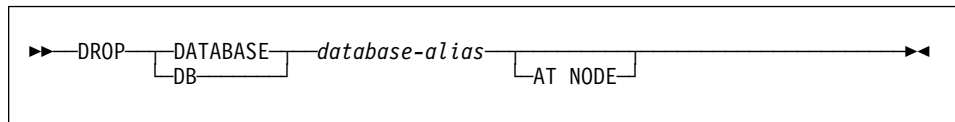
One of the following:

*sysadm*  
*sysctrl*

### Required Connection

Instance. An explicit attachment is not required. If the database is listed as remote, an instance attachment to the remote node is established for the duration of the command.

### Command Syntax



### Command Parameters

**DATABASE** *database-alias*

Specifies the alias of the database to be dropped. The database must be cataloged in the system database directory.

**AT NODE**

Specifies that the database is to be deleted only on the node that issued the DROP DATABASE command. This parameter is used by utilities supplied with DB2 Universal Database Extended Enterprise Edition, and is not intended for general use. Improper use of this parameter can cause inconsistencies in the system, so it should only be used with caution.

### Example

The following example deletes the database referenced by the database alias SAMPLE:

```
db2 drop database sample
```

### Usage Notes

DROP DATABASE deletes all user data and log files. If the log files are needed for a roll-forward recovery after a restore operation, the files should be saved prior to issuing this command.

## DROP DATABASE

The database must not be in use; all users must be disconnected from the database before the database can be dropped.

To be dropped, a database must be cataloged in the system database directory. Only the specified database alias is removed from the system database directory. If other aliases with the same database name exist, their entries remain. If the database being dropped is the last entry in the local database directory, the local database directory is deleted automatically.

If DROP DATABASE is issued from a remote client (or from a different instance on the same machine), the specified alias is removed from the client's system database directory. The corresponding database name is removed from the server's system database directory.

### See Also

"CATALOG DATABASE" on page 118

"CREATE DATABASE" on page 143

"UNCATALOG DATABASE" on page 369.



---

### DROP NODE VERIFY

Verifies if a node exists in the nodegroups of any databases, and if an event monitor is defined on the node. This command should be used prior to dropping a node from an MPP system.

#### Scope

This command only affects the node on which it is issued.

#### Authorization

*sysadm*

#### Command Syntax

```
▶▶—DROP NODE VERIFY—————▶▶
```

#### Command Parameters

None

#### Usage Notes

If a message is returned, indicating that the node is not in use, use “STOP DATABASE MANAGER” on page 365 with DROP NODENUM to remove the entry for the node from the `db2nodes.cfg` file, which removes the node from the database system.

If a message is returned, indicating that the node is in use, the following actions should be taken:

1. If the node contains data, redistribute the data to remove it from the node using “REDISTRIBUTE NODEGROUP” on page 312. Use either the DROP NODE option on the REDISTRIBUTE NODEGROUP command, or the ALTER NODEGROUP statement to remove the node from any nodegroups for the database. This must be done for each database that contains the node in a nodegroup. For more information, see the *SQL Reference*.
2. Drop any event monitors that are defined on the node.
3. Rerun DROP NODE VERIFY to ensure that the database is no longer in use.

#### See Also

“STOP DATABASE MANAGER” on page 365.

# ECHO

---

## ECHO

Permits the user to write character strings to standard output.

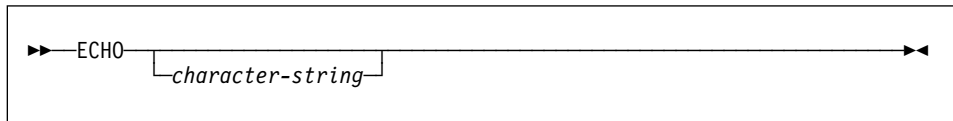
### Authorization

None

### Required Connection

None

### Command Syntax



### Command Parameters

*character-string*

Any character string.

### Usage Notes

If an input file is used as standard input, or comments are to be printed without being interpreted by the command shell, the ECHO command will print character strings directly to standard output.

One line is printed each time that ECHO is issued.

The ECHO command is not affected by the verbose (-v) option (see “Command Line Processor Invocation and Options” on page 69).

## EXPORT

Exports data from a database to one of several external file formats. The user specifies the data to be exported by supplying an SQL SELECT statement.

### Authorization

One of the following:

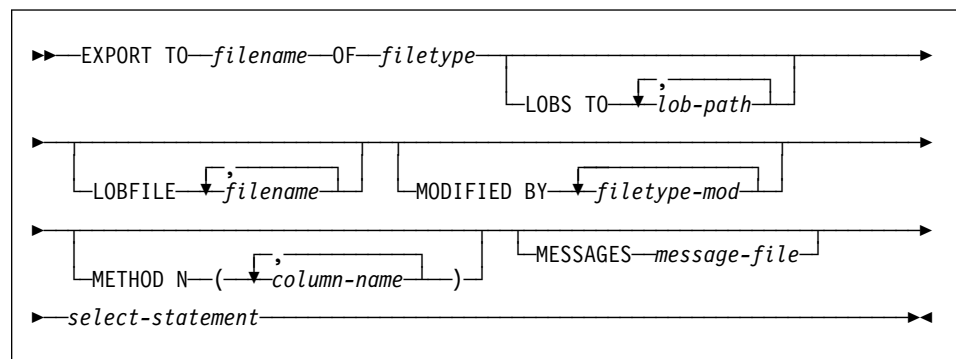
*sysadm*  
*dbadm*

or CONTROL or SELECT privilege on each participating table or view.

### Required Connection

Database. If implicit connect is enabled, a connection to the default database is established.

### Command Syntax



### Command Parameters

#### TO *filename*

Specifies the name of the file to which data is to be exported. If the path is omitted, the current working directory is used. If the complete path to the file is not specified, EXPORT uses the current directory and the default drive as the destination.

If the name of a file that already exists is specified, EXPORT overwrites the contents of the file; it does not append the information.

#### OF *filetype*

Specifies the format of the data in the output file:

- DEL (delimited ASCII format), which is used by a variety of database manager and file manager programs

## EXPORT

- WSF (work sheet format), which is used by programs such as:
  - Lotus 1-2-3
  - Lotus Symphony
- IXF (integrated exchange format, PC version), which makes it possible to import the file again into the same or another DB2 table. Definitions of the table, as well as any existing indexes, are saved in the IXF file, except when columns are specified in the SELECT statement.

For more information about file formats, see Appendix C, “IMPORT/EXPORT/LOAD Utility File Formats” on page 399.

**LOBS TO** *lob-path*

Specifies the path or paths to store the LOB files. When file space is exhausted on the first path, the second path will be used, and so on.

**LOBFILE** *filename*

Specifies the base file name or names of the LOB files. When name space is exhausted for the first name, the second name will be used, and so on.

When creating LOB files during an export, file names are constructed by appending the current base name from this list to the current path (from *lob-path*), and then appending a 3-digit sequence number. For example, if the current LOB path is the directory `/u/foo/lob/path`, and the current LOB file name is `bar`, the LOB files created will be `/u/foo/lob/path/bar.001`, `/u/foo/lob/path/bar.002`, and so on.

**MODIFIED BY** *filetype-mod*

Specifies additional options (see page 164).

**METHOD N** *column-name*

Specifies the column name(s) to be used in the output file. If this parameter is not specified, the column names in the existing table are used. This parameter is valid only for WSF and IXF files.

**MESSAGES** *message-file*

Specifies the destination for warning and error messages that occur during export. If the file already exists, EXPORT appends the information. If *message-file* is omitted, the messages are written to standard output.

*select-statement*

Specifies the SELECT statement that will get the information to be exported.

## Examples

The following example shows how to export information from the STAFF table in the SAMPLE database (to which the user must be connected) to `myfile.ixf`, with the output in IXF format:

```
db2 export to myfile.ixf of ixf messages msgs.txt select * from staff
```

The following example shows how to export the information about employees in Department 20 from the STAFF table (in the database to which the user must be connected) to awards.ixf, with the output in IXF format:

```
db2 export to awards.ixf of ixf messages msgs.txt select * from staff
      where dept = 20
```

### Usage Notes

Be sure to complete all table operations and release all locks before issuing the EXPORT command. This can be done either by issuing a COMMIT after closing all cursors opened WITH HOLD, or by issuing a ROLLBACK. A COMMIT is performed during the export process.

Table aliases can be used in the SELECT statement.

The messages placed in the message file include the information returned from the message retrieval service. Each message begins on a new line.

The export utility produces a warning message whenever a character column with a length greater than 254 is selected for export to DEL format files.

PC/IXF import should be used to move data between databases. If character data containing row separators is exported to a delimited ASCII (DEL) file and processed by a text transfer program (moving, for example, between OS/2 and AIX systems), fields containing the row separators will shrink or expand.

PC/IXF file format specifications permit migration of data between OS/2 (IBM Extended Services for OS/2, OS/2 Extended Edition and DB2 for OS/2) databases and DB2 for AIX databases via export, binary copying of files between OS/2 and AIX, and import. The file copying step is not necessary if the source and the target databases are both accessible from the same client.

DB2 Connect can be used to export tables from DRDA servers such as DB2 for MVS, SQL/DS, and OS/400. Only PC/IXF export is supported.

The EXPORT command will not create multiple-part PC/IXF files when invoked from an AIX system.

# EXPORT

## File Type Modifications

**Note:** The export utility does not issue a warning if an attempt is made to use unsupported file types with the MODIFIED BY option. If this is attempted, the export fails, and an error code is returned.

<i>Table 6 (Page 1 of 2). Valid File Type Modifications (EXPORT)</i>	
Modification	Description
<b>All File Formats</b>	
lobsinfile	<i>lob-path</i> specifies the path to the files containing LOB values.
<b>DEL (Delimited ASCII) File Format</b>	
char <del>x</del>	<p><i>x</i> is a single character string delimiter. The default value is a double quotation mark ("). The specified character is used in place of double quotation marks to enclose a character string.<sup>a</sup></p> <p>The single quotation mark (') can also be specified as a character string delimiter as follows:</p> <pre>modified by char<del>del</del>''</pre>
col <del>del</del> <i>x</i>	<p><i>x</i> is a single character column delimiter. The default value is a comma (.). The specified character is used in place of a comma to signal the end of a column.<sup>a</sup></p> <p>In the following example, col<del>del</del>; causes the export utility to interpret any semicolon (;) it encounters as a column delimiter:</p> <pre>db2 "export to temp of <del>del</del> modified by col<del>del</del>; select * from staff where dept = 20"</pre>
datesiso	Date format. Causes all date data values to be exported in ISO format.
decplusblank	Plus sign character. Causes positive decimal values to be prefixed with a blank space instead of a plus sign (+). The default action is to prefix positive decimal values with a plus sign.
decpt <i>x</i>	<i>x</i> is a single character substitute for the period as a decimal point character. The default value is a period (.). The specified character is used in place of a period as a decimal point character. <sup>a</sup>
<b>WSF File Format</b>	
1	Creates a WSF file that is compatible with Lotus 1-2-3 Release 1, or Lotus 1-2-3 Release 1a. <sup>b</sup> This is the default.
2	Creates a WSF file that is compatible with Lotus Symphony Release 1.0. <sup>b</sup>
3	Creates a WSF file that is compatible with Lotus 1-2-3 Version 2, or Lotus Symphony Release 1.1. <sup>b</sup>
4	Creates a WSF file containing DBCS characters.

Table 6 (Page 2 of 2). Valid File Type Modifications (EXPORT)

Modification	Description
<p><b>Note:</b></p> <ul style="list-style-type: none"> <li>• <sup>a</sup> Table 7 on page 165 lists the characters that can be used as delimiter overrides.</li> <li>• <sup>b</sup> These files can also be directed to a specific product by specifying an L for Lotus 1-2-3, or an S for Symphony in the <i>filetype-mod</i> parameter string.</li> </ul> <p>Only one value or product designator may be specified.</p>	

Table 7. Valid Delimiters

Hex	Char	Character Name
X'22'	"	Double Quotation Marks
X'25'	%	Percent Sign
X'26'	&	Ampersand
X'27'	'	Apostrophe
X'28'	(	Left Parenthesis
X'29'	)	Right Parenthesis
X'2A'	*	Asterisk
X'2C'	,	Comma
X'2E'	.	Period (not valid as a character string delimiter)
X'2F'	/	Slash
X'3A'	:	Colon
X'3B'	;	Semicolon
X'3C'	<	Less Than Sign
X'3D'	=	Equals Sign
X'3E'	>	Greater Than Sign
X'3F'	?	Question Mark
X'7C'		Vertical Bar
X'5F'	_	Underscore (valid in the SBCS environment only)

**Note:**

- The characters are the same for all code pages.
- The following cannot be used as delimiters: binary zero, the new line character, or a blank space.
- Specified delimiters must be different characters so that each delimiter can be uniquely identified.
- It is the user's responsibility to ensure that the chosen delimiter character is not part of the actual data to be loaded. If it is, unexpected errors may occur.

## EXPORT

### See Also

“IMPORT” on page 212  
“LOAD” on page 262.



## FORCE APPLICATION

Forces local or remote users or applications off the system to allow for maintenance on a server.

**Attention:** If an operation that cannot be interrupted (RESTORE DATABASE, for example) is forced, the operation must be successfully re-executed before the database becomes available.

### Scope

This command affects all nodes that are listed in the \$HOME/sq11ib/db2nodes.cfg file.

### Authorization

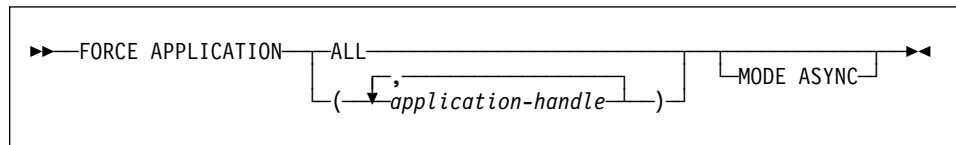
One of the following:

*sysadm*  
*sysctrl*

### Required Connection

Instance. To force users off a remote server, it is first necessary to attach to that server. If no attachment exists, this command is executed locally.

### Command Syntax



### Command Parameters

#### APPLICATION

*ALL*

All applications will be disconnected from the database.

*application-handle*

Specifies the agent to be terminated. List the values using "LIST APPLICATIONS" on page 225.

#### MODE ASYNC

The command does not wait for all specified users to be terminated before returning; it returns as soon as the function has been successfully issued or an error (such as invalid syntax) is discovered.

This is the only mode that is currently supported.

## FORCE APPLICATION

### Example

The following example forces two users, with *application-handle* values of 41408 and 55458, to disconnect from the database:

```
db2 force application ( 41408, 55458 )
```

### Usage Notes

**db2stop** cannot be executed during a force. The database manager remains active so that subsequent database manager operations can be handled without the need for **db2start**.

To preserve database integrity, only users who are idling or executing interruptible database operations can be terminated.

Users creating a database cannot be forced.

After a FORCE has been issued, the database will still accept requests to connect. Additional forces may be required to completely force all users off.

### See Also

“ATTACH” on page 91

“LIST APPLICATIONS” on page 225.

---

### GET ADMIN CONFIGURATION

Returns the values of individual entries in the database manager configuration file that are relevant to the DB2 Administration Server. The DB2 Administration Server is a special DB2 instance that enables remote administration of DB2 servers. The following database manager configuration parameters are displayed:

- AGENT\_STACK\_SZ
- AUTHENTICATION
- DIAGLEVEL
- DIAGPATH
- DISCOVER
- DISCOVER\_COMM
- FILESERVER
- IPX\_SOCKET
- NNAME
- OBJECTNAME
- QUERY\_HEAP\_SZ
- SVCENAME
- SYSADM\_GROUP
- SYSCTRL\_GROUP
- SYSMANT\_GROUP
- TPNAME
- TRUST\_ALLCLNTS
- TRUST\_CLNTAUTH

**Note:** It is not recommended that the SVCENAME parameter, set by the installation program, be modified by the user. The administration server service name is set to use the DB2 registered TCP/IP port (523).

For more information about these parameters, see “GET DATABASE MANAGER CONFIGURATION” on page 184.

### Scope

This command returns information on all nodes that share the same `$HOME/sqllib` directory, and can be issued from any of these nodes.

### Authorization

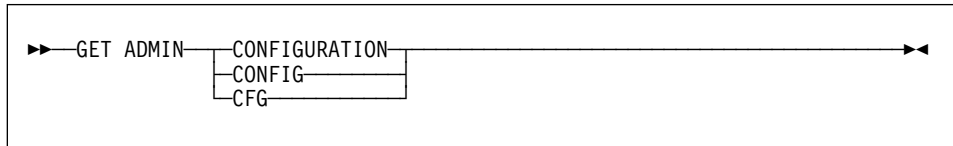
None

### Required Connection

None or instance. An instance attachment is not required to perform local DBM configuration operations, but is required to perform remote DBM configuration operations. To display the database manager configuration for a remote instance, it is necessary to first attach to that instance.

# GET ADMIN CONFIGURATION

## Command Syntax



## Command Parameters

None

## Example

The following is sample output from GET ADMIN CONFIGURATION:

```
Admin Server Configuration  
  
Node type = Database Server with local clients  
  
Database manager configuration release level          = 0x0800  
  
Diagnostic error capture level                        (DIAGLEVEL) = 3  
Diagnostic data directory path                       (DIAGPATH) =  
  
SYSADM group name                                   (SYSADM_GROUP) = BUILD  
SYSCTRL group name                                 (SYSCTRL_GROUP) =  
SYSMAINT group name                               (SYSMAINT_GROUP) =  
  
Database manager authentication                     (AUTHENTICATION) = SERVER  
  
Query heap size (4KB)                              (QUERY_HEAP_SZ) = 1200  
Discovery mode                                     (DISCOVER) = KNOWN  
Discovery communication protocols                   (DISCOVER_COMM) =
```

## Usage Notes

If an attachment to a remote instance (or a different local instance) exists, the admin configuration parameters for the attached server are returned; otherwise, the local admin configuration parameters are returned.

If an error occurs, the information returned is not valid. If the configuration file is invalid, an error message is returned. The user must install the database manager again to recover.

To set the configuration parameters to the default values shipped with the database manager, use "RESET ADMIN CONFIGURATION" on page 327.

## GET ADMIN CONFIGURATION

For more information about these parameters, see the *Administration Guide*.

### See Also

“RESET ADMIN CONFIGURATION” on page 327

“UPDATE ADMIN CONFIGURATION” on page 375.

## GET AUTHORIZATIONS

---

### GET AUTHORIZATIONS

Reports the authorities of the current user from values found in the database configuration file and the authorization system catalog view (SYSCAT.DBAUTH).

#### Authorization

None

#### Required Connection

Database. If implicit connect is enabled, a connection to the default database is established.

#### Command Syntax

```
▶▶ GET AUTHORIZATIONS ◀◀
```

#### Command Parameters

None

#### Example

The following is sample output from GET AUTHORIZATIONS:

```
Administrative Authorizations for Current User

Direct SYSADM authority           = NO
Direct SYSCTRL authority          = NO
Direct SYSMANT authority          = NO
Direct DBADM authority            = YES
Direct CREATETAB authority        = YES
Direct BINDADD authority          = YES
Direct CONNECT authority          = YES
Direct CREATE_NOT_FENC authority  = YES
Direct IMPLICIT_SCHEMA authority  = YES

Indirect SYSADM authority         = YES
Indirect SYSCTRL authority        = NO
Indirect SYSMANT authority        = NO
Indirect DBADM authority          = NO
Indirect CREATETAB authority      = YES
Indirect BINDADD authority        = YES
Indirect CONNECT authority        = YES
Indirect CREATE_NOT_FENC authority = NO
Indirect IMPLICIT_SCHEMA authority = YES
```

## GET AUTHORIZATIONS

### Usage Notes

Direct authorities are acquired by explicit commands that grant the authorities to a user ID. Indirect authorities are based on authorities acquired by the groups to which a user belongs.

**Note:** PUBLIC is a special group to which all users belong.

## GET CONNECTION STATE

---

### GET CONNECTION STATE

Displays the connection state. Possible states are:

- Connectable and connected
- Connectable and unconnected
- Unconnectable and connected
- Implicitly connectable (if implicit connect is available).

For more information about these connection states, see the *SQL Reference*.

This command also returns information about the database connection mode (SHARE or EXCLUSIVE), and the alias and name of the database to which a connection exists (if one exists).

#### Authorization

None

#### Required Connection

None

#### Command Syntax

```
▶▶—GET CONNECTION STATE—————▶▶
```

#### Command Parameters

None

#### Example

The following is sample output from GET CONNECTION STATE:

```
Database Connection State
Connection state      = Connectable and Connected
Connection mode      = SHARE
Local database alias = SAMPLE
Database name        = SAMPLE
```

#### Usage Notes

This command does not apply to type 2 connections (see “SET CLIENT” on page 353).



---

## GET DATABASE CONFIGURATION

Returns the values of individual entries in a specific database configuration file.

### Scope

This command returns information only for the node on which it is executed.

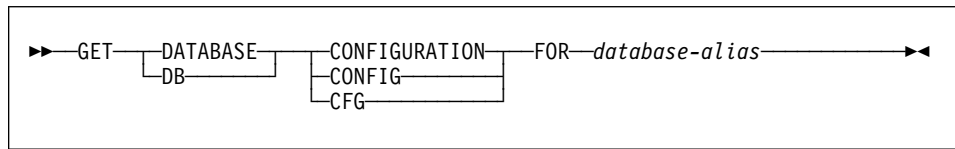
### Authorization

None

### Required Connection

Instance. An explicit attachment is not required. If the database is listed as remote, an instance attachment to the remote node is established for the duration of the command.

### Command Syntax



### Command Parameters

**FOR** *database-alias*

Specifies the alias of the database whose configuration is to be displayed.

### Example

#### Notes:

1. Output on different platforms may show small variations reflecting platform-specific parameters.
2. Parameters with keywords enclosed by parentheses can be changed using "UPDATE DATABASE CONFIGURATION" on page 379.
3. Fields that do not contain keywords are maintained by the database manager and cannot be updated.

The following is sample output from GET DATABASE CONFIGURATION (issued on AIX):

```

    Database Configuration for Database sample

    Database configuration release level      = 0x0800
    Database release level                   = 0x0800

    Database territory                       = En_US
    Database code page                       = 850
    Database code set                       = IBM-850
    Database country code                    = 1
  
```

## GET DATABASE CONFIGURATION

```

Directory object name                (DIR_OBJ_NAME) =
Discovery support for this database  (DISCOVER_DB) = ENABLE

Degree of parallelism                (DFT_DEGREE) = 1

Backup pending                       = NO

Database is consistent                = YES
Rollforward pending                  = NO
Restore pending                       = NO

Multi-page file allocation enabled    = NO

Log retain for recovery status        = NO
User exit for logging status         = NO

Default query optimization class      (DFT_QUERYOPT) = 5
Number of frequent values retained   (NUM_FREVALUES) = 10
Continue upon arithmetic exceptions   (DFT_SQLMATHWARN) = NO
Number of quantiles retained         (NUM_QUANTILES) = 20

Database heap (4KB)                  (DBHEAP) = 1200
Catalog cache size (4KB)             (CATALOGCACHE_SZ) = 64
Log buffer size (4KB)                (LOGBUFFSZ) = 8
Utilities heap size (4KB)            (UTIL_HEAP_SZ) = 5000
Buffer pool size (4KB)               (BUFFPAGE) = 1000
Extended storage segments size (4KB) (ESTORE_SEG_SZ) = 16000
Number of extended storage segments   (NUM_ESTORE_SEGS) = 0
Max storage for lock list (4KB)      (LOCKLIST) = 100

Max appl. control heap size (4KB)    (APP_CTL_HEAP_SZ) = 128

Sort list heap (4KB)                 (SORTHEAP) = 256
SQL statement heap (4KB)             (STMTHEAP) = 2048
Default application heap (4KB)       (APPLHEAPSZ) = 128
Package cache size (4KB)            (PCKCACHESZ) = (MAXAPPLS*8)
Statistics heap size (4KB)          (STAT_HEAP_SZ) = 4384

Interval for checking deadlock (ms)   (DLCHKTIME) = 10000
Percent. of lock lists per application (MAXLOCKS) = 10
Lock timeout (sec)                   (LOCKTIMEOUT) = -1

Changed pages threshold               (CHNGPGS_THRESH) = 60
Number of asynchronous page cleaners  (NUM_IOCLEANERS) = 1
Number of I/O servers                 (NUM_IOSERVERS) = 3
Index sort flag                       (INDEXSORT) = YES
Sequential detect flag                (SEQDETECT) = YES
Default prefetch size (4KB)          (DFT_PREFETCH_SZ) = 32

Default number of containers          = 1
Default tablespace extentsize (4KB)  (DFT_EXTENT_SZ) = 32

Max number of active applications     (MAXAPPLS) = 40
Average number of active applications (AVG_APPLS) = 1
Max DB files open per application     (MAXFILOP) = 64

Log file size (4KB)                  (LOGFILSIZ) = 1000
Number of primary log files           (LOGPRIMARY) = 3
Number of secondary log files         (LOGSECOND) = 2
Changed path to log files             (NEWLOGPATH) =
Path to log files                     = /home/smith/smith/NODE0000/SQL00011/SQLLOGDIR/
Next active log file                  =
First active log file                 =

Group commit count                    (MINCOMMIT) = 1
Percent log file reclaimed before soft ckcpt (SOFTMAX) = 100
Log retain for recovery enabled        (LOGRETAIN) = OFF

```

## GET DATABASE CONFIGURATION

User exit for logging enabled	(USEREXIT) = OFF
Auto restart enabled	(AUTORESTART) = ON
Index re-creation time	(INDEXREC) = SYSTEM (RESTART)
Default number of loadrec sessions	(DFT_LOADREC_SES) = 1
Recovery history retention (days)	(REC_HIS_RETENTN) = 366
ADSM management class	(ADSM_MGMTCLASS) =
ADSM node name	(ADSM_NODENAME) =
ADSM owner	(ADSM_OWNER) =
ADSM password	(ADSM_PASSWORD) =

These fields are identified below. Parameters whose name appears in lowercase are maintained by the database manager and cannot be updated. For more information about database configuration parameters, see the *Administration Guide*.

### ADSM\_MGMTCLASS

The ADSM management class specifies how the server should manage the backup versions or archive copies of the objects being backed up. The management class is assigned from the ADSM administrator. Once assigned, this parameter should be set to the management class name. When performing any ADSM backup, the database manager uses this parameter to pass the management class to ADSM.

### ADSM\_NODENAME

This parameter is used to override the default setting for the node name associated with the ADSM product. The node name is needed when restoring a database that was backed up to ADSM from another node.

### ADSM\_OWNER

This parameter is used to override the default setting for the owner associated with the ADSM product. The owner name is needed when restoring a database that was backed up to ADSM from another node.

### ADSM\_PASSWORD

This parameter is used to override the default setting for the password associated with the ADSM product. The password is needed when restoring a database that was backed up to ADSM from another node.

### APP\_CTL\_HEAP\_SZ

This parameter determines the maximum size, in 4KB pages, for the application control heap. The heap is required to share information among agents working on behalf of the same application at a node in an MPP or an SMP system. If complex applications are being run, or the MPP configuration has a large number of nodes, the size of this heap should be increased.

### APPLHEAPSZ

Specifies the size, in pages, of the application heap that is available for each individual agent.

### AUTORESTART

Indicates whether the database manager can automatically issue RESTART DATABASE on a connect if, for example, the database connection was disrupted, or the database was not terminated normally during the previous session.

OFF specifies that it must be done manually.

## GET DATABASE CONFIGURATION

ON specifies that the database manager does it automatically.

### **AVG\_APPLS**

Average number of active applications. Used by the SQL optimizer to help estimate how much buffer pool will be available for the chosen access plan at run time.

### **backup\_pending (Backup pending)**

NO specifies that the database is in a usable state.

YES specifies that an offline backup must be performed before the database can be used.

### **BUFFPAGE**

Specifies the size, in pages, of the buffer pool. The buffer pool is used to store and manipulate data read in from the database. This parameter is only used when a buffer pool's size has been explicitly set to -1 through either CREATE BUFFERPOOL, ALTER BUFFERPOOL, or "MIGRATE DATABASE" on page 282. The size of the buffer pool is normally controlled through SQL statements.

### **CATALOGCACHE\_SZ**

Controls the size, in pages, of the internal catalog cache (allocated from the *dbheap*), used by the SQL compiler to hold the packed descriptors for commonly referenced objects such as tables and constraints.

### **CHNGPGS\_THRESH**

Changed pages threshold. Used to specify the level (percentage) of changed pages at which the asynchronous page cleaners will be started, if they are not currently active.

### **codepage (Database code page)**

Specifies the code page of the database.

### **codeset (Database code set)**

Specifies the code set of the database.

### **COPYPROTECT (OS/2 only)**

Enables the copy-protect attribute.

### **country (Database country code)**

Specifies the country code of the database.

### **database\_consistent (Database is consistent)**

NO specifies that a transaction is pending, or some other task is pending on the database, and that the data is not consistent at this point.

YES specifies that all transactions have been committed or rolled back, and that the data is consistent.

### **database\_level (Database release level)**

Database release level. Specifies the release level of the database manager which can use the database.

### **DBHEAP**

Specifies the size, in pages, of the database heap that is used to hold control information on all open cursors accessing the database. Both *logbufsz* and *catalogcache\_sz* are allocated from the *dbheap*.

### **DFT\_DEGREE**

This parameter specifies the default value for the CURRENT DEGREE special register and the DEGREE bind option.

## GET DATABASE CONFIGURATION

### **DFT\_EXTENT\_SZ**

Default extent size of table spaces (in pages).

### **DFT\_LOADREC\_SES**

Default number of load recovery sessions. Specifies the default number of sessions that will be used during the recovery of a table load. Applicable only if roll-forward recovery is enabled.

### **DFT\_PREFETCH\_SZ**

Default prefetch size of table spaces (in pages).

### **DFT\_QUERYOPT**

The query optimization class is used to direct the optimizer to use different degrees of optimization when compiling SQL queries. This parameter provides additional flexibility by setting the default query optimization class used when neither the SET CURRENT QUERY OPTIMIZATION statement nor the QUERYOPT bind command are used.

### **DIR\_OBJ\_NAME**

Object name in DCE name space. The object name representing a database manager instance (or a database) in the directory. The concatenation of this value and the *dir\_path\_name* value yields a global name that uniquely identifies the database manager instance or database in the name space governed by the directory services specified in the *dir\_type* parameter.

### **DISCOVER\_DB**

This parameter can be set to DISABLE to prevent information about a database from being returned to a client when a discovery request is issued by the client against the server.

### **DLCHKTIME**

Time interval (in milliseconds) for checking deadlock. Defines the frequency at which the database manager checks for deadlocks among all the applications connected to a database.

### **ESTORE\_SEG\_SZ**

This parameter specifies the number of pages in each of the extended memory segments in the database. There are platform-dependent considerations when setting this configuration parameter.

### **INDEXREC**

Specifies when invalid indexes will be recreated. The default setting is SYSTEM, which uses the value of the database manager configuration parameter *indexrec*.

The possible output values are:

- SYSTEM(Access)
- SYSTEM(RESTART)
- ACCESS
- RESTART.

### **INDEXSORT**

Index sort flag. Indicates whether sorting of index keys will occur during index creation.

### **LOCKLIST**

Specifies the maximum storage, in pages, allocated to the lock list.

## GET DATABASE CONFIGURATION

### **LOCKTIMEOUT**

Specifies the number of seconds that an application will wait to obtain a lock.

### **LOGBUFSZ**

Specifies the number of pages used to buffer log records prior to writing them to disk. Allocated from *dbheap*.

### **LOGFILSIZ**

Specifies the amount of disk storage, in pages, allocated to log files used for data recovery. This parameter defines the size of each primary and secondary log file.

### **loghead (First active log file)**

Log head identification. Specifies the name of the log file containing the head of the active log. The next log record that is written will start at the head of the active log.

### **logpath (Path to log files)**

Location of log files. Contains the current path being used for logging purposes.

### **LOGPRIMARY**

Specifies the number of primary log files that can be used for database recovery.

### **LOGRETAIN**

Indicates whether the active log files are to be retained and become online archive log files for use in roll-forward recovery (log retention logging).

### **log\_retain\_status (Log retain for recovery status)**

Indicates whether log files are being retained for use in roll-forward recovery.

### **LOGSECOND**

Specifies the number of secondary log files that can be used for database recovery.

### **MAXAPPLS**

Specifies the maximum number of application programs (both local and remote) that can connect to the database at one time.

### **MAXFILOP**

Specifies the maximum number of database files that an application program can have open at one time.

### **MAXLOCKS**

Specifies the maximum percentage of the lock list that any one application program can use.

### **MINCOMMIT**

Specifies the number of SQL commits that can be grouped for a given database. Grouping SQL commits permits better control of the I/O and log activity when a commit is performed.

### **MULTIPAGE\_ALLOC**

Multi-page file allocation is used to improve insert performance. It applies to SMS table spaces only. If enabled, all SMS table spaces are affected: there is no selection possible for individual SMS table spaces.

## GET DATABASE CONFIGURATION

### **NEWLOGPATH**

Specifies an alternate path to the recovery log files for a database.

Since the *newlogpath* directory only accepts fully qualified directories, the absolute path must be specified.

### **nextactive (Next active log file)**

Specifies the name of the next recovery log file to be used for logging.

### **NUM\_ESTORE\_SEGS**

This parameter specifies the number of extended storage memory segments available for use by the database.

### **NUM\_FREQVALUES**

Number of frequent values retained. Used to specify the number of "most frequent values" that will be collected when the WITH DISTRIBUTION option is specified in "RUNSTATS" on page 350.

### **NUM\_IOCLEANERS**

Specifies the number of asynchronous page cleaners for a database.

### **NUM\_IOSERVERS**

Specifies the number of I/O servers for a database. I/O servers are used on behalf of the database agents to perform prefetch I/O and asynchronous I/O by utilities such as backup and restore.

### **NUM\_QUANTILES**

Number of quantiles for columns. Controls the number of quantiles that will be collected when the WITH DISTRIBUTION option is specified in "RUNSTATS" on page 350.

### **numsegs (Default number of containers)**

Determines the number of containers that will be created within the default SMS table spaces.

### **PCKCACHESZ**

Specifies the amount of memory to be used for caching packages and dynamic SQL statements.

If the value of this parameter is calculated at run time using other configuration parameters, the label (*calculated*) appears to the right of the internal value -1 in the output for "GET DATABASE CONFIGURATION" on page 175.

### **REC\_HIS\_RETENTN**

Recovery history retention period. Used to specify the number of days that historical information on backups is to be retained.

### **RESTORE\_PENDING**

This parameter indicates whether a RESTORE PENDING status exists in the database.

### **release (Database configuration release level)**

Specifies the release level of the database configuration file.

### **rollfwd\_pending (Rollforward pending)**

Indicates whether a roll-forward recovery procedure is required for the database.

The possible values are:

## GET DATABASE CONFIGURATION

- NO**  
Neither the database nor any table space is in roll-forward pending state.
- DATABASE**  
The database first needs to be rolled forward.
- TABLESPACES**  
One or more table spaces in the database requires roll-forward recovery.
- SEQDETECT**  
Indicates whether sequential detection for a database is to be enabled or disabled.
- SOFTMAX**  
This parameter is used to specify the frequency at which soft checkpoints are taken, and to specify the number of logs that are to be recovered after a crash.
- SORTHEAP**  
Specifies the number of private memory pages available for each sort in the application program.
- STAT\_HEAP\_SZ**  
Statistics heap size (in pages). Specifies the maximum size of the heap used in creating and collecting all table statistics when distribution statistics are being gathered.
- STMTHEAP**  
Specifies the heap size, in pages, that can be used for compiling SQL statements.
- territory (Database territory)**  
Specifies the territory of the database.
- USEREXIT**  
Indicates whether a user exit function for archiving or retrieving log files can be called the next time the database is opened.  
  
OFF specifies that a user exit function cannot be called.  
  
ON specifies that a user exit function can be called.
- user\_exit\_status (User exit for logging status)**  
OFF specifies that the user exit function cannot be called to store archive log files.  
  
ON specifies that the user exit function can be called to store archive log files.
- UTIL\_HEAP\_SZ**  
Utility heap size. Specifies the maximum amount of shared memory that can be used simultaneously by the backup, restore, and load utilities.

### Usage Notes

If an error occurs, the information returned is not valid. If the configuration file is invalid, an error message is returned. The database must be restored from a backup version.

To set the database configuration parameters to the database manager defaults, use "RESET DATABASE CONFIGURATION" on page 329.



## GET DATABASE CONFIGURATION

For more information about DB2's configuration parameters, see the *Administration Guide*.

### See Also

"RESET DATABASE CONFIGURATION" on page 329

"UPDATE DATABASE CONFIGURATION" on page 379.

# GET DATABASE MANAGER CONFIGURATION

---

## GET DATABASE MANAGER CONFIGURATION

Returns the values of individual entries in the database manager configuration file.

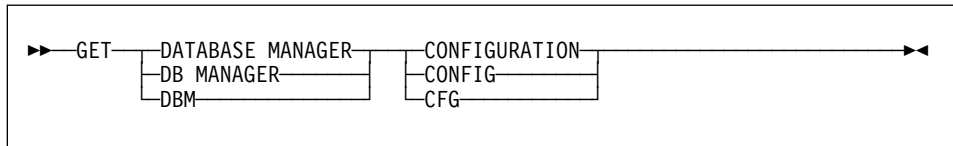
### Authorization

None

### Required Connection

None or instance. An instance attachment is not required to perform local DBM configuration operations, but is required to perform remote DBM configuration operations. To display the database manager configuration for a remote instance, it is necessary to first attach to that instance.

### Command Syntax



### Command Parameters

None

### Example

**Note:** Both node type and platform determine which configuration parameters are listed.

The following is sample output from GET DATABASE MANAGER CONFIGURATION (issued on AIX):

```
Database Manager Configuration

Node type = Database Server with local and remote clients

Database manager configuration release level          = 0x0800

CPU speed (millisec/instruction)                    (CPUSPEED) = 4.000000e-05

Maximum query degree of parallelism (MAX_QUERYDEGREE) = ANY
Enable intra-partition parallelism (INTRA_PARALLEL) = NO

Max number of concurrently active databases (NUMDB) = 8
Transaction processor monitor name (TP_MON_NAME) =

Default charge-back account (DFT_ACCOUNT_STR) =
Java Development Kit 1.1 installation path (JDK11_PATH) = /home/smith/jdk11

Diagnostic error capture level (DIAGLEVEL) = 3
Diagnostic data directory path (DIAGPATH) =
```

## GET DATABASE MANAGER CONFIGURATION

```
Default database monitor switches
  Buffer pool                (DFT_MON_BUFPOOL) = OFF
  Lock                      (DFT_MON_LOCK) = OFF
  Sort                      (DFT_MON_SORT) = OFF
  Statement                 (DFT_MON_STMT) = OFF
  Table                     (DFT_MON_TABLE) = OFF
  Unit of work              (DFT_MON_UOW) = OFF

Database monitor SQL statement size (bytes) (SQLSTMTSZ) = 256

SYSADM group name          (SYSADM_GROUP) = BUILD
SYSCTRL group name        (SYSCTRL_GROUP) =
SYSMAINT group name       (SYSMAINT_GROUP) =

Database manager authentication (AUTHENTICATION) = CLIENT
Trust all clients         (TRUST_ALLCLNTS) = YES
Trusted client authentication (TRUST_CLNTAUTH) = CLIENT

Default database path      (DFTDBPATH) = /home/smith

Database monitor heap size (4KB) (MON_HEAP_SZ) = 48
UDF shared memory set size (4KB) (UDF_MEM_SZ) = 256

Backup buffer default size (4KB) (BACKBUFSZ) = 1024
Restore buffer default size (4KB) (RESTBUFSZ) = 1024

Sort heap threshold (4KB) (SHEAPTHRES) = 20000

Directory cache support    (DIR_CACHE) = YES

Java Virtual Machine heap size (4KB) (JAVA_HEAP_SZ) = 512

Application support layer heap size (4KB) (ASLHEAPSZ) = 3000
Max requester I/O block size (bytes) (RQRIOBLK) = 32767
Query heap size (4KB) (QUERY_HEAP_SZ) = 15000
DRDA services heap size (4KB) (DRDA_HEAP_SZ) = 128

Priority of agents         (AGENTPRI) = SYSTEM
Max number of existing agents (MAXAGENTS) = 200
Agent pool size           (NUM_POOLAGENTS) = 4 (calculated)
Initial number of agents in pool (NUM_INITAGENTS) = 0
Max number of coordinating agents (MAX_COORDAGENTS) = MAXAGENTS
Max no. of concurrent coordinating agents (MAXCAGENTS) = MAX_COORDAGENTS

Keep DARI process         (KEEPDARI) = YES
Max number of DARI processes (MAXDARI) = MAX_COORDAGENTS

Index re-creation time    (INDEXREC) = RESTART

Transaction manager database name (TM_DATABASE) = 1ST_CONN
Transaction resync interval (sec) (RESYNC_INTERVAL) = 180

SPM name                  (SPM_NAME) =
SPM log size              (SPM_LOG_FILE_SZ) = 256
SPM resync agent limit    (SPM_MAX_RESYNC) = 20

TCP/IP Service name       (SVCENAME) =
APPC Transaction program name (TPNAME) =
```

## GET DATABASE MANAGER CONFIGURATION

Discovery mode	(DISCOVER) = SEARCH
Discovery communication protocols	(DISCOVER_COMM) =
Discover server instance	(DISCOVER_INST) = ENABLE
IPX/SPX File server name	(FILESERVER) =
IPX/SPX DB2 server object name	(OBJECTNAME) =
IPX/SPX Socket number	(IPX_SOCKET) = 879E
Directory services type	(DIR_TYPE) = NONE
Directory path name	(DIR_PATH_NAME) = /./:/subsys/database/
Directory object name	(DIR_OBJ_NAME) =
Routing information object name	(ROUTE_OBJ_NAME) =
Default client comm. protocols	(DFT_CLIENT_COMM) =
No. of int. communication buffers(4KB)	(FCM_NUM_BUFFERS) = 4096
Number of FCM request blocks	(FCM_NUM_RQB) = 2048
Number of FCM connection entries	(FCM_NUM_CONNECT) = (FCM_NUM_RQB * 0.75)
Number of FCM message anchors	(FCM_NUM_ANCHORS) = (FCM_NUM_RQB * 0.75)

These fields are identified as follows:

### **AGENT\_STACK\_SZ (OS/2 only)**

The amount of memory allocated and committed by the operating system for each agent. This parameter specifies the number of pages for each agent stack on the server.

### **AGENTPRI**

Execution priority assigned to database manager processes and threads on a particular machine.

### **ASLHEAPSZ**

Size (in pages) of the memory shared between a local client application and a database manager agent.

### **AUTHENTICATION**

Determines how and where authentication of a user takes place. A value of CLIENT indicates that all authentication takes place at the client. If the value is SERVER, the user ID and password are sent from the client to the server so that authentication can take place at the server.

### **BACKBUFSZ**

Size (in pages) of the buffer used when backing up the database, if the buffer size is not specified when calling the backup utility.

### **CONN\_ELAPSE (MPP only)**

This parameter specifies the number of seconds within which a TCP/IP connection is to be established between two nodes. If the attempt completes within the time specified by this parameter, communications are established. If it fails, another attempt is made to establish communications. If the connection is attempted the number of times specified by the MAX\_CONNRETRIES parameter and always times out, an error is returned.

### **CPUSPEED**

CPU speed (in milliseconds per instruction) used by the SQL optimizer to estimate the cost of performing certain operations. The value of this parameter is set automatically when the database manager is installed, but

## GET DATABASE MANAGER CONFIGURATION

can be modified to model a production environment on a test system, or to assess the impact of upgrading hardware.

### **DFT\_ACCOUNT\_STR**

Default accounting string.

### **DFT\_CLIENT\_ADPT (OS/2 only)**

This parameter defines the default client adapter number for the NETBIOS protocol whose server nname is extracted from DCE Directory Services. This parameter can only be used with DCE.

### **DFT\_CLIENT\_COMM (DCE only)**

Specifies the communication protocols that the client applications on a specific instance can use for remote connections.

### **DFT\_MON\_BUFPOOL**

Default value of the snapshot monitor's buffer pool switch.

### **DFT\_MON\_LOCK**

Default value of the snapshot monitor's lock switch.

### **DFT\_MON\_SORT**

Default value of the snapshot monitor's sort switch.

### **DFT\_MON\_STMT**

Default value of the snapshot monitor's statement switch.

### **DFT\_MON\_TABLE**

Default value of the snapshot monitor's table switch.

### **DFT\_MON\_UOW**

Default value of the snapshot monitor's unit of work (UOW) switch.

### **DFTDBPATH**

Default database path. If no path is specified when a database is created, the database is created on the path specified by this parameter.

### **DIAGLEVEL**

Diagnostic error capture level determines the severity of diagnostic errors recorded in the error log file (db2diag.log).

### **DIAGPATH**

The fully qualified path for DB2 diagnostic information.

### **DIR\_CACHE**

Directory cache support. If set to YES, database, node, and DCS directory files are cached in memory. This reduces connect costs by eliminating directory file I/O, and minimizing the directory searches required to retrieve directory information.

### **DIR\_OBJ\_NAME**

Object name in DCE name space. The object name representing a database manager instance (or a database) in the directory. The concatenation of this value and the *dir\_path\_name* value yields a global name that uniquely identifies the database manager instance or database in the name space governed by the directory services specified in the *dir\_type* parameter.

### **DIR\_PATH\_NAME**

Directory path name in DCE name space. The unique name of the database manager instance in the global name space is made up of this value and the value in the *dir\_obj\_name* parameter.

## GET DATABASE MANAGER CONFIGURATION

### **DIR\_TYPE**

Directory services type. Indicates whether the database manager instance uses the DCE global directory services.

### **DISCOVER**

This parameter defines the type of discovery request supported on a client or server. Discovery requests can be issued from the client configuration assistant or from control center tools. Specify **SEARCH** to support search discovery, in which the DB2 client searches the network for DB2 databases. Specify **KNOWN** to support known discovery, in which the discovery request is issued against the administration server specified by the user. Specify **DISABLE** to disable the client or server from supporting any type of discovery request.

### **DISCOVER\_COMM**

This parameter defines the communications protocols that clients use to issue search discovery requests, and servers use to listen for search discovery requests. More than one protocol can be specified, separated by commas, or the parameter can be left blank. Supported protocols are TCP/IP and NETBIOS.

### **DISCOVER\_INST**

This parameter enables or disables client discovery of an instance.

### **DOS\_RQRIOBLK**

DOS requester I/O block size. Applicable only on DOS clients, including DOS clients running under OS/2. This parameter controls the size of the I/O blocks that are allocated on the client and the server.

### **DRDA\_HEAP\_SZ**

Specifies the size, in pages, of the DRDA heap. This heap is used by the DRDA AS.

### **FCM\_NUM\_ANCHORS**

This parameter specifies the number of FCM message anchors. Agents use the message anchors to send messages among themselves.

### **FCM\_NUM\_BUFFERS**

This parameter specifies the number of 4KB buffers that are used for internal communications (messages) among the nodes in an instance.

### **FCM\_NUM\_CONNECT**

This parameter specifies the number of FCM connection entries. Agents use connection entries to pass data among themselves.

### **FCM\_NUM\_RQB**

This parameter specifies the number of FCM request blocks. Request blocks are the media through which information is passed between the FCM daemon and an agent.

### **FILESERVER**

IPX/SPX file server name. Specifies the name of the Novell NetWare file server where the internetwork address of the database manager server instance is registered.

**Note:** The following characters are not valid: / \ ; , \* ?

## GET DATABASE MANAGER CONFIGURATION

### INDEXREC

Specifies when invalid database indexes should be recreated. This parameter is used if the database configuration parameter *indexrec* is set to SYSTEM.

The possible output values are:

- ACCESS
- RESTART.

### INTRA\_PARALLEL

This parameter specifies whether the database manager can use intra-partition parallelism.

In a symmetric multiprocessor (SMP) environment, the default for this parameter is YES. In a non-SMP environment, the default for this parameter is NO. This parameter can be used on both partitioned and non-partitioned database systems. Some of the operations that can take advantage of parallel performance improvements when the value of this parameter is YES include database queries and index creation.

### IPX\_SOCKET

IPX/SPX socket number. Specifies a "well-known" socket number and represents the connection end point in a DB2 server's IPX/SPX internetwork address.

### KEEPDARI

Indicates whether to keep a DARI process after each DARI call. If NO, a new DARI process is created and terminated for each DARI invocation. If YES, a DARI process is reused for subsequent DARI calls, and is terminated only when the associated user application exits.

### MAX\_CONNRETRIES (MPP only)

If an attempt to establish communication between two nodes fails because the value specified by the *CONN\_ELAPSE* parameter is reached (for example, the attempt to establish TCP/IP communication times out), *MAX\_CONNRETRIES* specifies the number of connection retries that can be made to a node. If the value specified for this parameter is exceeded, an error is returned.

### MAX\_COORDAGENTS

This parameter determines the maximum number of coordinating agents that can exist at one time on a node.

### MAX\_QUERYDEGREE

This parameter specifies the maximum degree of parallelism used for any SQL statement executing on this instance of the database manager. An SQL statement will not use more than this number of parallel operations when the statement is executed. For a multi-node system, this parameter applies to the degree of parallelism within a single node.

### MAX\_TIME\_DIFF (MPP only)

Each node has its own system clock. This parameter specifies the maximum time difference, in minutes, that is permitted among the nodes listed in the *db2nodes.cfg* file.

## GET DATABASE MANAGER CONFIGURATION

### MAXAGENTS

Maximum number of database manager agents that can exist simultaneously on a node, regardless of which database is being used.

### MAXCAGENTS

Maximum number of database manager agents that can be concurrently executing a database manager transaction. Cannot exceed the value of *maxagents*.

### MAXDARI

Maximum number of DARI processes that can reside at the database server. Cannot exceed the value of *maxagents*.

### MAXTOTFILOP (OS/2 only)

Maximum number of files open per application. Defines the total database and application file handles that can be used by a specific process connected to a database.

### MIN\_PRIV\_MEM (OS/2 only)

Minimum committed private memory. Specifies the number of pages that the database server process will reserve as private virtual memory when a database manager instance is started (**db2start**).

### MON\_HEAP\_SZ

Database system monitor heap size. Specifies the amount (in 4KB pages) of memory to allocate for database system monitor data.

### NNAME (OS/2 only)

Name of the node or workstation. Database clients use *nname* to access database server workstations using NetBIOS. If the database server workstation changes the name specified in *nname*, all clients that access the database server workstation must catalog it again and specify the new name.

### nodetype (Node type)

Indicates whether the node is configured as a server with local and remote clients, a client, or a server with local clients.

### NUM\_INITAGENTS

This parameter determines the initial number of agents that are created in the agent pool when the database manager is started.

### NUM\_POOLAGENTS

This parameter specifies the size to which the agent pool is allowed to grow. The agent pool contains both idle agents (as in DB2/6000 Version 2), and MPP and SMP associated subagents. If more agents are created, they will be terminated and not return to the pool when they are finished executing.

If the value of this parameter is calculated at run time using other configuration parameters, the label (calculated) appears to the right of the value shown in the output for "GET DATABASE MANAGER CONFIGURATION" on page 184. If -1 (calculated) is shown in the output, the request was issued from a client, and the value was not available.

The obsolete database manager configuration parameter *max\_idleagents* can still be updated through "UPDATE DATABASE MANAGER



## GET DATABASE MANAGER CONFIGURATION

CONFIGURATION" on page 381, and is interpreted as an update to *num\_poolagents*.

### **NUMDB**

Maximum number of local databases that can be concurrently active (that is, have applications connected to them).

### **OBJECTNAME**

This parameter represents the database manager server instance as an object on the NetWare file server, where the server's IPX/SPX address is stored and retrieved. The value must be entered in uppercase. The value must be unique on the NetWare file server, and it is recommended that it be unique across the IPX/SPX network.

**Note:** The following characters are not valid: / \ ; , \* ?

### **PRIV\_MEM\_THRESH (OS/2 only)**

Private memory threshold. Sets a threshold below which a server will not release the memory associated with a client when that client's connection is terminated.

### **QUERY\_HEAP\_SZ**

Maximum amount of memory (in pages) that can be allocated for the query heap. A query heap is used to store each query in the agent's private memory.

### **release (Database manager configuration release level)**

Release level of the configuration file.

### **RESTBUFSZ**

Size (in 4KB pages) of the buffer used when restoring the database, if the buffer size is not specified when calling the restore utility.

### **RESYNC\_INTERVAL**

Time interval (in seconds) after which a Transaction Manager (TM) or a Resource Manager (RM) retries the recovery of any outstanding indoubt transactions found in the TM or the RM. Applicable when transactions are running in a distributed unit of work (DUOW) environment.

### **ROUTE\_OBJ\_NAME (DCE only)**

Routing information object name. Specifies the name of the default routing information object entry that will be used by all client applications attempting to access a DRDA server.

### **RQRIOBLK**

Client I/O block size. Specifies the size (in bytes) of the communication buffer between remote applications and their database agents on the database server.

### **SHEAPTHRESH**

Limit on the total amount of memory (in pages) available for sorting across the entire instance.

### **SPM\_NAME**

This parameter identifies the name of the Sync Point Manager (SPM) instance to the database manager. The *spm\_name* must be defined in the system database directory and, if remote, in the node directory.

## GET DATABASE MANAGER CONFIGURATION

### **SPM\_LOG\_FILE\_SZ**

This parameter identifies the Sync Point Manager (SPM) log file size in 4KB pages. The log file is contained in the `spmlog` sub-directory under `sql1lib` and is created the first time SPM is started.

### **SPM\_MAX\_RESYNC**

This parameter identifies the number of simultaneous agents that can perform resync operations.

### **SQLSTMTSZ**

Maximum amount of dynamic SQL statement text (in bytes) that will be returned by the database system monitor.

### **SS\_LOGON (OS/2 only)**

By accepting the default for this parameter, a LOGON user ID and password are required before issuing a DB2START or DB2STOP.

### **START\_STOP\_TIME (MPP only)**

This parameter specifies the time, in minutes, within which all nodes must respond to "START DATABASE MANAGER" on page 361, "STOP DATABASE MANAGER" on page 365, or "ADD NODE" on page 89.

### **SVCENAME**

The name used to update the database manager configuration file at the server. This value must be the same as the Connection Service name specified in the `services` file.

**Note:** It is not recommended that this parameter, which is set by the installation program, be modified by the user. The administration server service name is set to use the DB2 registered TCP/IP port (523).

### **SYSADM\_GROUP**

Defines the group name with system administration (*sysadm*) authority for the database manager instance. This is the highest level of authority within the database manager, and controls all database objects.

### **SYSCTRL\_GROUP**

Defines the group name with system control (*sysctrl*) authority for the database manager instance. This level has privileges allowing operations affecting system resources, but not allowing direct access to data.

### **SYSMAINT\_GROUP**

Defines the group name with system maintenance (*sysmaint*) authority for the database manager instance. This level has authority allowing maintenance operations on all databases associated with an instance, but not allowing direct access to data.

### **TM\_DATABASE**

Name of the transaction manager (TM) database for each DB2 instance.

### **TP\_MON\_NAME**

Name of the transaction processing (TP) monitor product being used.

### **TPNAME**

Name of the remote transaction program that the database client must use when it issues an allocate request to the database manager instance using the APPC communication protocol.

## GET DATABASE MANAGER CONFIGURATION

### TRUST\_ALLCLNTS

This parameter and the TRUST\_CLNTAUTH parameter are used to determine where users are validated to the database environment. By accepting the default for this parameter, all clients are treated as trusted clients. This means a level of security is available at the client, and that users can be validated at the client.

### TRUST\_CLNAUTH

This parameter and the TRUST\_ALLCLNTS parameter are used to determine where users are validated to the database environment. By accepting the default for this parameter, all users of trusted clients are validated at the client.

### UDF\_MEM\_SZ

For a fenced user defined function (UDF), specifies the default allocation for memory to be shared between the database process and the UDF. For an unfenced process, specifies the size of the private memory set. In both cases, this memory is used to pass data to a UDF and back to a database.

## Usage Notes

If an attachment to a remote instance (or a different local instance) exists, the database manager configuration parameters for the attached server are returned; otherwise, the local database manager configuration parameters are returned.

If an error occurs, the information returned is invalid. If the configuration file is invalid, an error message is returned. The user must install the database manager again to recover.

To set the configuration parameters to the default values shipped with the database manager, use "RESET DATABASE MANAGER CONFIGURATION" on page 331.

For more information about these parameters, see the *Administration Guide*.

## See Also

"RESET DATABASE MANAGER CONFIGURATION" on page 331

"UPDATE DATABASE MANAGER CONFIGURATION" on page 381.

# GET DATABASE MANAGER MONITOR SWITCHES

---

## GET DATABASE MANAGER MONITOR SWITCHES

Displays the status of the database system monitor switches. Monitor switches instruct the database system manager to collect database activity information. Each application using the database system monitor interface has its own set of monitor switches (see “GET MONITOR SWITCHES” on page 197). A database manager-level switch is on when any of the monitoring applications has turned it on. This command is used to determine if the database system monitor is currently collecting data for any monitoring application.

### Authorization

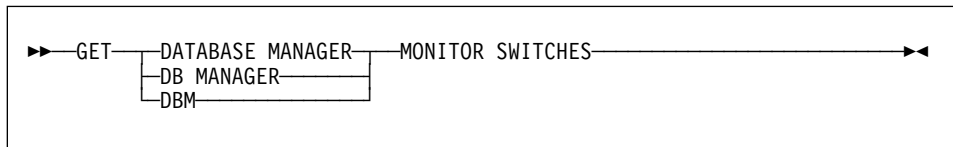
One of the following:

*sysadm*  
*sysctrl*  
*sysmaint*

### Required Connection

Instance. To display the settings for a remote instance, or for a different local instance, it is necessary to first attach to that instance.

### Command Syntax



### Command Parameters

None

### Example

The following is sample output from GET DATABASE MANAGER MONITOR SWITCHES:

```
DBM System Monitor Information Collected

Buffer Pool Activity Information (BUFFERPOOL) = ON   06-11-1997 10:11:01.738377
Lock Information (LOCK) = OFF
Sorting Information (SORT) = ON   06-11-1997 10:11:01.738400
SQL Statement Information (STATEMENT) = OFF
Table Activity Information (TABLE) = OFF
Unit of Work Information (UOW) = ON   06-11-1997 10:11:01.738353
```

## GET DATABASE MANAGER MONITOR SWITCHES

### Usage Notes

The six recording switches (BUFFERPOOL, LOCK, SORT, STATEMENT, TABLE, and UOW) are off by default, but may be switched on using "UPDATE MONITOR SWITCHES" on page 383. If a particular switch is on, this command also displays the time stamp for when the switch was turned on.

For a summary of all database monitor data elements and monitoring groups, see the *System Monitor Guide and Reference*.

### See Also

"GET MONITOR SWITCHES" on page 197

"GET SNAPSHOT" on page 199

"RESET MONITOR" on page 333

"UPDATE MONITOR SWITCHES" on page 383.

## GET INSTANCE

---

### GET INSTANCE

Returns the value of the **DB2INSTANCE** environment variable.

#### Authorization

None

#### Required Connection

None

#### Command Syntax

```
▶ GET INSTANCE ▶
```

#### Command Parameters

None

#### Example

The following is sample output from GET INSTANCE:

```
The current database manager instance is: smith
```

## GET MONITOR SWITCHES

Displays the status of the database system monitor switches for the current session. Monitor switches instruct the database system manager to collect database activity information. Each application using the database system monitor interface has its own set of monitor switches. This command displays them. To display the database manager-level switches, use “GET DATABASE MANAGER MONITOR SWITCHES” on page 194.

### Authorization

One of the following:

*sysadm*  
*sysctrl*  
*sysmaint*

### Required Connection

Instance. To display the settings for a remote instance, or for a different local instance, it is necessary to first attach to that instance.

### Command Syntax

```
▶▶ GET MONITOR SWITCHES ◀◀
```

### Command Parameters

None

### Example

The following is sample output from GET MONITOR SWITCHES:

```

Monitor Recording Switches

Buffer Pool Activity Information (BUFFERPOOL) = ON 02-20-1997 16:04:30.070073
Lock Information (LOCK) = OFF
Sorting Information (SORT) = OFF
SQL Statement Information (STATEMENT) = ON 02-20-1997 16:04:30.070073
Table Activity Information (TABLE) = OFF
Unit of Work Information (UOW) = ON 02-20-1997 16:04:30.070073
    
```

### Usage Notes

The six recording switches (BUFFERPOOL, LOCK, SORT, STATEMENT, TABLE, and UOW) are off by default, but may be switched on using “UPDATE MONITOR

## GET MONITOR SWITCHES

SWITCHES” on page 383. If a particular switch is on, this command also displays the time stamp for when the switch was turned on.

For a summary of all database monitor data elements and monitoring groups, see the *System Monitor Guide and Reference*.

### See Also

“GET DATABASE MANAGER MONITOR SWITCHES” on page 194

“GET SNAPSHOT” on page 199

“RESET MONITOR” on page 333

“UPDATE MONITOR SWITCHES” on page 383.



## GET SNAPSHOT

Collects database manager status information and returns it to a user-allocated data buffer. The information returned represents a *snapshot* of the database manager operational status at the time the command was issued.

### Scope

This command can be invoked from any node in the `db2nodes.cfg` file. It acts only on that node or partition.

### Authorization

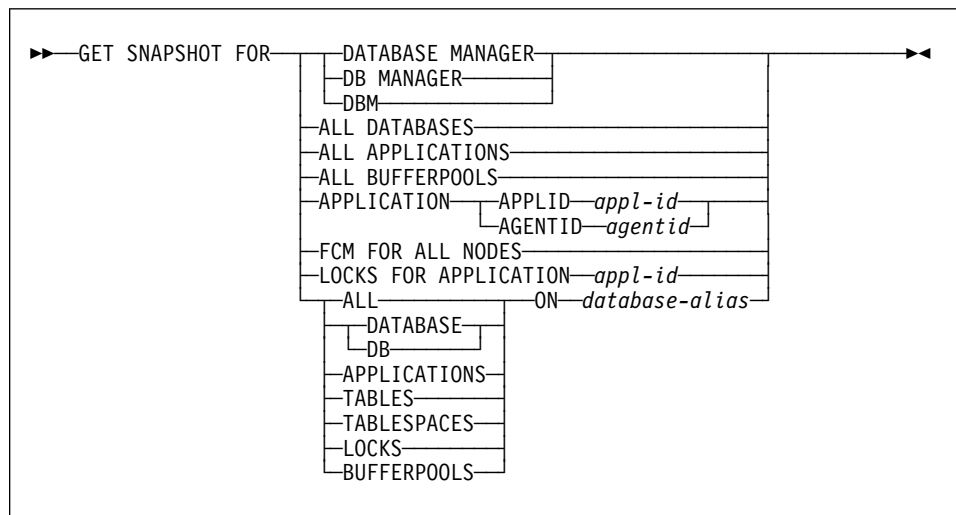
One of the following:

*sysadm*  
*sysctrl*  
*sysmaint*

### Required Connection

Instance. To obtain a snapshot of a remote instance, it is necessary to first attach to that instance.

### Command Syntax



**Note:** The monitor switches must be turned on to get some statistics (see “UPDATE MONITOR SWITCHES” on page 383).

# GET SNAPSHOT

## Command Parameters

### **DATABASE MANAGER**

Provides statistics for the active database manager instance.

### **ALL DATABASES**

Provides general statistics for all active databases on the current node.

### **ALL APPLICATIONS**

Provides information about all active applications that are connected to a database on the current node.

### **ALL BUFFERPOOLS**

Provides information about buffer pool activity for all active databases.

### **APPLICATION APPLID** *appl-id*

Provides information only about the application whose ID is specified. To get a specific application ID, use “LIST APPLICATIONS” on page 225.

### **APPLICATION AGENTID** *agentid*

Provides information only about the application whose agent ID is specified. The *agent-id* is a 32-bit number that uniquely identifies an application that is currently running. Use “LIST APPLICATIONS” on page 225 to get a specific agent ID.

### **FCM FOR ALL NODES**

Provides FCM statistics for all nodes.

### **LOCKS FOR APPLICATION** *appl-id*

Provides information about all locks held by the specified application.

### **ALL ON** *database-alias*

Provides general statistics and information about all applications, tables, table spaces, buffer pools, and locks for a specified database.

### **DATABASE ON** *database-alias*

Provides general statistics for a specified database.

### **APPLICATIONS ON** *database-alias*

Provides information about all applications connected to a specified database.

### **TABLES ON** *database-alias*

Provides information about tables in a specified database. This will include only those tables that have been accessed since the TABLE recording switch was turned on.

### **TABLESPACES ON** *database-alias*

Provides information about table spaces for a specified database.

### **LOCKS ON** *database-alias*

Provides information about every lock held by each application connected to a specified database.

### **BUFFERPOOLS ON** *database-alias*

Provides information about buffer pool activity for the specified database.

## Examples

In the following sample output listings, some of the information may not be available, depending on whether or not the appropriate database system monitor recording switch is turned on (see “UPDATE MONITOR SWITCHES” on page 383). If the information is unavailable, Not Collected appears in the output.

## GET SNAPSHOT

For more information about the fields displayed in the following output listings, see the *System Monitor Guide and Reference*.

The following is typical output resulting from a request for database manager information:

```
Database Manager Snapshot

Node type                = Database Server with local and remote clients
Instance name           = user1
Database manager status  = Active

Product name            =
Product identification   =
Service level           =

Sort heap allocated     = 0
Post threshold sorts    = 0
Piped sorts requested   = 3
Piped sorts accepted    = 3

Start Database Manager timestamp = 04-04-1997 10:54:32.357375
Last reset timestamp    = 04-04-1997 14:28:54.799017
Snapshot timestamp     = 04-06-1997 14:27:19.996209

Remote connections to db manager = 0
Remote connections executing in db manager = 0
Local connections       = 1
Local connections executing in db manager = 0
Active local databases  = 1

High water mark for agents registered = 2
High water mark for agents waiting for a token = 1
Agents registered       = 2
Agents waiting for a token = 0
Idle agents             = 0

Committed private Memory (Bytes) = 12435456

Buffer Pool Activity Information (BUFFERPOOL) = ON 04-04-1997 14:29:42
Lock Information (LOCK) = ON 04-04-1997 14:29:42
Sorting Information (SORT) = ON 04-04-1997 14:29:42
SQL Statement Information (STATEMENT) = ON 04-04-1997 14:29:42
Table Activity Information (TABLE) = ON 04-04-1997 14:29:42
Unit of Work Information (UOW) = ON 04-04-1997 14:29:42

Agents assigned from pool = 5
Agents created from empty pool = 2
Agents stolen from another application = 0
High water mark for coordinating agents = 2
Max agents overflow = 0
```

## GET SNAPSHOT

The following is typical output resulting from a request for database information:

### Database Snapshot

```
Database name                = SAMPLE
Database path                = /home/user1/user1/NODE0000/SQL00011/
Input database alias         = SAMPLE
Database status              = Active
Catalog node number         = 0

Catalog network node name    =
Operating system running at database server= AIX
Location of the database     = Remote

Locks held currently        = 7
Lock waits                   = 0
Time database waited on locks (ms) = 0
Lock list memory in use (Bytes) = 1400
Deadlocks detected          = 0
Lock escalations            = 0
Exclusive lock escalations  = 0
Current applications waiting on locks = 0
Lock Timeouts               = 0

Total sort heap allocated    = 0
Total sorts                  = 3
Total sort time (ms)        = 1
Sort overflows               = 0
Active sorts                 = 0

Data pages copied to extended storage = 0
Index pages copied to extended storage = 0
Data pages copied from extended storage = 0
Index pages copied from extended storage = 0

Buffer pool data logical reads = 32
Buffer pool data physical reads = 13
Asynchronous pool data page reads = 0
Buffer pool data writes      = 0
Asynchronous pool data page writes = 0
Buffer pool index logical reads = 55
Buffer pool index physical reads = 23
Asynchronous pool index page reads = 0
Buffer pool index writes     = 0
Asynchronous pool index page writes = 0
Total buffer pool read time (ms) = 364
Total buffer pool write time (ms) = 0
Total elapsed asynchronous read time = 0
Total elapsed asynchronous write time = 0
Asynchronous read requests   = 0
LSN Gap cleaner triggers     = 0
Dirty page steal cleaner triggers = 0
Dirty page threshold cleaner triggers = 0
Time waited for prefetch (ms) = 0

Direct reads                 = 34
Direct writes                 = 0
Direct read requests         = 4
Direct write requests        = 0
```

## GET SNAPSHOT

```
Direct reads elapsed time (ms)           = 1
Direct write elapsed time (ms)          = 0

Database files closed                   = 0

Commit statements attempted             = 1
Rollback statements attempted           = 0
Dynamic statements attempted            = 13
Static statements attempted             = 1
Failed statement operations             = 0
Select SQL statements executed           = 1
Update/Insert/Delete statements        = 0
executed

DDL statements executed                  = 0
Internal automatic rebinds              = 0
Internal rows deleted                   = 0
Internal rows inserted                  = 0
Internal rows updated                   = 0
Internal commits                        = 1
Internal rollbacks                      = 0
Internal rollbacks due to deadlock      = 0

Rows deleted                            = 0
Rows inserted                           = 0
Rows updated                            = 0
Rows selected                           = 8

Binds/precompiles attempted            = 0

First database connect timestamp         = 04-04-1997 14:29:55.197659
Last reset timestamp                    =
Last backup timestamp                   =
Snapshot timestamp                      = 04-04-1997 14:32:14.151875

High water mark for connections         = 1
High water mark for database heap       = 652811
Application connects                    = 1
Secondary connects total                 = 0
Applications connected currently         = 1
Appls. executing in db manager          = 0
currently

Maximum secondary log space used (Bytes) = 0
Maximum total log space used (Bytes)    = 0
Secondary logs allocated currently       = 0
Log pages read                          = 0
Log pages written                       = 0

Package cache lookups                   = 1
Package cache inserts                   = 1
Application section lookups              = 13
Application section inserts              = 1

Catalog cache lookups                   = 2
Catalog cache inserts                   = 2
Catalog cache overflows                  = 0
Catalog cache heap full                  = 0

Agents associated with appls.            = 1
Maximum agents associated with appls.    = 1
```

## GET SNAPSHOT

```
Maximum coordinating agents      = 0
Agents waiting on locks         = 0
```

The following is typical output resulting from a request for application information (by specifying either an application ID, an agent ID, all applications, or all applications on a database):

### Application Snapshot

```
Application handle                = 5
Application status                = UOW Waiting
Status change time               = 04-04-1997 14:31:46.930243
Application code page            = 850
Application country code         = 1
DUOW correlation token           = *LOCAL.user1.970404192955
Application name                  = db2bp_32
Application ID                    = *LOCAL.user1.970404192956
Sequence number                  = 0001
Connection request start timestamp = 04-04-1997 14:29:55.197659
Connect request completion timestamp = 04-04-1997 14:29:55.726359
Application idle time            = 28 seconds
Authorization ID                  = USER1
Execution ID                      = user1
Configuration NNAME of client    =
Client database manager product ID = SQL03010
Process ID of client application = 18660
Platform of client application    = AIX
Communication protocol of client = Local Client
Database name                     = SAMPLE
Database path                     = /home/user1/user1/NODE0000/SQL00011/
Client database alias             = sample
Input database alias              = SAMPLE
Last reset timestamp             =
Snapshot timestamp                = 04-04-1997 14:32:14.151875
The highest authority level granted =
  Direct DBADM authority
  Direct CREATETAB authority
  Direct BINDADD authority
  Direct CONNECT authority
  Direct CREATE_NOT_FENC authority
  Direct IMPLICIT_SCHEMA authority
  Indirect SYSADM authority
  Indirect CREATETAB authority
  Indirect BINDADD authority
  Indirect CONNECT authority
  Indirect IMPLICIT_SCHEMA authority
Coordinating node number         = 0
Current node number              = 0
Coordinator agent process or thread ID = 75340
Agents working for the application = 1
Agents stolen                     = 0
Agents waiting on locks          = 0
Maximum associated agents        = 1
Priority at which application agents work = 0
Priority type                     = Static

Locks held by application        = 7
Lock waits since connect         = 0
```

## GET SNAPSHOT

```
Time application waited on locks (ms)      = 0
Deadlocks detected                          = 0
Lock escalations                           = 0
Exclusive lock escalations                  = 0
Number of Lock Timeouts since connected    = 0
Total time UOW waited on locks (ms)       = 0

Total sorts                                 = 3
Total sort time (ms)                        = 1
Total sort overflows                        = 0

Data pages copied to extended storage       = 0
Index pages copied to extended storage     = 0
Data pages copied from extended storage    = 0
Index pages copied from extended storage   = 0
Buffer pool data logical reads             = 32
Buffer pool data physical reads            = 13
Buffer pool data writes                    = 0
Buffer pool index logical reads            = 55
Buffer pool index physical reads           = 23
Buffer pool index writes                   = 0
Total buffer pool read time (ms)           = 364
Total buffer pool write time (ms)          = 0
Time waited for prefetch (ms)             = 0
Direct reads                               = 34
Direct writes                              = 0
Direct read requests                       = 4
Direct write requests                      = 0
Direct reads elapsed time (ms)             = 1
Direct write elapsed time (ms)             = 0

Number of SQL requests since last commit   = 13
Commit statements                          = 1
Rollback statements                        = 0
Dynamic SQL statements attempted           = 13
Static SQL statements attempted            = 1
Failed statement operations                = 0
Select SQL statements executed              = 1
Update/Insert/Delete statements executed   = 0
DDL statements executed                    = 0
Internal automatic rebinds                 = 0
Internal rows deleted                      = 0
Internal rows inserted                     = 0
Internal rows updated                      = 0
Internal commits                           = 1
Internal rollbacks                         = 0
Internal rollbacks due to deadlock         = 0
Binds/precompiles attempted               = 0
Rows deleted                              = 0
Rows inserted                             = 0
Rows updated                              = 0
Rows selected                             = 8
Rows read                                 = 53
Rows written                              = 0

UOW log space used (Bytes)                 = 0
Previous UOW completion timestamp          = 04-04-1997 14:29:55.728025
UOW start timestamp                        = 04-04-1997 14:31:46.580691
UOW stop timestamp                        =
```

## GET SNAPSHOT

```
UOW completion status           =
Open remote cursors              = 0
Open remote cursors with blocking = 0
Rejected Block Remote Cursor requests = 0
Accepted Block Remote Cursor requests = 1
Open local cursors               = 0
Open local cursors with blocking  = 0

Total User CPU Time used by agent (s) = 0.070000
Total System CPU Time used by agent (s) = 0.020000
Package cache lookups            = 1
Package cache inserts            = 1
Application section lookups      = 13
Application section inserts      = 1
Catalog cache lookups            = 0
Catalog cache inserts            = 0
Catalog cache overflows          = 0
Catalog cache heap full          = 0

Most recent operation            = Close
Cursor name                      = SQLCUR201
Most recent operation start timestamp = 04-04-1997 14:31:46.859493
Most recent operation stop timestamp  = 04-04-1997 14:31:46.930287

Statement type                   = Dynamic SQL Statement
Statement                        = Select
Section number                   =
Application creator              =
Package name                     =
Cursor name                      = SQLCUR201
Statement node number            = 0
Statement start timestamp        = 04-04-1997 14:31:46.859493
Statement stop timestamp         = 04-04-1997 14:31:46.930287
Total user CPU time              = 0.000000
Total system CPU time            = 0.000000
SQL compiler cost estimate in timerons = 9355
SQL compiler cardinality estimate = 1600
Degree of parallelism requested  = 1
Number of agents working on statement = 1
Number of subagents created for statement = 1
Statement sorts                  = 3
Total sort time                  = 1
Sort overflows                   = 0
Rows read                        = 0
Rows written                     = 0
Rows deleted                     = 0
Rows updated                     = 0
Rows inserted                    = 0
Rows fetched                     = 0
Number of subsections           = 1
Dynamic SQL statement text      =
SELECT DEPTNAME, DEPTNUMB, MANAGER, NAME FROM ORG, STAFF
  WHERE DEPTNUMB = DEPT AND MANAGER = ID ORDER BY DEPTNAME
```



## GET SNAPSHOT

The following is typical output resulting from a request for buffer pool information:

### Bufferpool Snapshot

```
Bufferpool name           = IBMDEFAULTBP
Database name             = SAMPLE
Database path             = /home/user1/user1/NODE0000/SQL00011/
Input database alias      = SAMPLE
Buffer pool data logical reads = 32
Buffer pool data physical reads = 13
Buffer pool data writes   = 0
Buffer pool index logical reads = 55
Buffer pool index physical reads = 23
Total buffer pool read time (ms) = 364
Total buffer pool write time (ms) = 0
Database files closed     = 0

Asynchronous pool data page reads = 0
Asynchronous pool data page writes = 0
Buffer pool index writes = 0
Asynchronous pool index page reads = 0
Asynchronous pool index page writes = 0
Total elapsed asynchronous read time = 0
Total elapsed asynchronous write time = 0
Asynchronous read requests = 0
Direct reads = 34
Direct writes = 0
Direct read requests = 4
Direct write requests = 0
Direct reads elapsed time (ms) = 1
Direct write elapsed time (ms) = 0

Data pages copied to extended storage = 0
Index pages copied to extended storage = 0
Data pages copied from extended storage = 0
Index pages copied from extended storage = 0
```

The following is typical output resulting from a request for table information:

### Table Snapshot

```
First database connect timestamp = 04-04-1997 14:29:55.197659
Last reset timestamp =
Snapshot timestamp = 04-04-1997 14:32:14.151875
Database name = SAMPLE
Database path = /home/user1/user1/NODE0000/SQL00011/
Input database alias = SAMPLE
Number of accessed tables = 6
```

Table Schema	Table Name	Table Type	Rows Written	Rows Read	Overflows
USER1	STAFF	User	0	35	0
USER1	ORG	User	0	8	0
SYSIBM	SYSTABLES	Catalog	0	2	0
SYSIBM	SYSTABLESPACES	Catalog	0	3	0
SYSIBM	SYSPLAN	Catalog	0	1	0
SYSIBM	SYSDBAUTH	Catalog	0	3	0

## GET SNAPSHOT

The following is typical output resulting from a request for table space information:

### Tablespace Snapshot

```
First database connect timestamp      = 04-04-1997 14:29:55.197659
Last reset timestamp                  =
Snapshot timestamp                    = 04-04-1997 14:32:14.151875
Database name                         = SAMPLE
Database path                         = /home/user1/user1/NODE0000/SQL00011/
Input database alias                  = SAMPLE
Number of accessed tablespaces        = 3
```

```
Tablespace name                       = SYSCATSPACE
```

```
Data pages copied to extended storage = 0
Index pages copied to extended storage = 0
Data pages copied from extended storage = 0
Index pages copied from extended storage = 0
```

```
Buffer pool data logical reads        = 26
Buffer pool data physical reads       = 11
Asynchronous pool data page reads     = 0
Buffer pool data writes                = 0
Asynchronous pool data page writes    = 0
Buffer pool index logical reads        = 55
Buffer pool index physical reads       = 23
Asynchronous pool index page reads     = 0
Buffer pool index writes               = 0
Asynchronous pool index page writes    = 0
Total buffer pool read time (ms)      = 342
Total buffer pool write time (ms)     = 0
Total elapsed asynchronous read time  = 0
Total elapsed asynchronous write time = 0
Asynchronous read requests            = 0
```

```
Direct reads                          = 34
Direct writes                          = 0
Direct read requests                   = 4
Direct write requests                  = 0
Direct reads elapsed time (ms)        = 1
Direct write elapsed time (ms)        = 0
```

```
Number of files closed                 = 0
```

```
Tablespace name                       = TEMPSPACE1
```

```
Data pages copied to extended storage = 0
Index pages copied to extended storage = 0
Data pages copied from extended storage = 0
Index pages copied from extended storage = 0
```

```
Buffer pool data logical reads        = 0
Buffer pool data physical reads       = 0
Asynchronous pool data page reads     = 0
Buffer pool data writes                = 0
Asynchronous pool data page writes    = 0
Buffer pool index logical reads        = 0
Buffer pool index physical reads       = 0
Asynchronous pool index page reads     = 0
```

## GET SNAPSHOT

```
Buffer pool index writes           = 0
Asynchronous pool index page writes = 0
Total buffer pool read time (ms)   = 0
Total buffer pool write time (ms)  = 0
Total elapsed asynchronous read time = 0
Total elapsed asynchronous write time = 0
Asynchronous read requests         = 0

Direct reads                       = 0
Direct writes                      = 0
Direct read requests               = 0
Direct write requests              = 0
Direct reads elapsed time (ms)     = 0
Direct write elapsed time (ms)     = 0

Number of files closed              = 0

Tablespace name                    = USERSPACE1

Data pages copied to extended storage = 0
Index pages copied to extended storage = 0
Data pages copied from extended storage = 0
Index pages copied from extended storage = 0

Buffer pool data logical reads      = 6
Buffer pool data physical reads     = 2
Asynchronous pool data page reads  = 0
Buffer pool data writes             = 0
Asynchronous pool data page writes = 0
Buffer pool index logical reads     = 0
Buffer pool index physical reads    = 0
Asynchronous pool index page reads  = 0
Buffer pool index writes            = 0
Asynchronous pool index page writes = 0
Total buffer pool read time (ms)    = 22
Total buffer pool write time (ms)   = 0
Total elapsed asynchronous read time = 0
Total elapsed asynchronous write time = 0
Asynchronous read requests         = 0

Direct reads                       = 0
Direct writes                      = 0
Direct read requests               = 0
Direct write requests              = 0
Direct reads elapsed time (ms)     = 0
Direct write elapsed time (ms)     = 0

Number of files closed              = 0
```

The following is typical output resulting from a request for lock information:

### Database Lock Snapshot

```
Database name           = SAMPLE
Database path          = /home/user1/user1/NODE0000/SQL00011/
Input database alias   = SAMPLE
Locks held              = 7
Applications currently connected = 1
Applications currently waiting on locks = 0
Snapshot timestamp     = 04-04-1997 14:32:14.151875
```

## GET SNAPSHOT

```
Application handle          = 5
Application ID             = *LOCAL.user1.970404192956
Sequence number           = 0001
Application name           = db2bp_32
Authorization ID           = USER1
Application status         = UOW Waiting
Status change time        =
Application code page      = 850
Locks held                 = 7
Total wait time (ms)      = 0
```

Object Name	Object Type	Tablespace Name	Table Schema	Table Name	Mode	Status
1545	Row	SYSCATSPACE	SYSIBM	SYSTABLES	NS	Granted
1544	Row	SYSCATSPACE	SYSIBM	SYSTABLES	NS	Granted
2	Table	SYSCATSPACE	SYSIBM	SYSTABLES	IS	Granted
27	Table	SYSCATSPACE	SYSIBM	SYSTABLESPACES	S	Granted
257	Row	SYSCATSPACE	SYSIBM	SYSPLAN	S	Granted
7	Table	SYSCATSPACE	SYSIBM	SYSPLAN	IS	Granted
0	Internal				S	Granted

### Usage Notes

To obtain a snapshot from a remote instance (or a different local instance), it is necessary to first attach to that instance. If an alias for a database residing at a different instance is specified, an error message is returned.

To obtain some statistics, it is necessary that the database system monitor switches are turned on.

No data is returned following a request for table information if any of the following is true:

- The TABLE recording switch is turned off
- No tables have been accessed since the switch was turned on
- No tables have been accessed since the last RESET MONITOR command was issued.

### See Also

“GET MONITOR SWITCHES” on page 197

“LIST APPLICATIONS” on page 225

“RESET MONITOR” on page 333.

---

**HELP**

Permits the user to invoke help from the Information Center.

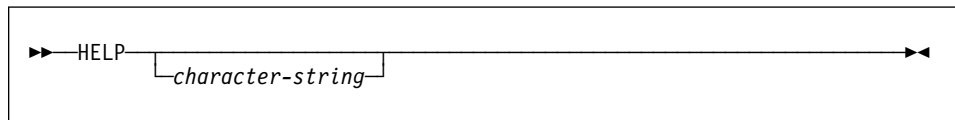
This command is not available on UNIX based systems.

**Authorization**

None

**Required Connection**

None

**Command Syntax****Command Parameters**

**HELP** *character-string*

Any SQL or DB2 command, or any other item listed in the Information Center.

**Usage Notes**

The Information Center must be installed on the user's system. HTML books in the DB2 library must be located in the \sql11ib\doc\html subdirectory.

The command line processor will not know if the command succeeds or fails, and cannot report error conditions.

## IMPORT

---

### IMPORT

Inserts data from an external file with a supported file format into a table or view. A faster alternative is “LOAD” on page 262.

#### Authorization

- IMPORT using the INSERT option requires one of the following:

*sysadm*

*dbadm*

CONTROL privilege on the table or view

INSERT and SELECT privilege on the table or view.

- IMPORT to an existing table using the INSERT\_UPDATE, REPLACE, or the REPLACE\_CREATE option, requires one of the following:

*sysadm*

*dbadm*

CONTROL privilege on the table or view.

- IMPORT to a table that does not exist using the CREATE, or the REPLACE\_CREATE option, requires one of the following:

*sysadm*

*dbadm*

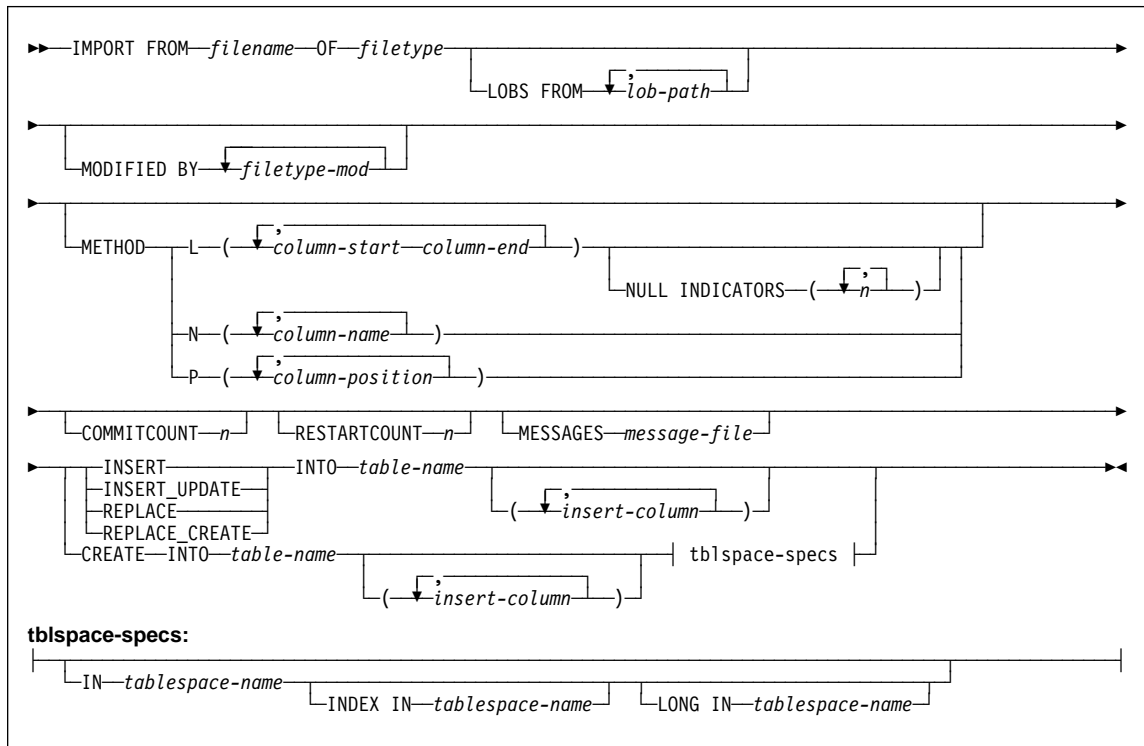
CREATETAB authority on the database, and one of:

- IMPLICIT\_SCHEMA authority on the database, if the schema name of the table does not exist
- CREATEIN privilege on the schema, if the schema of the table exists.

#### Required Connection

Database. If implicit connect is enabled, a connection to the default database is established.

## Command Syntax



## Command Parameters

### FROM *filename*

Specifies the file that contains the data being imported. If the path is omitted, the current working directory is used.

### OF *filetype*

Specifies the format of the data in the input file:

- ASC (non-delimited ASCII format)
- DEL (delimited ASCII format), which is used by a variety of database manager and file manager programs
- WSF (work sheet format), which is used by programs such as:
  - Lotus 1-2-3
  - Lotus Symphony
- IXF (integrated exchange format, PC version), which means it was exported from the same or another DB2 table. An IXF file also contains the table definition and definitions of any existing indexes, except when columns are specified in the SELECT statement.

## IMPORT

For more information about file formats, see Appendix C, “IMPORT/EXPORT/LOAD Utility File Formats” on page 399.

### **LOBS FROM** *lob-path*

Specifies the path or paths that store LOB files. The names of the LOB data files are stored in the main data file (ASC, DEL, or IXF), in the column that will be loaded into the LOB column. This option is ignored if *lobsinfile* is not specified within the *filetype-mod* string.

### **MODIFIED BY** *filetype-mod*

Specifies additional information unique to the ASC, DEL, WSF, or IXF file format (see page 217).

### **METHOD**

*L*

Specifies the start and end column numbers from which to import data.

**Note:** This method can only be used with ASC files, and is the only valid option for that file type.

*N*

Specifies the names of the columns to be imported.

**Note:** This method can only be used with IXF files.

*P*

Specifies the numbers of the columns to be imported.

**Note:** This method can only be used with IXF or DEL files, and is the only valid option for the DEL file type.

### **NULL INDICATORS** *n*

Specifies a column (by number) to be used as a null indicator field. If this option is used, a null indicator column for each data column must also be specified. Zero (0) indicates that the data column is not nullable, and that there will always be data in that column.

While processing each row, a Y indicates that the column data is NULL, while an N indicates that the column data is not NULL, and that column data specified by the METHOD L option will be imported.

### **COMMITCOUNT** *n*

Performs a commit every *n* records.

### **RESTARTCOUNT** *n*

Specifies that an import is to be started at record *n* + 1. The first *n* records are skipped.

### **MESSAGES** *message-file*

Specifies the destination for warning and error messages that occur during import. If the file already exists, IMPORT appends the information. If the complete path to the file is not specified, IMPORT uses the current directory and the default drive as the destination. If *message-file* is omitted, the messages are written to standard output.

### **INSERT**

Adds the imported data to the table without changing the existing table data.



**INSERT\_UPDATE**

Adds rows of imported data to the target table, or updates existing rows (of the target table) with matching primary keys.

**REPLACE**

Deletes all existing data in the table, and inserts the imported data. The table definition and the index definitions are not changed. This option can only be used if the table exists.

**REPLACE\_CREATE**

If the table exists, deletes all existing data in the table, and inserts the imported data without changing the table definition or the index definitions.

If the table does not exist, creates the table definition and row contents. If the data was exported from a database manager or a DB2 for OS/2 table, indexes are also created.

This option can only be used with IXF files.

**CREATE**

Creates the table definition and row contents. If the data was exported from a database manager table, indexes are also created. This option can only be used with IXF files.

**Note:** If the data was exported from an MVS host database, and it contains LONGVAR fields whose lengths, calculated on the page size, are less than 254, CREATE may fail because the rows are too long. In this case, the table should be created manually, and IMPORT with INSERT should be invoked, or, alternatively, the LOAD command should be used.

**INTO** *table-name*

Specifies the database table into which the data is to be imported. This table cannot be a system table.

One can use an alias for INSERT, INSERT\_UPDATE, or REPLACE, except in the case of a down-level server, when the fully qualified or the unqualified table name should be used. A qualified table name is in the form: *schema.tablename*. The *schema* is the user name under which the table was created.

*insert-column*

Specifies the name of a column within the table or view into which the data is to be inserted.

**IN** *tablespace-name*

Identifies the table space in which the table will be created. The table space must exist, and must be a REGULAR table space. If no other table space is specified, all table parts are stored in this table space. If this clause is not specified, the table is created in a table space created by the authorization ID. If none is found, the table is placed into the default table space USERSPACE1. If USERSPACE1 has been dropped, the table creation fails.

**INDEX IN** *tablespace-name*

Identifies the table space in which any indexes on the table will be created. This option is allowed only when the primary table space specified in the IN

## IMPORT

clause is a DMS table space. The specified table space must exist, and must be a REGULAR DMS table space.

**Note:** Specifying which table space will contain a table's index can only be done when the table is created.

### **LONG IN** *tablespace-name*

Identifies the table space in which the values of any long columns (LONG VARCHAR, LONG VARGRAPHIC, LOB data types, or distinct types with any of these as source types) will be stored. This option is allowed only when the primary table space specified in the IN clause is a DMS table space. The table space must exist, and must be a LONG DMS table space.

## Example

The following example shows how to import information from `myfile.ixf` to the STAFF table:

```
db2 import from myfile.ixf of ixf messages msg.txt insert into staff
```

```
SQL3150N The H record in the PC/IXF file has product "DB2 01.00", date
"19970220", and time "140848".

SQL3153N The T record in the PC/IXF file has name "ex", qualifier " ",
and source " ".

SQL3109N The utility is beginning to load data from file "ex".

SQL3110N The utility has completed processing. "58" rows were read from the
input file.

SQL3221W ...Begin COMMIT WORK. Input Record Count = "58".

SQL3222W ...COMMIT of any database changes was successful.

SQL3149N "58" rows were processed from the input file. "58" rows were
successfully inserted into the table. "0" rows were rejected.
```

## Usage Notes

The database table must exist before data in the ASC, DEL, or WSF file formats can be imported; however, if the table does not already exist, `IMPORT CREATE` or `IMPORT REPLACE_CREATE` creates the table when it imports data from a PC/IXF file.

The import utility will issue a `COMMIT` or a `ROLLBACK` statement. Ensure that all database activity in the current transaction has completed, and that all locks are released (by issuing a `COMMIT` or a `ROLLBACK`) before issuing the `IMPORT` command.

PC/IXF import should be used to move data between databases. If character data containing row separators is exported to a delimited ASCII (DEL) file and processed by

## IMPORT

a text transfer program (moving, for example between OS/2 and AIX systems), fields containing the row separators will shrink or expand.

PC/IXF file format specifications permit migration of data between OS/2 (IBM Extended Services for OS/2, OS/2 Extended Edition, and DB2 for OS/2) databases and DB2 for AIX databases via export, binary copying of files between OS/2 and AIX, and import. The file copying step is not necessary if the source and the target databases are both accessible from the same client.

The data in ASC and DEL files is assumed to be in the code page of the client application performing the import.

IXF files, which allow for different code pages, are recommended when importing data in different code pages. If the IXF file and the import utility are in the same code page, processing occurs as for a regular application. If the two differ, and the FORCEIN option is specified, IMPORT assumes that data in the IXF file has the same code page as the application performing the import. This occurs even if there is a conversion table for the two code pages. If the two differ, FORCEIN is not specified, and there is a conversion table, all data in the IXF file will be converted from the file code page to the application code page. If the two differ, FORCEIN is not specified, and there is no conversion table, the import will fail. This applies only to IXF files on DB2 for AIX clients.

DB2 Connect can be used to import data to DRDA servers such as DB2 for MVS, SQL/DS, and OS/400. Only PC/IXF import (INSERT option) is supported. The RESTARTCOUNT parameter, but not the COMMITCOUNT parameter, is also supported.

Importing a multiple-part PC/IXF file whose individual parts are copied from an OS/2 system to an AIX system is supported on DB2.

### File Type Modifications

**Note:** The import utility does not issue a warning if an attempt is made to use unsupported file types with the MODIFIED BY option. If this is attempted, the import fails, and an error code is returned.

<i>Table 8 (Page 1 of 3). Valid File Type Modifications (IMPORT)</i>	
Modification	Description
<b>All File Formats</b>	
compound= <i>x</i>	<i>x</i> is a number between 1 and 100 inclusive (7 on DOS/Windows). Uses nonatomic compound SQL to insert the data, and <i>x</i> statements will be attempted each time.
lobsinfile	<i>lob-path</i> specifies the path to the files containing LOB values.
<b>ASCII File Formats (ASC/DEL)</b>	

# IMPORT

<i>Table 8 (Page 2 of 3). Valid File Type Modifications (IMPORT)</i>	
<b>Modification</b>	<b>Description</b>
implieddecimal	The location of an implied decimal point is determined by the column definition; it is no longer assumed to be at the end of the value. For example, the value 12345 is loaded into a DECIMAL(8,2) column as 123.45, <i>not</i> 12345.00.
noeofchar	The optional end-of-file character x'1A' is not recognized as the end of file. Processing continues as if it were a normal character.
<b>ASC (Non-delimited ASCII) File Format</b>	
reclen=x	x is an integer with a maximum value of 32767. x characters are read for each row, and a new-line character is not used to indicate the end of the row.
striptblanks	<p>Truncates any trailing blank spaces when loading data into a variable-length field. If this option is not specified, blank spaces are kept.</p> <p>In the following example, striptblanks causes the import utility to truncate trailing blank spaces:</p> <pre>db2 import from myfile.asc of asc modified by striptblanks method l (1 10, 12 15) messages msgs.txt insert into staff</pre> <p>This option cannot be specified together with striptnulls. These are mutually exclusive options.</p> <p><b>Note:</b> This option replaces the obsolete t option, which is supported for back-level compatibility only.</p>
striptnulls	<p>Truncates any trailing NULLs (0x00 characters) when loading data into a variable-length field. If this option is not specified, NULLs are kept.</p> <p>This option cannot be specified together with striptblanks. These are mutually exclusive options.</p> <p><b>Note:</b> This option replaces the obsolete padwithzero option, which is supported for back-level compatibility only.</p>
<b>DEL (Delimited ASCII) File Format</b>	
chardelx	<p>x is a single character string delimiter. The default value is a double quotation mark ("). The specified character is used in place of double quotation marks to enclose a character string.<sup>ab</sup></p> <p>The single quotation mark (') can also be specified as a character string delimiter. In the following example, chardel'' causes the import utility to interpret any single quotation mark (') it encounters as a character string delimiter:</p> <pre>db2 "import from myfile.del of del modified by chardel'' method p (1, 4) insert into staff (id, years)"</pre>

Table 8 (Page 3 of 3). Valid File Type Modifications (IMPORT)

Modification	Description
coldelx	<p>x is a single character column delimiter. The default value is a comma (.). The specified character is used in place of a comma to signal the end of a column.<sup>ab</sup></p> <p>In the following example, coldel; causes the import utility to interpret any semicolon (;) it encounters as a column delimiter:</p> <pre>db2 import from myfile.del of del modified by coldel; messages msgs.txt insert into staff</pre>
decptx	<p>x is a single character substitute for the period as a decimal point character. The default value is a period (.). The specified character is used in place of a period as a decimal point character.<sup>ab</sup></p> <p>In the following example, decpt; causes the import utility to interpret any semicolon (;) it encounters as a decimal point:</p> <pre>db2 "import from myfile.del of del modified by char del' decpt; messages msgs.txt insert into staff"</pre>
<b>IXF File Format</b>	
forcein	<p>Directs the utility to accept data despite code page mismatches, and to suppress translation between code pages.</p> <p>Fixed length target fields are checked to verify that they are large enough for the data. If nochecklengths is specified, no checking is done, and an attempt is made to load each row.</p>
indexif	<p>Directs the utility to drop all indexes currently defined on the existing table, and to create new ones from the index definitions in the PC/IXF file. This option can only be used when the contents of a table are being replaced. It cannot be used with a view, or when a <i>insert-column</i> is specified.</p>
indexschema= <i>schema</i>	<p>Uses the specified <i>schema</i> for the index name during index creation. If <i>schema</i> is not specified (but the keyword <i>indexschema</i> is specified), uses the connection user ID. If the keyword is not specified, uses the schema in the IXF file.</p>
<p><b>Note:</b></p> <ul style="list-style-type: none"> <li>a Table 7 on page 165 lists the characters that can be used as delimiter overrides.</li> <li>b The character must be specified in the code page of the source data.</li> </ul> <p>The character code point (instead of the character symbol), can be specified using the syntax xJJ or 0xJJ, where JJ is the hexadecimal representation of the code point. For example, to specify the # character as a column delimiter, use one of the following:</p> <pre>... modified by coldel# ... ... modified by coldel0x23 ... ... modified by coldelX23 ...</pre>	

## **IMPORT**

### **See Also**

“EXPORT” on page 161  
“LOAD” on page 262.

---

## INITIALIZE TAPE

DB2 for Windows NT supports backup and restore to streaming tape devices. Use this command for tape initialization.

This command is available on Windows NT only.

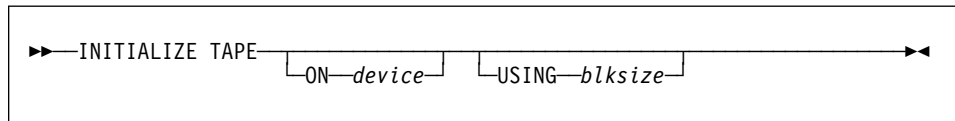
### Authorization

None

### Required Connection

None

### Command Syntax



### Command Parameters

**ON** *device*

Specifies a valid tape device name. The default value is `\\.\TAPE0`.

**USING** *blksize*

Specifies the block size for the device. The value must be a multiple of 4KB.

### See Also

“REWIND TAPE” on page 343

“SET TAPE POSITION” on page 360.

## INVOKE STORED PROCEDURE

---

### INVOKE STORED PROCEDURE

Invokes a procedure stored at the location of a database. Also known as the Database Application Remote Interface (DARI). The server procedure executes at the location of the database, and returns data to the client application.

The application programmer designs the program to run in two parts, one on the client and the other on the server. The server procedure at the database runs within the same transaction as the client application. If the client application and the server procedure are on the same node, the server procedure is executed locally.

**Note:** This command has been replaced by the SQL CALL statement (see the *SQL Reference*). SQL CALL cannot be used from within the CLP.

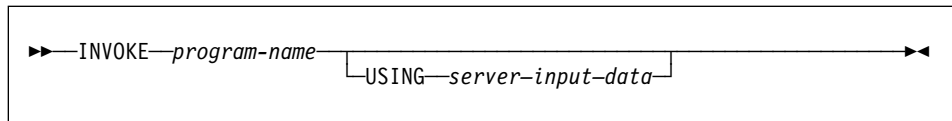
#### Authorization

CONNECT privilege on a database.

#### Required Connection

Database

#### Command Syntax



**Note:** Do not use INVOKE to call server procedures that use input or output SQLDA structures, including server procedures that return data. For more information, see the *Embedded SQL Programming Guide*.

#### Command Parameters

*program-name*

Specifies the procedure to be run on the server. This parameter can be specified in one of the following ways:

- By a *procName* with no extensions. This notation tells the database manager to load a DARI library named *procName* into memory. It is assumed that the name of the function routine to be executed is identical to the library name. For example,

```
db2 invoke foo
```

will load the DARI library named F00 and execute the function routine F00() within the library.

The database manager will find the DARI libraries in the default directory \$HOME/sql1lib/function of the instance owner.

- By the exclamation sign (!) delimited name, as in *procName!funcName*. This notation tells the database manager to load



## INVOKE STORED PROCEDURE

a DARI library, named `procName`, into memory. The function routine to be executed is `funcName`. This convention allows similar function routines to be packaged in the same DARI library.

- By an absolute path name, as in `/u/cche/procName!/funcName`, for example. This notation includes the storage path of the DARI library. In this example, the DARI library named `procName` is stored in the directory `/u/cche`. The function routine to be executed is `funcName`.

**Note:** To support portability between various versions of DB2 products, the `!` delimiter can be replaced by the backslash (`\`) delimiter.

### **USING** *server-input-data*

Specifies any information that is passed to the server routine. A variable character string, free form, flexible parameter that can be used to transmit input data according to specific needs.

## LIST ACTIVE DATABASES

---

### LIST ACTIVE DATABASES

Displays a subset of the information listed by the GET SNAPSHOT FOR ALL DATABASES command (see “GET SNAPSHOT” on page 199). For each active database, this command displays the following:

- Database name
- Number of applications currently connected to the database
- Database path.

### Scope

This command can be issued from any node that is listed in \$HOME/sqllib/db2nodes.cfg. It returns the same information from any of these nodes.

### Authorization

None

### Command Syntax

▶—LIST ACTIVE DATABASES—▶

### Command Parameters

None

### Examples

Following is sample output from the LIST ACTIVE DATABASES command:

```
Active Databases
Database name           = TEST
Applications connected  = 0
currently
Database path           = /home/smith/smith/NODE0000/SQL00002/
Database name           = SAMPLE
Applications connected  = 1
currently
Database path           = /home/smith/smith/NODE0000/SQL00001/
```

### See Also

“GET SNAPSHOT” on page 199.

## LIST APPLICATIONS

Displays to standard output the application program name, authorization ID (user name), application handle, application ID, and database name of all active database applications. This command can also optionally display an application's sequence number, status, status change time, and database path.

### Scope

This command only returns information for the node on which it is issued.

### Authorization

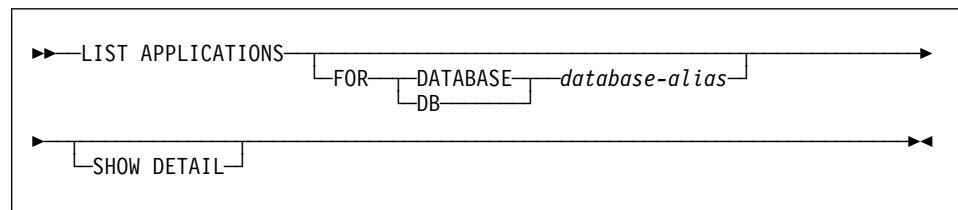
One of the following:

*sysadm*  
*sysctrl*  
*sysmaint*

### Required Connection

Instance. To list applications for a remote instance, it is necessary to first attach to that instance.

### Command Syntax



### Command Parameters

#### FOR DATABASE *database-alias*

Information for each application that is connected to the specified database is to be displayed. Database name information is not displayed. If this option is not specified, the command displays the information for each application that is currently connected to any database at the node to which the user is currently attached.

The default application information is comprised of the following:

- Authorization ID
- Application program name
- Application handle
- Application ID
- Database name.

## LIST APPLICATIONS

### SHOW DETAIL

Output will include the following additional information:

- Sequence #
- Application status
- Status change time
- Database path.

**Note:** If this option is specified, it is recommended that the output be redirected to a file, and that the report be viewed with the help of an editor. The output lines may wrap around when displayed on the screen.

### Example

The following is sample output from LIST APPLICATIONS:

Auth Id	Application Name	Appl. Handle	Application Id	DB Name	# of Agents
smith	db2bp_32	12	*LOCAL.smith.970220191502	TEST	1
smith	db2bp_32	11	*LOCAL.smith.970220191453	SAMPLE	1

**Note:** For more information about these fields, see the *System Monitor Guide and Reference*.

### Usage Notes

The database administrator can use the output from this command as an aid to problem determination. In addition, this information is required if the database administrator wants to use "GET SNAPSHOT" on page 199 or "FORCE APPLICATION" on page 167 in an application.

To list applications at a remote instance (or a different local instance), it is necessary to first attach to that instance. If FOR DATABASE is specified when an attachment exists, and the database resides at an instance which differs from the current attachment, the command will fail.

---

## LIST BACKUP/HISTORY

Lists restore sets for full database and table space level backups, or lists entries in the recovery history file.

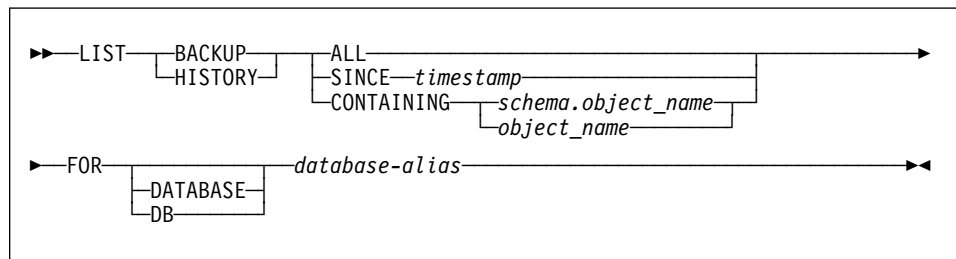
### Authorization

None

### Required Connection

Instance. An explicit attachment is not required. If the database is listed as remote, an instance attachment to the remote node is established for the duration of the command.

### Command Syntax



### Command Parameters

#### BACKUP

Lists backups and restores.

#### HISTORY

Lists backups, restores, and loads.

#### ALL

Lists all entries in the recovery history file.

#### SINCE *timestamp*

A complete time stamp (format *yyyymmddhhnss*), or an initial prefix (minimum *yyyy*) can be specified. All entries with time stamps equal to or less than the time stamp provided are listed.

#### CONTAINING *schema.object\_name*

This qualified name uniquely identifies a table.

#### CONTAINING *object\_name*

This unqualified name uniquely identifies a table space.

#### FOR DATABASE *database-alias*

Used to identify the database whose recovery history file is to be listed.

### Examples

```
db2 list backup containing t3.table50 for sample
```

```
db2 list history since 19970201 for sample
```

## LIST BACKUP/HISTORY

### Usage Notes

The report generated by this command contains the following symbols:

- Device
  - A - ADSM
  - D - Disk
  - K - Diskette
  - O - Other
  - T - Tape
  - U - User Exit
- Object
  - D - Database
  - P - Tablespace
  - T - Table
- Operation
  - B - Backup
  - C - Copy
  - L - Load
  - R - Restore
- Type
  - F - Offline
  - I - Insert(LOAD)
  - N - Online

---

### LIST COMMAND OPTIONS

Lists the current settings for the environment variables:

- **DB2BQTIME**
- **DB2DQTRY**
- **DB2RQTIME**
- **DB2IQTIME**
- **DB2OPTIONS.**

#### Authorization

None

#### Required Connection

None

#### Command Syntax

```
▶—LIST COMMAND OPTIONS—▶
```

#### Command Parameters

None

## LIST COMMAND OPTIONS

### Example

The following is sample output from LIST COMMAND OPTIONS:

Command Line Processor Option Settings		
Backend process wait time (seconds)		(DB2BQTIME) = 1
No. of retries to connect to backend		(DB2BQTRY) = 60
Request queue wait time (seconds)		(DB2RQTIME) = 5
Input queue wait time (seconds)		(DB2IQTIME) = 5
Command options		(DB2OPTIONS) =
Option	Description	Current Setting
-a	Display SQLCA	OFF
-c	Auto-Commit	ON
-e	Display SQLCODE/SQLSTATE	OFF
-f	Read from input file	OFF
-l	Log commands in history file	OFF
-o	Display output	ON
-p	Display interactive input prompt	ON
-r	Save output to report file	OFF
-s	Stop execution on command error	OFF
-t	Set statement termination character	OFF
-v	Echo current command	OFF
-w	Display FETCH/SELECT warning messages	ON
-z	Save all output to output file	OFF

### Usage Notes

For detailed information about these options, see “Command Line Processor Invocation and Options” on page 69.



---

## LIST DATABASE DIRECTORY

Lists the contents of the system database directory. If a path is specified, the contents of the local database directory are listed.

### Scope

If this command is issued without the *ON path* parameter, the system database directory is returned. This information is the same at all nodes.

If the *ON path* parameter is specified, the local database directory on that path is returned. This information is not the same at all nodes.

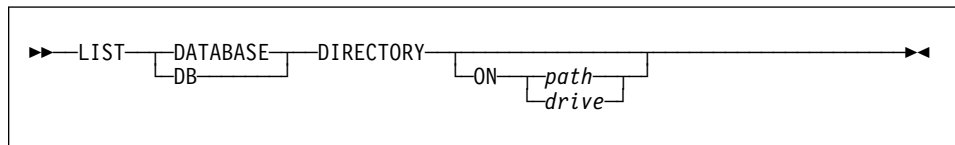
### Authorization

None

### Required Connection

None. Directory operations affect the local directory only.

### Command Syntax



### Command Parameters

**ON** *path/drive*

Specifies the local database directory from which to list information. If not specified, the contents of the system database directory are listed.

## LIST DATABASE DIRECTORY

### Examples

The following shows sample output for a system database directory:

```
System Database Directory

Number of entries in the directory = 2

Database 1 entry:

Database alias           = SAMPLE
Database name           = SAMPLE
Database drive          = /home/smith
Database release level  = 8.00
Comment                 =
Directory entry type    = Indirect
Catalog node number     = 0

Database 2 entry:

Database alias           = GDB1
Database global name    = ../../cell_name/dir_name/gdb1
Database release level  = 8.00
Comment                 = DCE database
Directory entry type    = DCE
Catalog node number     = -1
```

The following shows sample output for a local database directory:

```
Local Database Directory on /u/smith

Number of entries in the directory = 1

Database 1 entry:

Database alias           = SAMPLE
Database name           = SAMPLE
Database directory      = SQL00001
Database release level  = 8.00
Comment                 =
Directory entry type    = Home
Catalog node number     = 0
Node number             = 0
```

These fields are identified as follows:

#### **Database alias**

The value of the *alias* parameter when the database was created or cataloged. If an alias was not entered when the database was cataloged,

## LIST DATABASE DIRECTORY

the database manager uses the value of the *database-name* parameter when the database was cataloged.

### **Database global name**

The fully qualified name that uniquely identifies the database in the DCE name space.

### **Database name**

The value of the *database-name* parameter when the database was cataloged. This name is usually the name under which the database was created.

### **Local database directory**

The path on which the database resides. This field is filled in only if the system database directory has been scanned.

### **Database directory/Database drive**

The name of the directory or drive where the database resides. This field is filled in only if the local database directory has been scanned.

### **Node name**

The name of the remote node. This name corresponds to the value entered for the *nodename* parameter when the database and the node were cataloged.

### **Database release level**

The release level of the database manager that can operate on the database.

### **Comment**

Any comments associated with the database that were entered when it was cataloged.

### **Directory entry type**

The location of the database:

- A *remote* entry describes a database that resides on another node.
- An *indirect* entry describes a database that is local. Databases that reside on the same node as the system database directory are thought to indirectly reference the home entry (to a local database directory), and are considered indirect entries.
- A *home* entry indicates that the database directory is on the same path as the local database directory.

All entries in the system database directory are either remote or indirect. All entries in local database directories are identified in the system database directory as indirect entries.

### **Authentication**

The authentication type cataloged at the client is used to determine whether the connection is being done with system or with DCE security.

### **Catalog node number**

Specifies which node is the catalog node (MPP systems only). This is the node on which the CREATE DATABASE command was issued.

## LIST DATABASE DIRECTORY

### **Node number**

Specifies the number that is assigned in `db2nodes.cfg` to the node where the command was issued (MPP systems only). In non-MPP systems where there is no `db2nodes.cfg` file, the node number will always be zero.

### **See Also**

“CHANGE DATABASE COMMENT” on page 139

“CREATE DATABASE” on page 143.

---

## LIST DCS APPLICATIONS

Displays the contents of the Database Connection Services (DCS) directory to standard output.

### Authorization

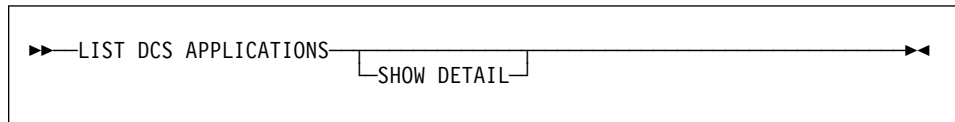
One of the following:

*sysadm*  
*sysctrl*  
*sysmaint*

### Required Connection

Instance. To list the DCS applications at a remote instance, it is necessary to first attach to that instance.

### Command Syntax



### Command Parameters

#### LIST DCS APPLICATIONS

The default application information includes:

- Host authorization ID (*username*)
- Application program name
- Agent ID
- Outbound application ID (*luwid*).

#### SHOW DETAIL

Specifies that output include the following additional information:

- Application ID
- Application sequence number
- Client database alias
- Client node name (*nname*)
- Client product ID
- Code page ID
- Code page
- Outbound sequence number
- Host database name
- Host product ID.

## LIST DCS APPLICATIONS

### Example

The following is sample output from LIST DCS APPLICATIONS:

Auth Id	Application Name	Agent Id	Outbound Application Id
DDCSUS1	db2bp	89330	CAIBMOML.OMXT4H08.A79EAA3C6E29

**Note:** For more information about these fields, see the *System Monitor Guide and Reference*.

### Usage Notes

The database administrator can use this command to match client application connections *to* the gateway with corresponding host connections *from* the gateway.

The database administrator can also use agent ID information to force specified applications off a DDCS server.

### See Also

“FORCE APPLICATION” on page 167.

---

**LIST DCS DIRECTORY**

Lists the contents of the Database Connection Services (DCS) directory.

**Authorization**

None

**Required Connection**

None

**Command Syntax**

```
▶—LIST DCS DIRECTORY—▶
```

**Command Parameters**

None

**Example**

The following is sample output from LIST DCS DIRECTORY:

```

Database Connection Services (DCS) Directory

Number of entries in the directory = 1

DCS 1 entry:

Local database name           = DB2
Target database name          = DSN_DB_1
Application requestor name    =
DCS parameters                =
Comment                       = DB2/MVS Location name DSN_DB_1
DCS directory release level   = 0x0100

```

These fields are identified as follows:

**Local database name**

Specifies the alias of the target host database. This corresponds to the *database-name* parameter entered when the host database was cataloged in the DCS directory.

## LIST DCS DIRECTORY

### Target database name

Specifies the name of the host database that can be accessed. This corresponds to the *target-database-name* parameter entered when the host database was cataloged in the DCS directory.

### Application requester name

Specifies the name of the program residing on the application requester or server.

### DCS parameters

String that contains the connection and operating environment parameters to use with the application requester. Corresponds to the parameter string entered when the host database was cataloged. The string must be enclosed by double quotation marks, and the parameters must be separated by commas.

For more information about DCS parameters, see the *DB2 Connect User's Guide*.

### Comment

Describes the database entry.

### DCS directory release level

Specifies the version number of the Distributed Database Connection Services program under which the database was created.

## Usage Notes

The DCS directory is created the first time that "CATALOG DCS DATABASE" on page 121 is invoked. It is maintained on the path/drive where DB2 was installed, and provides information about host databases that the workstation can access if the DDCS program has been installed. The host databases can be:

- DB2 for MVS/ESA databases on System/370 and System/390 architecture host computers
- DB2 for VSE & VM databases on System/370 and System/390 architecture host computers
- DB2 for AS/400 databases on Application System/400 (AS/400) host computers.



---

### LIST DRDA INDOUBT TRANSACTIONS

Provides a list of transactions that are indoubt between partner LUs connected by LU 6.2 protocols.

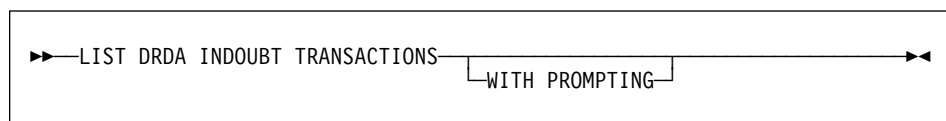
#### Authorization

*sysadm*

#### Required Connection

Instance

#### Command Syntax



#### Command Parameters

##### WITH PROMPTING

Indicates that indoubt transactions are to be processed. If this parameter is specified, an interactive dialog mode is initiated, permitting the user to commit or roll back indoubt transactions. If this parameter is not specified, indoubt transactions are written to the standard output device, and the interactive dialog mode is not initiated.

**Note:** A forget option is not supported. Once the indoubt transaction is committed or rolled back, the transaction is automatically forgotten.

Interactive dialog mode permits the user to:

- List all indoubt transactions (enter 1)
- List indoubt transaction number *x* (enter 1, followed by a valid transaction number)
- Quit (enter q)
- Commit transaction number *x* (enter c, followed by a valid transaction number)
- Roll back transaction number *x* (enter r, followed by a valid transaction number).

**Note:** A blank space must separate the command letter from its argument.

Before a transaction is committed or rolled back, the transaction data is displayed, and the user is asked to confirm the action.

## LIST DRDA INDOUBT TRANSACTIONS

### Usage Notes

DRDA indoubt transactions occur when communication is lost between coordinators and participants in distributed units of work.

A distributed unit of work lets a user or application read and update data at multiple locations within a single unit of work. Such work requires a two-phase commit.

The first phase requests all the participants to prepare for commit. The second phase commits or rolls back the transactions. If a coordinator or participant becomes unavailable after the first phase, the distributed transactions are indoubt.

Issuing a LIST DRDA INDOUBT TRANSACTIONS will return information on indoubt transactions.

Before issuing LIST DRDA INDOUBT TRANSACTIONS, the application process must be connected to the Sync Point Manager (SPM) instance. Use the *spm\_name* database manager configuration parameter as the *dbalias* on the CONNECT statement (see the *SQL Reference* for more information about using CONNECT).

---

### LIST INDOUBT TRANSACTIONS

Provides a list of transactions that are indoubt. The user can interactively commit, roll back, or forget the indoubt transactions.

The two-phase commit protocol comprises:

1. The PREPARE phase, in which the resource manager writes the log pages to disk, so that it can respond to either a COMMIT or a ROLLBACK primitive
2. The COMMIT (or ROLLBACK) phase, in which the transaction is actually committed or rolled back.

An indoubt transaction is one which has been prepared, but not yet committed or rolled back.

#### Scope

This command only affects the node on which it is executed.

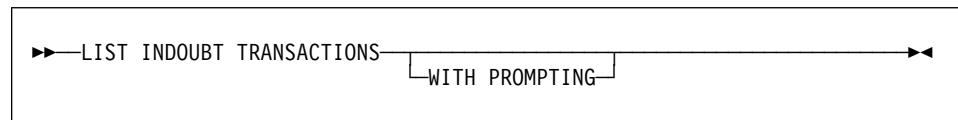
#### Authorization

*dbadm*

#### Required Connection

Database. If implicit connect is enabled, a connection to the default database is established.

#### Command Syntax



#### Command Parameters

##### WITH PROMPTING

Indicates that indoubt transactions are to be processed. If this parameter is specified, an interactive dialog mode is initiated, permitting the user to commit, roll back, or forget indoubt transactions. If this parameter is not specified, indoubt transactions are written to the standard output device, and the interactive dialog mode is not initiated.

Interactive dialog mode permits the user to:

- List all indoubt transactions (enter 1)
- List indoubt transaction number *x* (enter 1, followed by a valid transaction number)
- Quit (enter q)
- Commit transaction number *x* (enter c, followed by a valid transaction number)

## LIST INDOUBT TRANSACTIONS

- Roll back transaction number *x* (enter *r*, followed by a valid transaction number)
- Forget transaction number *x* (enter *f*, followed by a valid transaction number).

**Note:** A blank space must separate the command letter from its argument.

Before a transaction is committed, rolled back, or forgotten, the transaction data is displayed, and the user is asked to confirm the action.

### Example

The following is sample dialog generated by LIST INDOUBT TRANSACTIONS:

In-doubt Transactions for Database SAMPLE

```
1.  originator: XA
    appl_id: *LOCAL.DB2.95051815165159      sequence_no: 0001 status: i
timestamp: 05-18-1997 16:51:59 auth_id: SMITH log_full: n type: RM
xid: 53514C2000000017 00000000544D4442 00000000002F93DD A92F8C4FF3000000
0000BD
```

```
2.  originator: XA
    appl_id: *LOCAL.DATABASE.950407161043 sequence_no: 0002 status: i
timestamp: 04-07-1997 16:10:43 auth_id: JONES log_full: n type: RM
xid: 53514C2000000017 00000000544D4442 00000000002F95FE B62F8C4FF3000000
0000C1
```

```
.
.
.
```

Enter in-doubt transaction command or 'q' to quit.

e.g. 'c 1' heuristically commits transaction 1.

c/r/f/l/q: c 1

```
1.  originator: XA
    appl_id: *LOCAL.DB2.95051815165159      sequence_no: 0001 status: i
timestamp: 05-18-1997 16:51:59 auth_id: SMITH log_full: n type: RM
xid: 53514C2000000017 00000000544D4442 00000000002F93DD A92F8C4FF3000000
0000BD
```

Do you want to heuristically commit this in-doubt transaction ? (y/n) y

DB20000I "COMMIT INDOUBT TRANSACTION" completed successfully

c/r/f/l/q: c 5

DB20030E "5" is not a valid in-doubt transaction number.

c/r/f/l/q: 1

In-doubt Transactions for Database SAMPLE

```
1.  originator: XA
    appl_id: *LOCAL.DB2.95051815165159      sequence_no: 0001 status: c
timestamp: 05-18-1997 16:51:59 auth_id: SMITH log_full: n type: RM
```

## LIST INDOUBT TRANSACTIONS

xid: 53514C2000000017 00000000544D4442 00000000002F93DD A92F8C4FF3000000  
0000BD

2. originator: XA  
appl\_id: \*LOCAL.DATABASE.950407161043 sequence\_no: 0002 status: i  
timestamp: 04-07-1997 16:10:43 auth\_id: JONES log\_full: n type: RM  
xid: 53514C2000000017 00000000544D4442 00000000002F95FE B62F8C4FF3000000  
0000C1

.  
.  
.

c/r/f/l/q: r 2

2. originator: XA  
appl\_id: \*LOCAL.DATABASE.950407161043 sequence\_no: 0002 status: i  
timestamp: 04-07-1997 16:10:43 auth\_id: JONES log\_full: n type: RM  
xid: 53514C2000000017 00000000544D4442 00000000002F95FE B62F8C4FF3000000  
0000C1

Do you want to heuristically rollback this in-doubt transaction ? (y/n) y

DB20000I "ROLLBACK INDOUBT TRANSACTION" completed successfully

c/r/f/l/q: l 2

2. originator: XA  
appl\_id: \*LOCAL.DATABASE.950407161043 sequence\_no: 0002 status: r  
timestamp: 04-07-1997 16:10:43 auth\_id: JONES log\_full: n type: RM  
xid: 53514C2000000017 00000000544D4442 00000000002F95FE B62F8C4FF3000000  
0000C1

c/r/f/l/q: f 2

2. originator: XA  
appl\_id: \*LOCAL.DATABASE.950407161043 sequence\_no: 0002 status: r  
timestamp: 04-07-1997 16:10:43 auth\_id: JONES log\_full: n type: RM  
xid: 53514C2000000017 00000000544D4442 00000000002F95FE B62F8C4FF3000000  
0000C1

Do you want to forget this in-doubt transaction ? (y/n) y

DB20000I "FORGET INDOUBT TRANSACTION" completed successfully

c/r/f/l/q: l 2

2. originator: XA  
appl\_id: \*LOCAL.DATABASE.950407161043 sequence\_no: 0002 status: f  
timestamp: 04-07-1997 16:10:43 auth\_id: JONES log\_full: n type: RM  
xid: 53514C2000000017 00000000544D4442 00000000002F95FE B62F8C4FF3000000  
0000C1

c/r/f/l/q: q

## LIST INDOUBT TRANSACTIONS

### Usage Notes

An indoubt transaction is a global transaction that was left in an indoubt state. This occurs when either the Transaction Manager (TM) or at least one Resource Manager (RM) becomes unavailable after successfully completing the first phase (that is, the PREPARE phase) of the two-phase commit protocol. The RMs do not know whether to commit or to roll back their branch of the transaction until the TM can consolidate its own log with the indoubt status information from the RMs when they again become available.

If LIST INDOUBT TRANSACTIONS is issued against the currently connected database, the command returns the information on indoubt transactions in that database.

Only transactions whose status is indoubt (i) can be committed.

Only transactions whose status is indoubt (i) or ended (e) can be rolled back.

Only transactions whose status is committed (c) or rolled back (r) can be forgotten.

Indoubt transaction information is valid only at the time that the command is issued. Once in interactive dialog mode, transaction status may change because of external activities. If this happens, and an attempt is made to process an indoubt transaction which is no longer in an appropriate state, an error message is displayed.

After this type of error occurs, the user should quit (q) the interactive dialog and reissue the LIST INDOUBT TRANSACTIONS WITH PROMPTING command to refresh the information shown.

For more information, see the *Administration Guide*.

---

**LIST NODE DIRECTORY**

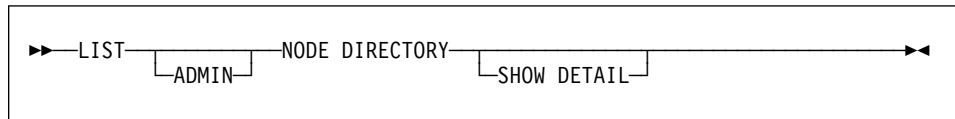
Lists the contents of the node directory.

**Authorization**

None

**Required Connection**

None

**Command Syntax****Command Parameters****ADMIN**

Specifies administration server nodes.

**SHOW DETAIL**

Specifies that the output should include the following information:

- Remote instance name
- System
- Operating system type

**Example**

The following is sample output from LIST NODE DIRECTORY:

Node Directory

Number of entries in the directory = 6

Node 1 entry:

Node name	= DB2APPC1
Comment	= A remote APPC node
Protocol	= APPC
Symbolic destination name	= db2inst1
Security type	= PROGRAM

Node 2 entry:

Node name	= DB2IPX1
Comment	= A remote IPX/SPX node
Protocol	= IPXSPX
File server name	= netwsrv
Bindery object name	= db2inst1

## LIST NODE DIRECTORY

Node 3 entry:

Node name	= DB2IPX2
Comment	= IPX/SPX node using direct addr
Protocol	= IPXSPX
File server name	= *
Bindery object name	= 09212700.400011527745.879E

Node 4 entry:

Node name	= DB2NETB1
Comment	= A remote NetBIOS node
Protocol	= NETBIOS
Adapter number	= 0
Server NNAME	= DB2INST1

Node 5 entry:

Node name	= DB2TCP1
Comment	= A remote TCP/IP node
Protocol	= TCPIP
Hostname	= tcphost
Service name	= db2inst1

Node 6 entry:

Node name	= DB2TCP2
Comment	= TCP/IP node using IP address
Protocol	= TCPIP
Hostname	= 9.21.15.235
Service name	= db2inst2

The common fields are identified as follows:

### Node name

The name of the remote node. This corresponds to the name entered for the *nodename* parameter when the node was cataloged.

### Comment

A comment associated with the node, entered when the node was cataloged. To change a comment in the node directory, uncatalog the node, and then catalog it again with the new comment.

### Protocol

The communications protocol cataloged for the node.

**Note:** For information about fields associated with a specific node type, see the applicable CATALOG...NODE command.

## Usage Notes

A node directory is created and maintained on each database client. It contains an entry for each remote workstation having databases that the client can access. The DB2 client uses the communication end point information in the node directory whenever a database connection or instance attachment is requested.



## LIST NODE DIRECTORY

The database manager creates a node entry and adds it to the node directory each time it processes a CATALOG...NODE command. The entries can vary, depending on the communications protocol being used by the node.

The node directory can contain entries for the following types of nodes:

- APPC
- APPCLU
- APPN
- IPX/SPX
- Local
- Named pipe
- NetBIOS
- TCP/IP.

### See Also

“CATALOG APPC NODE” on page 111

“CATALOG APPCLU NODE” on page 114

“CATALOG APPN NODE” on page 116

“CATALOG IPX/SPX NODE” on page 126

“CATALOG LOCAL NODE” on page 129

“CATALOG NAMED PIPE NODE” on page 131

“CATALOG NETBIOS NODE” on page 133

“CATALOG TCP/IP NODE” on page 136.

## LIST NODEGROUPS

---

### LIST NODEGROUPS

Lists all nodegroups associated with the current database.

#### Scope

This command can be issued from any node that is listed in `$HOME/sq11ib/db2nodes.cfg`. It returns the same information from any of these nodes.

#### Authorization

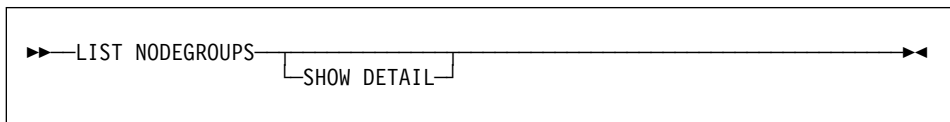
For the system catalogs `SYSCAT.NODEGROUPS` and `SYSCAT.NODEGROUPDEF`, one of the following is required:

- *sysadm* or *dbadm* authority
- CONTROL privilege
- SELECT privilege.

#### Required Connection

Database

#### Command Syntax



#### Command Parameters

##### SHOW DETAIL

Specifies that the output should include the following information:

- Partitioning map ID
- Node number
- In-use flag

#### Examples

Following is sample output from the LIST NODEGROUPS command:

```
NODEGROUP NAME
-----
IBMCATGROUP
IBMDEFAULTGROUP
IBMTEMPGROUP

3 record(s) selected.
```

## LIST NODEGROUPS

Following is sample output from the LIST NODEGROUPS SHOW DETAIL command:

NODEGROUP NAME	PMAP_ID	NODE NUMBER	IN_USE
IBMCATGROUP	0		0 Y
IBMDEFAULTGROUP	1		0 Y

2 record(s) selected.

The fields are identified as follows:

### NODEGROUP NAME

The name of the nodegroup. The name is repeated for each node in the nodegroup.

### PMAP\_ID

The ID of the partitioning map. The ID is repeated for each node in the nodegroup.

### NODE NUMBER

The number of the node.

### IN\_USE

One of four values:

Y The node is being used by the nodegroup.

D The node is going to be dropped from the nodegroup as a result of a REDISTRIBUTE NODEGROUP operation. When the operation completes, the node will not be included in reports from LIST NODEGROUPS.

A The node has been added to the nodegroup but is not yet added to the partitioning map. The containers for the table spaces in the nodegroup have been added on this node. The value is changed to Y when the REDISTRIBUTE NODEGROUP operation completes successfully.

T The node has been added to the nodegroup, but is not yet added to the partitioning map. The containers for the table spaces in the nodegroup have not have been added on this node. Table space containers must be added on the new node for each table space in the node group. The value is changed to A when containers have successfully been added.

For more information, see the *SQL Reference*.

## See Also

“REDISTRIBUTE NODEGROUP” on page 312.

## LIST NODES

---

### LIST NODES

Lists all nodes associated with the current database.

#### Scope

This command can be issued from any node that is listed in `$HOME/sqllib/db2nodes.cfg`. It returns the same information from any of these nodes.

#### Authorization

None

#### Required Connection

Database

#### Command Syntax

```
▶▶—LIST NODES—▶▶
```

#### Command Parameters

None

#### Examples

Following is sample output from the LIST NODES command:

```
NODE NUMBER
-----
                0
                2
                5
                7
                9

5 record(s) selected.
```

#### See Also

“REDISTRIBUTE NODEGROUP” on page 312.

**LIST ODBC DATA SOURCES**

Lists all available user or system ODBC data sources.

A *data source*, in ODBC (Open Database Connectivity) terminology, is a user-defined name for a specific database or file system. That name is used to access the database or file system through ODBC APIs. On Windows NT and Windows 95, either user or system data sources can be cataloged. A user data source is only visible to the user who cataloged it, whereas a system data source is visible to and can be used by all other users.

This command is available on Windows NT, Windows 95, and Windows 3.1 only.

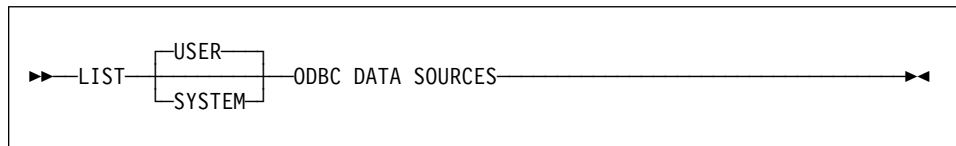
**Authorization**

None

**Required Connection**

None

**Command Syntax**



**Command Parameters**

**USER**

List only user ODBC data sources. This is the default if no keyword is specified.

**SYSTEM**

List only system ODBC data sources.

**Example**

The following is sample output from the LIST ODBC DATA SOURCES command:

User ODBC Data Sources	
Data source name	Description
-----	
SAMPLE	IBM DB2 ODBC DRIVER

## LIST ODBC DATA SOURCES

### See Also

“CATALOG ODBC DATA SOURCE” on page 135

“UNCATALOG ODBC DATA SOURCE” on page 374.

---

## LIST PACKAGES/TABLES

Lists packages or tables associated with the current database.

### Authorization

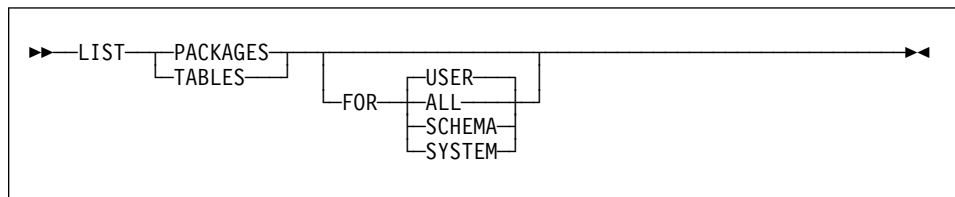
For the system catalogs SYSCAT.PACKAGES (LIST PACKAGES) and SYSCAT.TABLES (LIST TABLES), one of the following is required:

- *sysadm* or *dbadm* authority
- CONTROL privilege
- SELECT privilege.

### Required Connection

Database. If implicit connect is enabled, a connection to the default database is established.

### Command Syntax



### Command Parameters

#### FOR

If the FOR clause is not specified, the packages or tables for USER are listed.

#### ALL

Lists all packages or tables in the database.

#### SCHEMA

Lists all packages or tables in the database for the specified schema only.

#### SYSTEM

Lists all system packages or tables in the database.

#### USER

Lists all user packages or tables in the database for the current user.

## LIST PACKAGES/TABLES

### Examples

The following is sample output from LIST PACKAGES:

Package	Schema	Bound by	Total sections	Valid	Format	Isolation level	Blocking
P1	SMITH	SMITH		1 Yes	0	CS	U

1 record(s) selected.

The following is sample output from LIST TABLES:

Table/View	Schema	Type	Creation time
DEPARTMENT	SMITH	T	1997-02-19-13.32.25.971890
EMP_ACT	SMITH	T	1997-02-19-13.32.27.851115
EMP_PHOTO	SMITH	T	1997-02-19-13.32.29.953624
EMP_RESUME	SMITH	T	1997-02-19-13.32.37.837433
EMPLOYEE	SMITH	T	1997-02-19-13.32.26.348245
ORG	SMITH	T	1997-02-19-13.32.24.478021
PROJECT	SMITH	T	1997-02-19-13.32.29.300304
SALES	SMITH	T	1997-02-19-13.32.42.973739
STAFF	SMITH	T	1997-02-19-13.32.25.156337

9 record(s) selected.

### Usage Notes

LIST PACKAGES and LIST TABLES commands are available to provide a quick Version 1 interface to the system tables. However, Version 2 system tables offer a greater granularity of information, and should be used whenever possible.

The following Version 2 SELECT statements return similar information. They can be expanded to select the additional information that Version 2 provides.

```
select tabname, tabschema, type, create_time
from syscat.tables
order by tabschema, tabname;
```

```
select pkgname, pkgschema, boundby, total_sect,
       valid, format, isolation, blocking
from syscat.packages
order by pkgschema, pkgname;
```

```
select tabname, tabschema, type, create_time
from syscat.tables
where tabschema = 'SYSCAT'
order by tabschema, tabname;
```



## LIST PACKAGES/TABLES

```
select pkgname, pkgschema, boundby, total_sect,  
       valid, format, isolation, blocking  
from syscat.packages  
where pkgschema = 'NULLID'  
order by pkgschema, pkgname;
```

```
select tabname, tabschema, type, create_time  
from syscat.tables  
where tabschema = USER  
order by tabschema, tabname;
```

```
select pkgname, pkgschema, boundby, total_sect,  
       valid, format, isolation, blocking  
from syscat.packages  
where pkgschema = USER  
order by pkgschema, pkgname;
```

## LIST TABLESPACE CONTAINERS

---

### LIST TABLESPACE CONTAINERS

Lists containers for the specified table space.

#### Scope

This command returns information only for the node on which it is executed.

#### Authorization

One of the following:

*sysadm*  
*sysctrl*  
*sysmaint*  
*dbadm*

#### Required Connection

Database

#### Command Syntax

```
▶—LIST TABLESPACE CONTAINERS FOR—tablespace-id—[SHOW DETAIL]—▶
```

#### Command Parameters

##### **FOR** *tablespace-id*

An integer that uniquely represents a table space used by the current database. To get a list of all the table spaces used by the current database, use “LIST TABLESPACES” on page 258.

##### **SHOW DETAIL**

If this option is not specified, only the following basic information about each container is provided:

- Container ID
- Name
- Type (file, disk, or path).

If this option is specified, the following additional information about each container is provided:

- Total number of pages
- Number of useable pages
- Accessible (yes or no).

## LIST TABLESPACE CONTAINERS

### Examples

The following is sample output from LIST TABLESPACE CONTAINERS:

```
Tablespace Containers for Tablespace 0
Container ID          = 0
Name                  = /home/smith/smith/NODE0000/SQL00001/SQLT0000.0
Type                  = Path
```

The following is sample output from LIST TABLESPACE CONTAINERS with SHOW DETAIL specified:

```
Tablespace Containers for Tablespace 0
Container ID          = 0
Name                  = /home/smith/smith/NODE0000/SQL00001/SQLT0000.0
Type                  = Path
Total pages          = 895
Useable pages        = 895
Accessible            = Yes
```

### See Also

“LIST TABLESPACES” on page 258.

## LIST TABLESPACES

---

### LIST TABLESPACES

Lists table spaces for the current database.

#### Scope

This command returns information only for the node on which it is executed.

#### Authorization

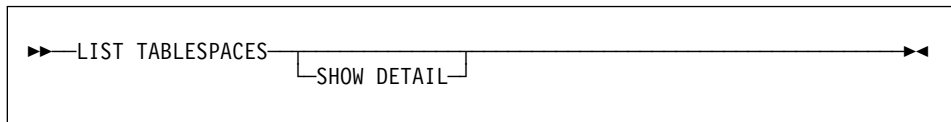
One of the following:

*sysadm*  
*sysctrl*  
*sysmaint*  
*dbadm*

#### Required Connection

Database

#### Command Syntax



#### Command Parameters

##### SHOW DETAIL

If this option is not specified, only the following basic information about each table space is provided:

- Table space ID
- Name
- Type (system managed space or database managed space)
- Contents (any data, long data only, or temporary data)
- State, a hexadecimal value indicating the current table space state.

The externally visible state of a table space is composed of the hexadecimal sum of certain state values. For example, if the state is "quiesced: EXCLUSIVE" and "Load pending", the value is 0x0004 + 0x0008, which is 0x000c. "db2tbst - Get Tablespace State" on page 59 can be used to obtain the table space state associated with a given hexadecimal value. Following are the bit definitions listed in `sqlutil.h`:

0x0	Normal
0x1	Quiesced: SHARE
0x2	Quiesced: UPDATE
0x4	Quiesced: EXCLUSIVE
0x8	Load pending
0x10	Delete pending

## LIST TABLESPACES

0x20	Backup pending
0x40	Roll forward in progress
0x80	Roll forward pending
0x100	Restore pending
0x100	Recovery pending (not used)
0x200	Disable pending
0x400	Reorg in progress
0x800	Backup in progress
0x1000	storage must be defined
0x2000	Restore in progress
0x2000000	storage may be defined
0x4000000	storDef is in 'final' state
0x8000000	storDef was changed prior to rollforward
0x10000000	dms rebalancer is active
0x20000000	TBS deletion in progress
0x40000000	TBS creation in progress
0x8	For service use only

If this option is specified, the following additional information about each table space is provided:

- Total number of pages
- Number of useable pages
- Number of used pages
- Number of free pages
- High water mark (in pages)
- Page size (in bytes)
- Extent size (in pages)
- Prefetch size (in pages)
- Number of containers
- Minimum recovery time (displayed only if not zero)
- State change table space ID (displayed only if the table space state is "load pending" or "delete pending")
- State change object ID (displayed only if the table space state is "load pending" or "delete pending")
- Number of quiescers (displayed only if the table space state is "quiesced: SHARE", "quiesced: UPDATE", or "quiesced: EXCLUSIVE")
- Table space ID and object ID for each quiescer (displayed only if the number of quiescers is greater than zero).

### Example

The following is sample output from LIST TABLESPACES SHOW DETAIL.

```
Tablespaces for Current Database
Tablespace ID          = 0
Name                   = SYSCATSPACE
Type                   = System managed space
Contents               = Any data
State                  = 0x0000
Detailed explanation:
  Normal
Total pages            = 895
```

## LIST TABLESPACES

```
Useable pages           = 895
Used pages              = 895
Free pages              = Not applicable
High water mark (pages) = Not applicable
Page size (bytes)      = 4096
Extent size (pages)    = 32
Prefetch size (pages)  = 32
Number of containers    = 1

Tablespace ID          = 1
Name                   = TEMPSPACE1
Type                   = System managed space
Contents               = Temporary data
State                  = 0x0000
  Detailed explanation:
    Normal
Total pages            = 1
Useable pages          = 1
Used pages             = 1
Free pages             = Not applicable
High water mark (pages) = Not applicable
Page size (bytes)     = 4096
Extent size (pages)   = 32
Prefetch size (pages) = 32
Number of containers   = 1

Tablespace ID          = 2
Name                   = USERSPACE1
Type                   = System managed space
Contents               = Any data
State                  = 0x000c
  Detailed explanation:
    Quiesced: EXCLUSIVE
    Load pending
Total pages            = 337
Useable pages          = 337
Used pages             = 337
Free pages             = Not applicable
High water mark (pages) = Not applicable
Page size (bytes)     = 4096
Extent size (pages)   = 32
Prefetch size (pages) = 32
Number of containers   = 1
State change tablespace ID = 2
State change object ID  = 3
Number of quiescers    = 1
  Quiescer 1:
    Tablespace ID      = 2
    Object ID          = 3
```

DB21011I In a partitioned database server environment, only the table spaces on the current node are listed.

### Usage Notes

In a multi-node environment, this command does not return all the table spaces in the database. To obtain a list of all the table spaces, query `SYSCAT.SYSTABLESPACES`.

## LIST TABLESPACES

### See Also

“db2tbst - Get Tablespace State” on page 59

“LIST TABLESPACE CONTAINERS” on page 256.

## LOAD

---

### LOAD

Loads data from files, tapes, or named pipes into a DB2 table.

### Scope

This command only affects the node on which it is executed.

In a multi-node environment, this command can be used only with ASC or DEL files. IXF files can be loaded only if the table exists on a single node nodegroup.

### Authorization

One of the following:

*sysadm*  
*dbadm*

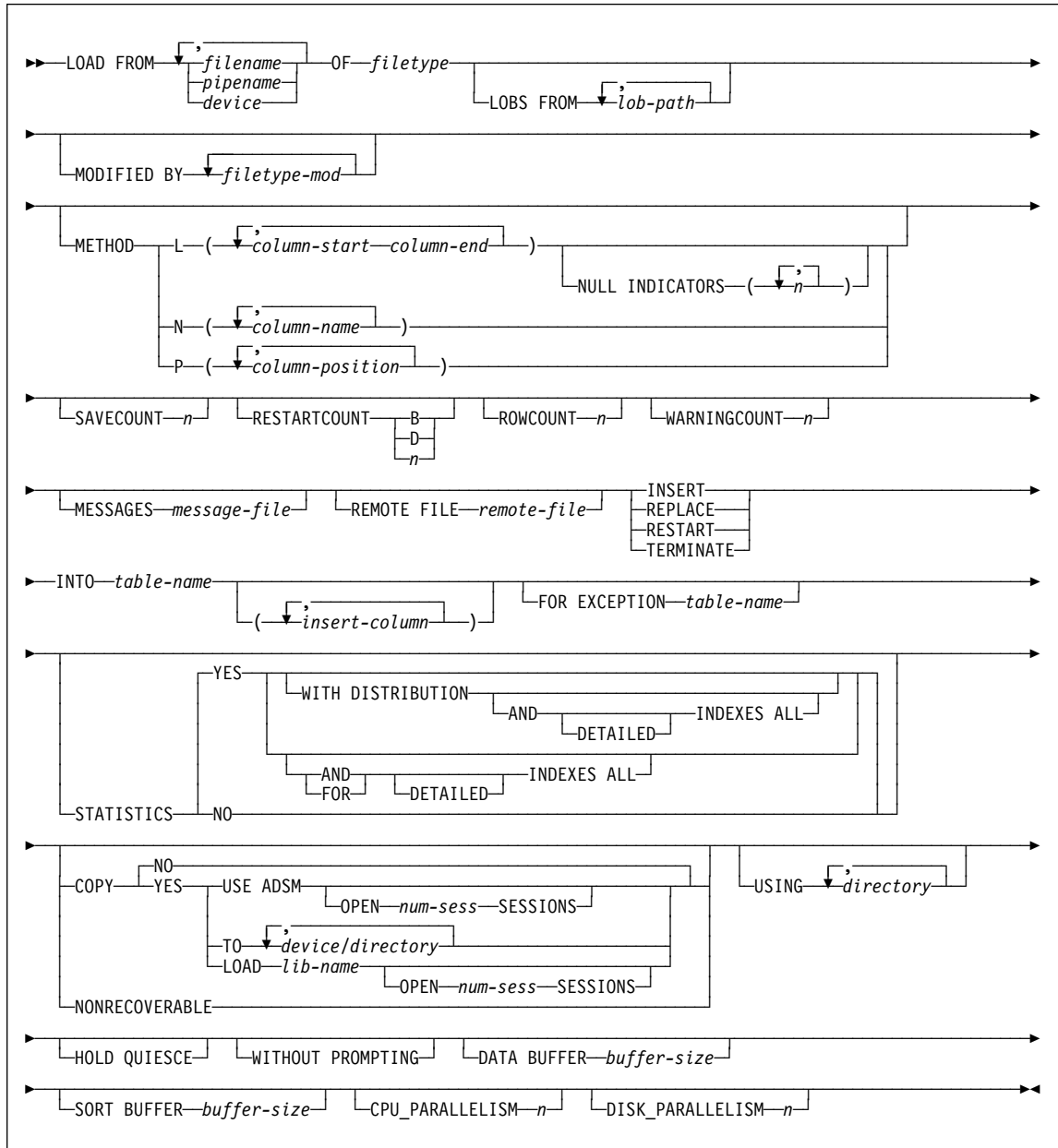
### Required Connection

Database. If implicit connect is enabled, a connection to the default database is established.

Instance. An explicit attachment is not required. If a connection to the database has been established, an implicit attachment to the local instance is attempted.



## Command Syntax



# LOAD

## Command Parameters

### **FROM** *filename/pipe/device*

Specifies the file, pipe, or device that contains the data being loaded. This file/pipe/device must reside on the node where the database resides. If several names are specified, they will be processed in sequence. If the last item specified is a tape device, the user is prompted for another tape. Valid response options are:

- c** Continue. Continue using the device that generated the warning message (for example, when a new tape has been mounted)
- d** Device terminate. Stop using the device that generated the warning message (for example, when there are no more tapes)
- t** Terminate. Terminate all devices.

### **Notes:**

1. It is recommended that the fully qualified file name be used. If the server is remote, the fully qualified file name must be used. If the database resides on the same node as the caller, relative paths may be used.
2. Loading data from multiple IXF files is supported if the files are physically separate, but logically one file. It is *not* supported if the files are both logically and physically separate.

### **OF** *filetype*

Specifies the format of the data in the input file:

- ASC (non-delimited ASCII format)
- DEL (delimited ASCII format)
- IXF (integrated exchange format, PC version), exported from the same or from another DB2 table.

For more information about file formats, see Appendix C, “IMPORT/EXPORT/LOAD Utility File Formats” on page 399.

### **LOBS FROM** *lob-path*

The path to the data files containing LOB values to be loaded. The path must end with a slash (/). The names of the LOB data files are stored in the main data file (ASC, DEL, or IXF), in the column that will be loaded into the LOB column. This option is ignored if *lobsinfile* is not specified within the *filetype-mod* string.

### **MODIFIED BY** *filetype-mod*

Specifies additional information unique to the ASC, DEL, or IXF file format (see page 273).

### **METHOD**

**L**

Specifies the start and end column numbers from which to load data.

**Note:** This method can only be used with ASC files, and is the only valid option for that file type.

*N*

Specifies the names of the columns in the data file to be loaded. The case of these column names must match the case of the corresponding names in the system catalogs. Each column in the table that is not nullable should be included in this list. Specify only complete subsets of column names (for example, given file columns F1, F2, F3, F4, F5, and F6, and table columns C1 INT, C2 INT NOT NULL, C3 INT NOT NULL, and C4 INT, method *N* (F1,F2,F3,F4) insert into table\_name (C1,C2,C3,C4) is a valid request, while method *N* (F1,F4) is not valid, since there will be no data to put into C3.

**Note:** This method can only be used with IXF files.

*P*

Specifies the numbers of the columns to be loaded. Each column in the table that is not nullable should be included in this list. Specify only complete subsets of column numbers (for example, given file columns F1, F2, F3, F4, F5, and F6, and table columns C1 INT, C2 INT NOT NULL, C3 INT NOT NULL, and C4 INT, method *P* (1,2,3,4) is a valid request, while method *P* (1,4) is not valid.

**Note:** This method can only be used with IXF or DEL files, and is the only valid option for the DEL file type.

#### NULL INDICATORS *n*

Specifies a column (by number) to be used as a NULL indicator field. If this option is used, a NULL indicator column for each data column must also be specified. A value of zero indicates that the data column is not nullable, and that there will always be data in that column.

A value of *Y* in the NULL indicator column specifies that the column data is NULL. Any character *other than Y* in the NULL indicator column specifies that the column data is not NULL, and that column data specified by the METHOD *L* option will be loaded.

The NULL indicator character can be changed using the MODIFIED BY option (see page 273).

#### SAVECOUNT *n*

Specifies that LOAD is to establish consistency points after every *n* rows. This value is converted to a page count, and rounded up to intervals of the extent size. Since a message is issued at each consistency point, this option should be selected if the load will be monitored using "LOAD QUERY" on page 280. If the value of *n* is not sufficiently high, the synchronization of activities performed at each consistency point will impact performance.

The default value is 0, meaning that no consistency points will be established, unless necessary.

# LOAD

## RESTARTCOUNT

*B*

LOAD will restart at the build phase.

*D*

LOAD will restart at the delete phase.

*n*

An integer specifying that the load is to be started at record  $n+1$ . The first  $n$  records are skipped.

This option can be specified with any of the INSERT, REPLACE, or RESTART modes. B or D must not be specified for the INSERT or the REPLACE mode.

## ROWCOUNT *n*

Specifies the number of  $n$  physical records in the file to be loaded. Allows a user to load only the first  $n$  rows in a file.

## WARNINGCOUNT *n*

Stops the load after  $n$  warnings. Set this parameter if no warnings are expected, but verification that the correct file and table are being used is desired. If  $n$  is 0, or this option is not specified, the load will continue regardless of the number of warnings issued.

If the load is stopped because the threshold of warnings was encountered, another load can be started in RESTART mode by specifying the RESTARTCOUNT option. Alternatively, another load can be initiated in REPLACE mode, starting at the beginning of the input file.

## MESSAGES *message-file*

Specifies the destination for warning and error messages that occur during the load. If a message file is not specified, messages are written to a file in the current directory.

If the complete path to the file is not specified, LOAD uses the current directory and the default drive as the destination.

If the name of a file that already exists is specified, LOAD appends the information.

## REMOTE FILE *remote-file*

Specifies the base name to be used when creating temporary files during a load, and should be fully qualified according to the server node. For more information about remote files, see page 278).

## INSERT

One of four modes under which the load utility can execute. Adds the loaded data to the table without changing the existing table data.

## REPLACE

One of four modes under which the load utility can execute. Deletes all existing data from the table, and inserts the loaded data. The table definition and index definitions are not changed.

**RESTART**

One of four modes under which the load utility can execute. Restarts LOAD after a previous load was interrupted.

It is important to keep track of the last commit point. This information is stored in the message file and is passed to LOAD. Use “LOAD QUERY” on page 280 to get this information if the database connection was lost during the load.

**TERMINATE**

One of four modes under which the load utility can execute. Terminates a previously interrupted load and moves the table spaces in which the table resides from load pending state to recovery pending state. The table spaces cannot be used until a backup has been restored and the table spaces have been rolled forward. A restart should be issued before attempting to complete an interrupted load.

**Note:** This option is not recommended for general use; it should only be selected if an unrecoverable error has occurred.

**INTO** *table-name*

Specifies the table within the database into which the data is to be loaded. The table cannot be a system catalog table. An alias, or the fully qualified or unqualified table name can be specified. A qualified table name is in the form *schema.tablename*. If an unqualified table name is specified, the table will be qualified with the current authorization ID.

*insert-column*

Specifies the name of a column within the table into which the data is to be inserted.

**FOR EXCEPTION** *table-name*

Specifies the exception table into which rows in error will be copied. Any row that is in violation of a unique index or a primary key index is copied.

**STATISTICS YES**

Specifies that statistics will be gathered for the table and for any existing indexes. This option is not supported if the load is in INSERT or in RESTART mode.

*WITH DISTRIBUTION*

Specifies that distribution statistics are requested.

*AND INDEXES ALL*

Update statistics for both the table and its indexes.

*FOR INDEXES ALL*

Update statistics for the indexes only.

*DETAILED*

Specifies that extended index statistics are requested.

**STATISTICS NO**

Specifies that no statistics will be gathered, and that the statistics in the catalogs will not be altered.

**COPY NO**

Specifies that the table space in which the table resides will be placed in backup pending state if forward recovery is enabled (that is, *logretain* or

## LOAD

*userexit* is on). The data will not be accessible until a table space backup or a full database backup is made.

### **COPY YES**

Specifies that a copy of the changes made will be saved. This option is invalid if forward recovery is disabled (both *logretain* and *userexit* are off).

### **USE ADSM**

Specifies that the copy will be stored using ADSM.

### **OPEN *num-sess* SESSIONS**

The number of I/O sessions to be used with ADSM or the vendor product. The default value is 1.

### **TO *device/directory***

Specifies the device or directory on which the copy image will be created.

### **LOAD *lib-name***

The name of the shared library (DLL on OS/2 or the Windows operating system) containing the vendor backup and restore I/O functions to be used. It may contain the full path. If the full path is not given, it will default to the path where the user exit programs reside.

### **NONRECOVERABLE**

Specifies that the load transaction is to be marked as non-recoverable, and that it will not be possible to recover it by a subsequent rollforward action. The rollforward utility will skip the transaction, and will mark the table into which data was being loaded as "invalid". The utility will also ignore any subsequent transactions against that table. After the roll forward is completed, such a table can only be dropped.

With this option, table spaces are not put in backup pending state following the load operation, and a copy of the loaded data does not have to be made during the load.

### **USING *directory***

Temporary files are used when indexes are created. The directories in which these temporary files are created can be specified. Otherwise, the files are created in the `sql1ib/tmp` directory of the home directory of the **DB2INSTANCE** owner. Ensure that there is enough space on these directories to hold all index keys for the data being loaded.

If more than one directory is specified, each directory should be on a different file system, and each file system should be on a different disk, to optimize performance.

### **HOLD QUIESCE**

Specifies that the utility should leave the table in quiesced exclusive state after the load.

### **WITHOUT PROMPTING**

Specifies that the list of data files contains all the files that are to be loaded, and that the devices/directories listed are sufficient for the entire load. If a continuation input file is not found, or the copy targets are filled before the load finishes, the load will fail, and the table will remain in load pending state.

If this option is not specified, and the tape device encounters an end of tape for the copy image, or the last item listed is a tape device, the user is prompted for a new tape on that device.

**DATA BUFFER** *buffer-size*

Specifies the number of 4KB pages (regardless of the degree of parallelism) to use as buffered space for transferring data within the utility. If the value specified is less than the algorithmic minimum, the minimum required resource is used, and no warning is returned.

This memory is allocated directly from the utility heap, whose size can be modified through the *util\_heap\_sz* database configuration parameter.

If a value is not specified, an intelligent default is calculated by the utility at run time. The default is based on a percentage of the free space available in the utility heap at the instantiation time of the loader, as well as some characteristics of the table.

**SORT BUFFER** *buffer-size*

Use this parameter to specify the number of 4KB pages of memory that are to be used for sorting the index keys during a load operation.

**Note:** Sort buffer size has a very large impact on sort performance. Therefore, for very large tables (for example, tables in excess of 100M), this buffer should be set as large as possible.

If a value is not specified, the utility uses the larger of:

- 2MB for OS/2 or Windows NT, or 6MB for all other platforms
- The minimum size allowed by the sort algorithm
- 15% of the free space remaining in the utility heap.

If a value greater than zero, but less than the required minimum is specified, the minimum value for that load is returned.

**CPU\_PARALLELISM** *n*

Specifies the number of processes or threads that the load utility will spawn for parsing, converting and formatting records when building table objects. This parameter is designed to exploit SMP parallelism. It is particularly useful when loading presorted data, because record order in the source data is preserved. If the value of this parameter is zero, the load utility uses an intelligent default value at run time.

**Note:** If this parameter is used with tables containing either LOB or LONG VARCHAR fields, its value becomes one, regardless of the number of system CPUs or the value specified by the user.

**DISK\_PARALLELISM** *n*

Specifies the number of processes or threads that the load utility will spawn for writing data to the table space containers. If a value is not specified, the utility selects an intelligent default based on the number of table space containers and the characteristics of the table.

# LOAD

## Example

### Example 1

TABLE1 has 5 columns:

```
COL1 VARCHAR 20 NOT NULL WITH DEFAULT
COL2 SMALLINT
COL3 CHAR 4
COL4 CHAR 2 NOT NULL WITH DEFAULT
COL5 CHAR 2 NOT NULL
```

ASCFILE1 has 6 elements:

```
ELE1 positions 01 to 20
ELE2 positions 21 to 22
ELE5 positions 23 to 23
ELE3 positions 24 to 27
ELE4 positions 28 to 31
ELE6 positions 32 to 32
ELE6 positions 33 to 40
```

Data Records:

```
1...5...10...15...20...25...30...35...40
Test data 1      XXN 123abcdN
Test data 2 and 3  QQY   wxyzN
Test data 4,5 and 6 WVN6789   Y
```

The following command loads the table from the file:

```
db2 load from ascfile1 of asc modified by T reclen=40
method L (1 20, 21 22, 24 27, 28 31)
null indicators (0,0,23,32)
insert into table1 (col1, col5, col2, col3)
```

### Note:

The specification of T in the MODIFIED BY parameter forces the truncation of blanks in VARCHAR columns (COL1, for example, which is 11, 17 and 19 bytes long, in rows 1, 2 and 3, respectively).

The specification of reclen=40 in the MODIFIED BY parameter indicates that there is no new-line character at the end of each input record, and that each record is 40 bytes long. The last 8 bytes are not used to load the table.

Since COL4 is not provided in the input file, it will be inserted into TABLE1 with its default value (it is defined NOT NULL WITH DEFAULT).

Positions 23 and 32 are used to indicate whether COL2 and COL3 of TABLE1 will be loaded NULL for a given row. If there is a Y in the column's null indicator position for a given record, the column will be NULL. If there is an N, the data values in the column's data positions of the input record (as defined in L(.....))



are used as the source of column data for the row. In this example, neither column in row 1 is NULL; COL2 in row 2 is NULL; and COL3 in row 3 is NULL.

In this example, the NULL INDICATORS for COL1 and COL5 are specified as 0 (zero), indicating that the data is not nullable.

The NULL INDICATOR for a given column can be anywhere in the input record, but the position must be specified, and the Y or N values must be supplied.

### Example 2 (Loading LOBs from Files)

TABLE1 has 3 columns:

```
COL1 CHAR 4 NOT NULL WITH DEFAULT
LOB1 LOB
LOB2 LOB
```

ASCFILE1 has 3 elements:

```
ELE1 positions 01 to 04
ELE2 positions 06 to 13
ELE3 positions 15 to 22
```

The following files reside in either /u/user1 or /u/user1/bin:

```
ASCFILE2 has LOB data
ASCFILE3 has LOB data
ASCFILE4 has LOB data
ASCFILE5 has LOB data
ASCFILE6 has LOB data
ASCFILE7 has LOB data
```

Data Records in ASCFILE1:

```
1...5...10...15...20...25...30.
REC1 ASCFILE2 ASCFILE3
REC2 ASCFILE4 ASCFILE5
REC3 ASCFILE6 ASCFILE7
```

The following command loads the table from the file:

```
db2 load from ascfile1 of asc
lobs from /u/user1, /u/user1/bin
modified by lobsinfile reflen=22
method L (1 4, 6 13, 15 22)
insert into table1
```

#### Note:

The specification of lobsinfile in the MODIFIED BY parameter tells the loader that all LOB data is to be loaded from files.

## LOAD

The specification of `recLen=22` in the `MODIFIED BY` parameter indicates that there is no new-line character at the end of each input record, and that each record is 22 bytes long.

LOB data is contained in 6 files, `ASCFILE2` through `ASCFILE7`. Each file contains the data that will be used to load a LOB column for a specific row. The relationship between LOBs and other data is specified in `ASCFILE1`. The first record of this file tells the loader to place `REC1` in `COL1` of row 1. The contents of `ASCFILE2` will be used to load `LOB1` of row 1, and the contents of `ASCFILE3` will be used to load `LOB2` of row 1. Similarly, `ASCFILE4` and `ASCFILE5` will be used to load `LOB1` and `LOB2` of row 2, and `ASCFILE6` and `ASCFILE7` will be used to load the LOBs of row 3.

The `LOBS FROM` parameter contains 2 paths that will be searched for the named LOB files when those files are required by the loader.

To load LOBs directly from `ASCFILE1` (a non-delimited ASCII file), without the `LOBSINFILE` option, the following rules must be observed:

- The total length of any record, including LOBs, cannot exceed 32K.
- LOB fields in the input records must be of fixed length, and LOB data padded with blanks as necessary.
- The `stripblanks` option of `MODIFIED BY` must be specified, so that the trailing blanks used to pad LOBs can be removed as the LOBs are inserted into the database.

### Usage Notes

Data is loaded in the sequence that appears in the input file. If a particular sequence is desired, the data should be sorted before `LOAD` is executed.

`LOAD` builds indexes based on existing definitions. The exception tables are used to handle duplicates on unique keys. `LOAD` does not perform referential integrity or constraint checking. If these are included in the table definition, the tables are placed in check pending state, and the user must either force the check flag, or execute the `SET CONSTRAINTS` statement.

## File Type Modifications

**Note:** The load utility does not issue a warning if an attempt is made to use unsupported file types with the MODIFIED BY option. If this is attempted, the load fails, and an error code is returned.

Table 9 (Page 1 of 6). Valid File Type Modifications (LOAD)	
Modification	Description
<b>All File Formats</b>	
anyorder	This modifier is used in conjunction with the <i>cpu_parallelism</i> parameter. Specifies that the preservation of source data order is not required, yielding significant additional performance benefit on SMP systems. If the value of <i>cpu_parallelism</i> is 1, this option is ignored. This option is not supported if SAVECOUNT > 0, since crash recovery after a consistency point requires that data be loaded in sequence.
fastparse	<p>Reduced data checking is done on user-supplied column values, and performance is enhanced. Tables loaded under this option are guaranteed to be architecturally correct, and the utility is guaranteed to perform sufficient data checking to prevent a segmentation violation or trap. Data that is in correct form will be loaded correctly.</p> <p>This option does not affect referential integrity checking or constraints checking; it merely reduces syntax checking of the supplied data. For example, if the value 123qwr4 were encountered as a field entry for an integer column in an ASC file, the load utility would ordinarily flag a syntax error, since the value does not represent a valid number. With FASTPARSE, a syntax error is not detected, and an arbitrary number is loaded into the integer field. Care must be taken to use this modifier with clean data only. Performance improvements using this option with ASCII data can be quite substantial. FASTPARSE does not significantly enhance performance with PC/IXF data, since IXF is a binary format, and FASTPARSE affects parsing and conversion from ASCII to internal forms.</p>
lobsinfile	<i>lob-path</i> specifies the path to the files containing LOB values. The ASC, DEL, or IXF load input files contain the names of the files having LOB data in the LOB column.
nodefaults	<p>If a source column for a target table column is not explicitly specified, and the table column is not nullable, default values are not loaded. Without this option, if a source column for one of the target table columns is not explicitly specified, one of the following occurs:</p> <ul style="list-style-type: none"> <li>• If the column is defaultable, the default value is loaded</li> <li>• If the column is nullable and not defaultable, a NULL is loaded</li> <li>• If the column is not nullable and not defaultable, an error is returned, and the utility stops processing.</li> </ul>

## LOAD

Table 9 (Page 2 of 6). Valid File Type Modifications (LOAD)

Modification	Description
noheader	<p>Skips the header verification code.</p> <p>"db2split - Data Declustering Tool" on page 53 writes a header to each file contributing data to a table in a multi-node nodegroup. The header includes the node number, the partitioning map, and the partitioning key specification. The load utility requires this information to verify that the data is being loaded at the right node. When loading files into a table that exists on a single-node nodegroup, the headers do not exist, and this option causes the load utility to skip the header verification code.</p>
norowwarnings	Suppresses all warnings about rejected rows.
pagefreespace= <i>x</i>	<p><i>x</i> is an integer between 0 and 100 inclusive. The value is interpreted as the percentage of each data page that is to be left as free space.</p> <p>If the specified value is invalid because of the minimum row size, (for example, a row that is at least 3000 bytes long, and an <i>x</i> value of 50), the row will be placed on a new page. If a value of 100 is specified, each row will reside on a new page.</p>
totalfreespace= <i>x</i>	<p><i>x</i> is an integer between 0 and 100 inclusive. The value is interpreted as the percentage of the total pages in the table that is to be appended to the end of the table as free space. For example, if <i>x</i> is 20, and the table has 100 data pages, 20 additional empty pages will be appended. The total number of data pages for the table will be 120.</p>
usedefaults	<p>If a source column for a target table column has been specified, but it contains no data for one or more row instances, default values are loaded. Examples of missing data are:</p> <ul style="list-style-type: none"> <li>• For DEL files: ",," is specified for the column</li> <li>• For DEL/ASC/WSF files: A row that does not have enough columns, or is not long enough for the original specification.</li> </ul> <p>Without this option, if a source column contains no data for a row instance, one of the following occurs:</p> <ul style="list-style-type: none"> <li>• If the column is nullable, a NULL is loaded</li> <li>• If the column is not nullable, the utility rejects the row.</li> </ul>
<b>ASCII File Formats (ASC/DEL)</b>	

Table 9 (Page 3 of 6). Valid File Type Modifications (LOAD)

Modification	Description
codepage=x	<p>x is an ASCII character string. The value is interpreted as the code page of the data in the input data set. Converts character data (and numeric data specified in characters) from this code page to the database code page during the load operation.</p> <p>The following rules apply:</p> <ul style="list-style-type: none"> <li>• For pure DBCS (graphic), mixed DBCS, and EUC, delimiters are restricted to the range of x00 to x3F, inclusive.</li> <li>• For DEL data specified in an EBCDIC code page, the delimiters may not coincide with the shift-in and shift-out DBCS characters.</li> <li>• nullindchar must specify symbols included in the standard ASCII set between code points x20 and x7F, inclusive. This refers to ASCII symbols and code points. EBCDIC data can use the corresponding symbols, even though the code points will be different.</li> </ul>
dumpfile = x	<p>x is the fully qualified (according to the server node) name of an exception file to which rejected rows are written. A maximum of 32K of data is written per record. For example,</p> <pre>db2 load from data of del   modified by dumpfile = /u/user/filename   insert into table_name</pre>
implieddecimal	<p>The location of an implied decimal point is determined by the column definition; it is no longer assumed to be at the end of the value. For example, the value 12345 is loaded into a DECIMAL(8,2) column as 123.45, <i>not</i> 12345.00.</p>
noeofchar	<p>The optional end-of-file character x'1A' is not recognized as the end of file. Processing continues as if it were a normal character.</p>
<b>ASC (Non-delimited ASCII) File Format</b>	

## LOAD

Table 9 (Page 4 of 6). Valid File Type Modifications (LOAD)

Modification	Description
binarynumerics	<p>Numeric (but not DECIMAL) data must be in binary form, not the character representation. This avoids costly conversions.</p> <p>This option is supported only with positional ASC, using fixed length records specified by the reclen option. The noeofchar option is assumed.</p> <p>The following rules apply:</p> <ul style="list-style-type: none"> <li>• No conversion between data types is performed, with the exception of INTEGER and SMALLINT.</li> <li>• Data lengths must match their target column definitions.</li> <li>• FLOATs must be in IEEE Floating Point format.</li> <li>• Binary data must be in the binary format of the platform on which the load is performed.</li> </ul> <p><b>Note:</b> NULLs cannot be present in the data for columns affected by this modifier. Blanks (normally interpreted as NULL) are interpreted as a binary value when this modifier is used.</p> <p>Regardless of the server platform, the byte order of binary data in the load source file is assumed to be big-endian; that is, when using this modifier on OS/2 or on the Windows operating system, the byte order must not be reversed.</p>
nullindchar=x	<p>x is a single character. Changes the character denoting a null value to x. The default value of x is Y.<sup>b</sup></p>
packeddecimal	<p>Loads packed-decimal data directly, since the binarynumerics modifier does not include the DECIMAL field type.</p> <p>This option is supported only with positional ASC, using fixed length records specified by the reclen option. The noeofchar option is assumed.</p> <p>Supported values for the sign nibble are:</p> <pre>+ = 0xC 0xA 0xE 0xF - = 0xD 0xB</pre> <p><b>Note:</b> NULLs cannot be present in the data for columns affected by this modifier. Blanks (normally interpreted as NULL) are interpreted as a binary value when this modifier is used.</p> <p>Regardless of the server platform, the byte order of binary data in the load source file is assumed to be big-endian; that is, when using this modifier on OS/2 or on the Windows operating system, the byte order must not be reversed.</p>
reclen=x	<p>x is an integer with a maximum value of 32767. x characters are read for each row, and a new-line character is not used to indicate the end of the row.</p>

<i>Table 9 (Page 5 of 6). Valid File Type Modifications (LOAD)</i>	
<b>Modification</b>	<b>Description</b>
striptblanks	<p>Truncates any trailing blank spaces when loading data into a variable-length field. If this option is not specified, blank spaces are kept.</p> <p>This option cannot be specified together with striptnulls. These are mutually exclusive options.</p> <p><b>Note:</b> This option replaces the obsolete t option, which is supported for back-level compatibility only.</p>
striptnulls	<p>Truncates any trailing NULLs (0x00 characters) when loading data into a variable-length field. If this option is not specified, NULLs are kept.</p> <p>This option cannot be specified together with striptblanks. These are mutually exclusive options.</p> <p><b>Note:</b> This option replaces the obsolete padwithzero option, which is supported for back-level compatibility only.</p>
<b>DEL (Delimited ASCII) File Format</b>	
chardelx	<p>x is a single character string delimiter. The default value is a double quotation mark ("). The specified character is used in place of double quotation marks to enclose a character string.<sup>ab</sup></p> <p>The single quotation mark (') can also be specified as a character string delimiter as follows:</p> <p style="text-align: center;">modified by charde1''</p>
coldelx	<p>x is a single character column delimiter. The default value is a comma (.). The specified character is used in place of a comma to signal the end of a column.<sup>ab</sup></p>
decptx	<p>x is a single character substitute for the period as a decimal point character. The default value is a period (.). The specified character is used in place of a period as a decimal point character.<sup>ab</sup></p>
<b>IXF File Format</b>	
forcein	<p>Directs the utility to accept data despite code page mismatches, and to suppress translation between code pages.</p> <p>Fixed length target fields are checked to verify that they are large enough for the data. If nochecklengths is specified, no checking is done, and an attempt is made to load each row.</p>

## LOAD

Table 9 (Page 6 of 6). Valid File Type Modifications (LOAD)

Modification	Description
<b>Note:</b>	
<ul style="list-style-type: none"><li><sup>a</sup> Table 7 on page 165 lists the characters that can be used as delimiter overrides.</li><li><sup>b</sup> The character must be specified in the code page of the source data.</li></ul>	
The character code point (instead of the character symbol), can be specified using the syntax xJJ or 0xJJ, where JJ is the hexadecimal representation of the code point. For example, to specify the # character as a column delimiter, use one of the following:	
... modified by colde1# ...	
... modified by colde10x23 ...	
... modified by colde1X23 ...	

### Remote Files

Remote file is a base file name to which DB2 appends different extensions to create files used by other commands (for example, .msg for "LOAD QUERY" on page 280).

The remote file resides on the server machine and is accessed by the DB2 instance exclusively. Therefore, it is imperative that any file name qualification given to this parameter reflects the directory structure of the server, not the client, and that the DB2 instance owner has read and write permission on this file. In addition, the user must ensure that two loads are not issued that have the same fully-qualified remote file name.

There are several ways that the remote file name can be selected and qualified when the user has just given a partially qualified name, or no name at all:

- No remote file name is given in a load operation where the user is on the same machine as the database instance. In this case, the load utility will use the name *db2utmp* and qualify it with the current working directory of the user. Two loads from the same directory with this option will clash on the use of the remote file name, therefore this option is not recommended.
- No remote file name is given in a load operation, where the user is on a different machine than the database instance. In this case, the load utility will generate a name that will reside in the database directory. This effectively prevents the user from using the load query facility, since it requires the name of the remote file. In addition, the file name generated is not guaranteed to be unique, and therefore clashes may occur between different load operations. Therefore this option is not recommended.
- A non-fully-qualified file name is given in a load operation, where the user is on the same machine as the database instance. In this case the name is qualified by using the current directory of the user. The user must ensure that two loads are not issued from the same directory with the same remote file name.



## LOAD

- A non-fully-qualified file name is given in a load operation, where the user is on a different machine than the database instance. In this case the load utility will reject the file name. It must be fully qualified from the client.
- A fully-qualified file name is given in a load operation. This will be the file name used. The user must ensure that two loads are not issued with the same remote file name. This is the recommended usage.

**Note:** In an MPP system, the remote file must reside on a local disk, not on an NFS mount. If the file is on an NFS mount, there will be a significant performance decrement during the load operation.

### See Also

“LOAD QUERY” on page 280

“QUIESCE TABLESPACES FOR TABLE” on page 305.

## LOAD QUERY

---

### LOAD QUERY

Checks the status of LOAD during processing.

#### Authorization

None

#### Required Connection

Database

#### Command Syntax

```
▶—LOAD QUERY—remote-file—┐
                              └─TO—local-message-file—▶
```

#### Command Parameters

*remote-file*

Specifies the base name that was used when creating temporary files during a load.

**TO** *local-message-file*

Specifies the destination for warning and error messages that occur during the load. This file cannot be the *message-file* specified for LOAD. If the file already exists, all messages that the load utility has generated are appended to it.

#### Example

A user loading a large amount of data into the STAFF table wants to check the status of the load. Assuming that the LOAD parameter *remote-file* was set as:

```
remote file /u/remotedir/rmsg/staff
```

the following commands are issued:

```
db2 connect to <database>
```

```
db2 load query /u/remotedir/rmsg/staff to /u/mydir/staff.tempmsg
```

## LOAD QUERY

The output file `/u/mydir/staff.tempsmsg` might look like the following:

```
SQL3500W The utility is beginning the "LOAD" phase at time
"02-13-1997 19:40:29.645353".

SQL3519W  Begin Load Consistency Point. Input record count = "0".

SQL3520W  Load Consistency Point was successful.

SQL3109N The utility is beginning to load data from file
"/u/mydir/data/staffbig.ixf".

SQL3150N The H record in the PC/IXF file has product "DB2 01.00",
date "19970111", and time "194554".

SQL3153N The T record in the PC/IXF file has name
"data/staffbig.ixf", qualifier " ", and source " ".

SQL3519W  Begin Load Consistency Point. Input record count =
"111152".

SQL3520W  Load Consistency Point was successful.

SQL3519W  Begin Load Consistency Point. Input record count =
"222304".

SQL3520W  Load Consistency Point was successful.
```

### See Also

"LOAD" on page 262.

## MIGRATE DATABASE

---

### MIGRATE DATABASE

Converts previous versions of DB2 databases to current formats. Following are the database releases that are supported in the DB2 V5.0 database migration process:

- DB2 for OS/2 Version 1.x and Version 2.x to Version 5.0
- DB2 for AIX Version 1.x and Version 2.x to Version 5.0
- DB2 for HP-UX Version 2.x to Version 5.0
- DB2 for Solaris Version 2.x to Version 5.0
- DB2 for Windows NT Version 2.x to Version 5.0
- DB2 Parallel Edition Version 1.x to Version 5.0.

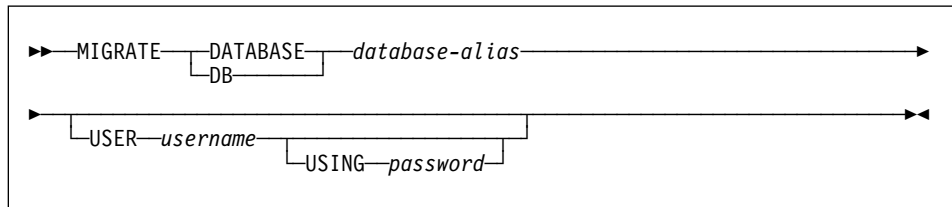
### Authorization

*sysadm*

### Required Connection

This command establishes a database connection.

### Command Syntax



### Command Parameters

**DATABASE** *database-alias*

Specifies the alias of the database to be migrated to the currently installed version of the database manager.

**USER** *username*

Identifies the user name under which the database is to be migrated.

**USING** *password*

The password used to authenticate the user name. If the password is omitted, but a user name was specified, the user is prompted to enter it.

### Example

The following example migrates the database cataloged under the database alias `sales`:

```
db2 migrate database sales
```

### Usage Notes

This command will only migrate a database to a newer version, and cannot be used to convert a migrated database to its previous version.

The database must be cataloged before migration.

Use “db2ckmig - Database Pre-migration Tool” on page 15 to determine whether the database can be migrated. For more information, see one of the *Quick Beginnings* books.

**Note:** Back up all databases prior to migration.

## PRECOMPILE PROGRAM

---

### PRECOMPILE PROGRAM

Processes an application program source file containing embedded SQL statements. A modified source file is produced, containing host language calls for the SQL statements and, by default, a package is created in the database.

#### Scope

This command can be issued from any node in `db2nodes.cfg`. It updates the database catalogs on the catalog node. Its effects are visible to all nodes.

#### Authorization

One of the following:

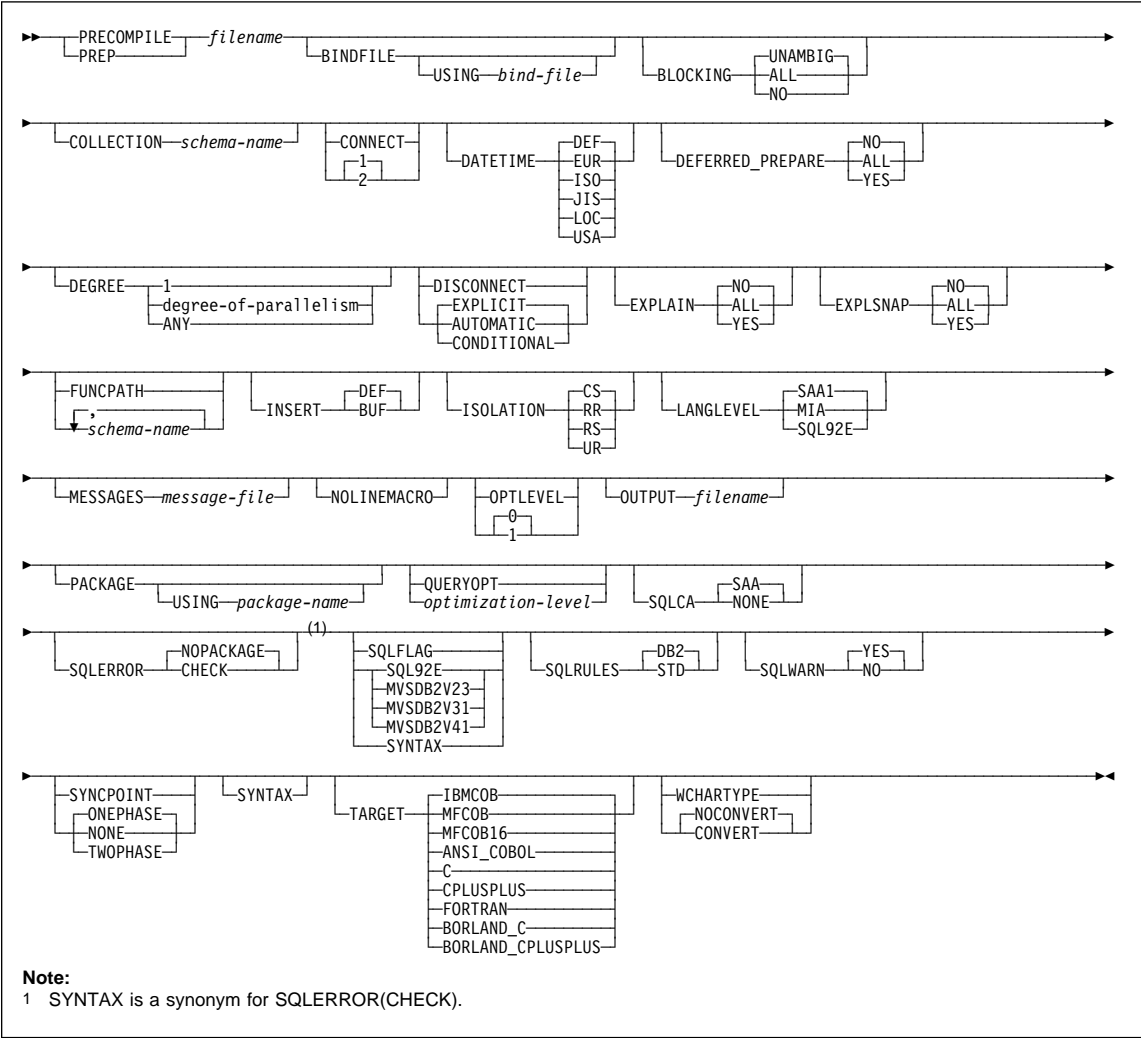
- *sysadm* or *dbadm* authority
- BINDADD privilege if a package does not exist, and one of:
  - IMPLICIT\_SCHEMA authority on the database if the schema name of the package does not exist
  - CREATEIN privilege on the schema if the schema name of the package exists
- ALTERIN privilege on the schema if the package exists
- BIND privilege on the package if it exists.

The user also needs all privileges required to compile any static SQL statements in the application. Privileges granted to groups are not used for authorization checking of static statements. If the user has *sysadm* authority, but not explicit privileges to complete the bind, the database manager grants explicit *dbadm* authority automatically.

#### Required Connection

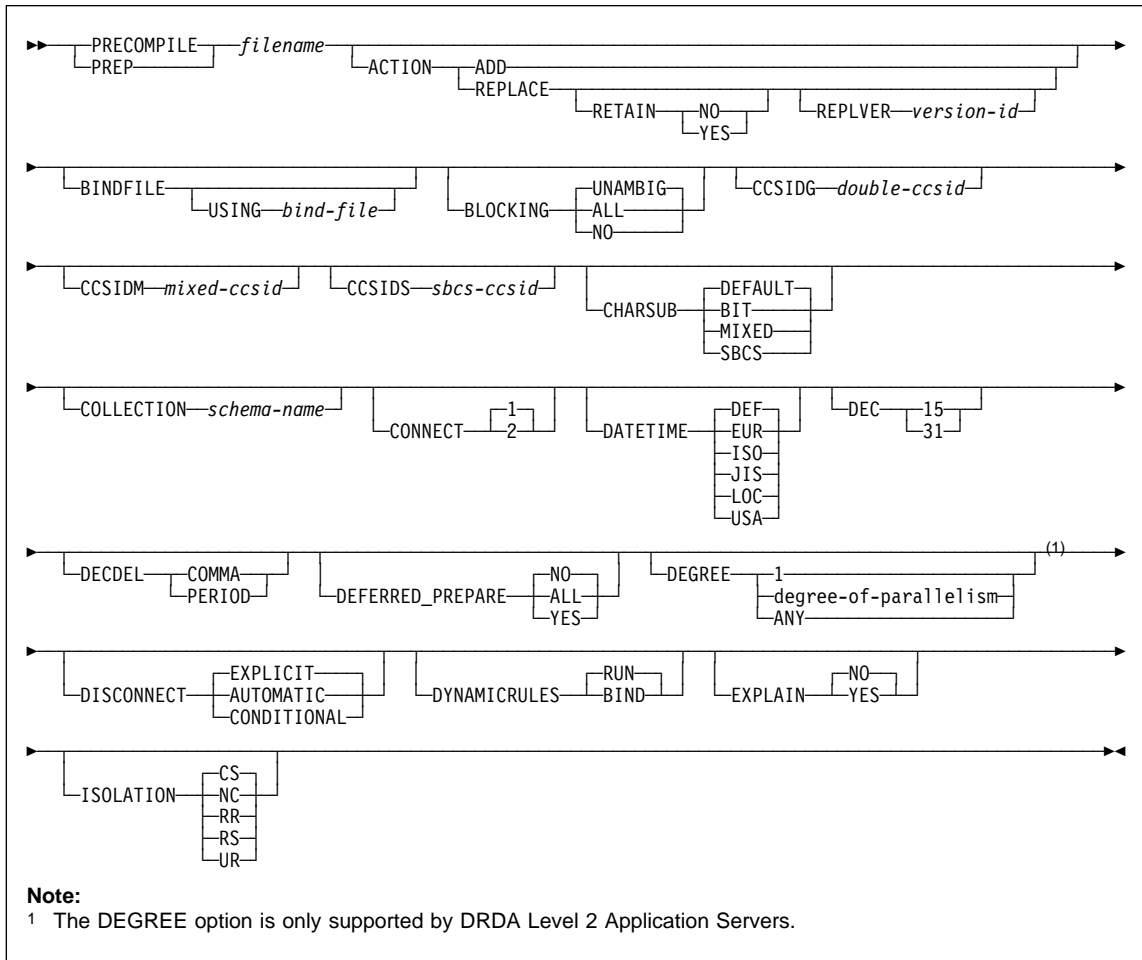
Database. If implicit connect is enabled, a connection to the default database is established.

**Command Syntax  
For DB2**



# PRECOMPILE PROGRAM

## For DRDA

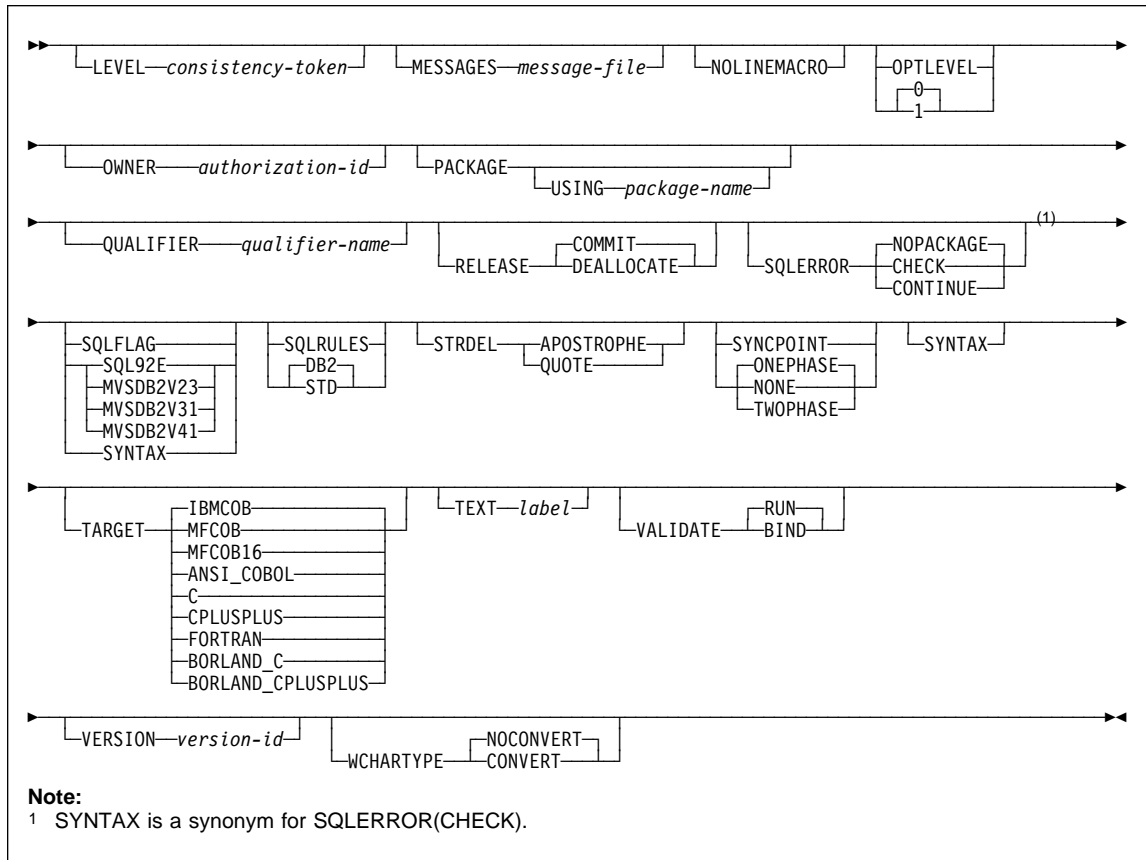


Continued on next page



# PRECOMPILE PROGRAM

DRDA—Continued



## Command Parameters

*filename*

Specifies the source file to be precompiled. An extension of:

- .sqc must be specified for C applications (generates a .c file)
- .sqx (OS/2 or the Windows operating system), or .sqc (UNIX based systems) must be specified for C++ applications (generates a .cxx file on OS/2 or the Windows operating system, or a .C file on UNIX based systems)
- .sqb must be specified for COBOL applications (generates a .cb1 file)
- .sqf must be specified for FORTRAN applications (generates a .for file on OS/2 or the Windows operating system, or a .f file on UNIX based systems).

## ACTION

Indicates whether the package can be added or replaced. This DRDA precompile/bind option is not supported by DB2.

## PRECOMPILE PROGRAM

### *ADD*

Indicates that the named package does not exist, and that a new package is to be created. If the package already exists, execution stops, and a diagnostic error message is returned.

### *REPLACE*

Indicates that the old package is to be replaced by a new one with the same location, collection, and package name.

### *RETAIN*

Indicates whether EXECUTE authorities are to be preserved when a package is replaced. If ownership of the package changes, the new owner grants the BIND and EXECUTE authority to the previous package owner.

### *NO*

Does not preserve EXECUTE authorities when a package is replaced.

### *YES*

Preserves EXECUTE authorities when a package is replaced.

### *REPLVER version-id*

Replaces a specific version of a package. The version identifier specifies which version of the package is to be replaced. Maximum length is 254 characters.

## **BINDFILE**

Results in the creation of a bind file. A package is not created unless the **package** option is also specified. If a bind file is requested, but no package is to be created, as in the following example:

```
db2 prep sample.sqc bindfile
```

object existence and authentication SQLCODEs will be treated as warnings instead of errors. This will allow a bind file to be successfully created, even if the database being used for precompilation does not have all of the objects referred to in static SQL statements within the application. The bind file can be successfully bound, creating a package, once the required objects have been created.

### *USING bind-file*

The name of the bind file that is to be generated by the precompiler. The file name must have an extension of `.bnd`. If a file name is not entered, the precompiler uses the name of the program (entered as the `filename` parameter), and adds the `.bnd` extension. If a path is not provided, the bind file is created in the current directory.

### BLOCKING

For information about row blocking, see the *Administration Guide* or the *Embedded SQL Programming Guide*.

#### *ALL*

Specifies to block for:

- Read-only cursors
- Cursors not specified as FOR UPDATE OF

Ambiguous cursors are treated as read-only.

#### *NO*

Specifies not to block any cursors. Ambiguous cursors are treated as updateable.

#### *UNAMBIG*

Specifies to block for:

- Read-only cursors
- Cursors not specified as FOR UPDATE OF

Ambiguous cursors are treated as updateable.

### **CCSIDG** *double-ccsid*

An integer specifying the coded character set identifier (CCSID) to be used for double byte characters in character column definitions (without a specific CCSID clause) in CREATE and ALTER TABLE SQL statements. This DRDA precompile/bind option is not supported by DB2. The DRDA server will use a system defined default value if this option is not specified.

### **CCSIDM** *mixed-ccsid*

An integer specifying the coded character set identifier (CCSID) to be used for mixed byte characters in character column definitions (without a specific CCSID clause) in CREATE and ALTER TABLE SQL statements. This DRDA precompile/bind option is not supported by DB2. The DRDA server will use a system defined default value if this option is not specified.

### **CCSIDS** *sbcsc-ccsid*

An integer specifying the coded character set identifier (CCSID) to be used for single byte characters in character column definitions (without a specific CCSID clause) in CREATE and ALTER TABLE SQL statements. This DRDA precompile/bind option is not supported by DB2. The DRDA server will use a system defined default value if this option is not specified.

### CHARSUB

Designates the default character sub-type that is to be used for column definitions in CREATE and ALTER TABLE SQL statements. This DRDA precompile/bind option is not supported by DB2.

#### *BIT*

Use the FOR BIT DATA SQL character sub-type in all new character columns for which an explicit sub-type is not specified.

#### *DEFAULT*

Use the target system defined default in all new character columns for which an explicit sub-type is not specified.

## PRECOMPILE PROGRAM

### *MIXED*

Use the FOR MIXED DATA SQL character sub-type in all new character columns for which an explicit sub-type is not specified.

### *SBCS*

Use the FOR SBCS DATA SQL character sub-type in all new character columns for which an explicit sub-type is not specified.

### **COLLECTION** *schema-name*

Specifies an 8-character collection identifier for the package. If not specified, the authorization identifier for the user processing the package is used.

### **CONNECT**

*1*

Specifies that a CONNECT statement is to be processed as a type 1 CONNECT.

*2*

Specifies that a CONNECT statement is to be processed as a type 2 CONNECT.

### **DATETIME**

Specifies the date and time format to be used. For more information about date and time formats, see the *SQL Reference*.

### *DEF*

Use a date and time format associated with the country code of the database.

### *EUR*

Use the IBM standard for Europe date and time format.

### *ISO*

Use the date and time format of the International Standards Organization.

### *JIS*

Use the date and time format of the Japanese Industrial Standard.

### *LOC*

Use the date and time format in local form associated with the country code of the database.

### *USA*

Use the IBM standard for U.S. date and time format.

### **DEC**

Specifies the maximum precision to be used in decimal arithmetic operations. This DRDA precompile/bind option is not supported by DB2. The DRDA server will use a system defined default value if this option is not specified.

*15*

15-digit precision is used in decimal arithmetic operations.

31

31-digit precision is used in decimal arithmetic operations.

### DECDEL

Designates whether a period (.) or a comma (,) will be used as the decimal point indicator in decimal and floating point literals. This DRDA precompile/bind option is not supported by DB2. The DRDA server will use a system defined default value if this option is not specified.

*COMMA*

Use a comma (,) as the decimal point indicator.

*PERIOD*

Use a period (.) as the decimal point indicator.

### DEFERRED\_PREPARE

Provides a performance enhancement when accessing DB2 common server databases or DRDA databases. This option combines the SQL PREPARE statement flow with the associated OPEN, DESCRIBE, or EXECUTE statement flow to minimize inter-process or network flow.

*NO*

The PREPARE statement will be executed at the time it is issued.

*YES*

Execution of the PREPARE statement will be deferred until the corresponding OPEN, DESCRIBE, or EXECUTE statement is issued.

The PREPARE statement will not be deferred if it uses the INTO clause, which requires an SQLDA to be returned immediately. However, if the PREPARE INTO statement is issued for a cursor that does not use any parameter markers, the processing will be optimized by pre-OPENing the cursor when the PREPARE is executed.

*ALL*

Same as YES, except that a PREPARE INTO statement which contains parameter markers *is* deferred. If a PREPARE INTO statement does not contain parameter markers, pre-OPENing of the cursor will still be performed.

If the PREPARE statement uses the INTO clause to return an SQLDA, the application must not reference the content of this SQLDA until the OPEN, DESCRIBE, or EXECUTE statement is issued and returned.

### DEGREE

Specifies the degree of parallelism for the execution of static SQL statements in an SMP system. This option does not affect CREATE INDEX parallelism.

1

The execution of the statement will not use parallelism.

## PRECOMPILE PROGRAM

### *degree-of-parallelism*

Specifies the degree of parallelism with which the statement can be executed, a value between 2 and 32 767 (inclusive).

### *ANY*

Specifies that the execution of the statement can involve parallelism using a degree determined by the database manager.

## DISCONNECT

### *AUTOMATIC*

Specifies that all database connections are to be disconnected at commit.

### *CONDITIONAL*

Specifies that the database connections that have been marked RELEASE or have no open WITH HOLD cursors are to be disconnected at commit.

### *EXPLICIT*

Specifies that only database connections that have been explicitly marked for release by the RELEASE statement are to be disconnected at commit.

## DYNAMICRULES

Specifies which authorization identifier to use when dynamic SQL in a package is executed. This DRDA precompile/bind option is not supported by DB2.

### *BIND*

Indicates that the authorization identifier used for the execution of dynamic SQL is the package owner.

### *RUN*

Indicates that the authorization identifier used for the execution of dynamic SQL is the *authid* of the person executing the package.

## EXPLAIN

Stores information in the Explain tables about the access plans chosen for each SQL statement in the package. DRDA does not support the ALL value for this option.

### *NO*

Explain information will not be captured.

### *YES*

Explain tables will be populated with information about the chosen access plan.

### *ALL*

Explain information for each eligible static SQL statement will be placed in the Explain tables. In addition, Explain information will be gathered for eligible dynamic SQL statements at run time, even if the CURRENT EXPLAIN SNAPSHOT register is

## PRECOMPILE PROGRAM

set to NO. For more information about special registers, see the *SQL Reference*.

**Note:** This value for EXPLAIN is not supported by DRDA.

### EXPLSNAP

Stores Explain Snapshot information in the Explain tables. This DB2 precompile/bind option is not supported by DRDA.

*NO*

An Explain Snapshot will not be captured.

*YES*

An Explain Snapshot for each eligible static SQL statement will be placed in the Explain tables.

*ALL*

An Explain Snapshot for each eligible static SQL statement will be placed in the Explain tables. In addition, Explain Snapshot information will be gathered for eligible dynamic SQL statements at run time, even if the CURRENT EXPLAIN SNAPSHOT register is set to NO. For more information about special registers, see the *SQL Reference*.

### FUNCPATH

Specifies the function path to be used in resolving user-defined distinct types and functions in static SQL. If this option is not specified, the default function path is "SYSIBM","SYSFUN",USER where USER is the value of the USER special register. This DB2 precompile/bind option is not supported by DRDA.

*schema-name*

A short SQL identifier, either ordinary or delimited, which identifies a schema that exists at the application server. No validation that the schema exists is made at precompile or at bind time. The same schema cannot appear more than once in the function path. The number of schemas that can be specified is limited by the length of the resulting function path, which cannot exceed 254 bytes. The schema SYSIBM does not need to be explicitly specified; it is implicitly assumed to be the first schema if it is not included in the function path. For more information, see the *SQL Reference*.

### INSERT

Allows a program being precompiled or bound from a DB2 V2.1 client to a DATABASE 2 Parallel Edition server to request that data inserts be buffered to increase performance.

*BUF*

Specifies that inserts from an application should be buffered.

*DEF*

Specifies that inserts from an application should not be buffered.

## PRECOMPILE PROGRAM

### ISOLATION

Determines how far a program bound to this package can be isolated from the effect of other executing programs. For more information about isolation levels, see the *SQL Reference*.

*CS*

Specifies Cursor Stability as the isolation level.

*NC*

No Commit. Specifies that commitment control is not to be used. This isolation level is not supported by DB2.

*RR*

Specifies Repeatable Read as the isolation level.

*RS*

Specifies Read Stability as the isolation level. Read Stability ensures that the execution of SQL statements in the package is isolated from other application processes for rows read and changed by the application.

*UR*

Specifies Uncommitted Read as the isolation level.

### LANGLEVEL

Specifies the SQL rules that apply for both the syntax and the semantics for both static and dynamic SQL in the application. This option is not supported by DDCS. For more information about this option, see the *Embedded SQL Programming Guide*.

*MIA*

Select the ISO/ANS SQL92 rules as follows:

- To support checking of SQLCODE or SQLSTATE values, variables by this name may be declared in the host variable declare section (if neither is declared, SQLCODE is assumed during precompilation).
- C null-terminated strings are padded with blanks and always include a null-terminating character, even if truncation occurs.
- The FOR UPDATE clause is optional for all columns to be updated in a positioned UPDATE.
- A searched UPDATE or DELETE requires SELECT privilege on the object table of the UPDATE or DELETE statement if a column of the object table is referenced in the search condition or on the right hand side of the assignment clause.
- A column function that can be resolved using an index (for example MIN or MAX) will also check for nulls and return warning SQLSTATE 01003 if there were any nulls.
- An error is returned when a duplicate unique constraint is included in a CREATE or ALTER TABLE statement.
- An error is returned when no privilege is granted and the grantor has no privileges on the object (otherwise a warning is returned).



## PRECOMPILE PROGRAM

### SAA1

Select the common IBM DB2 rules as follows:

- To support error SQLCODE or SQLSTATE checking, an SQLCA must be declared in the application code.
- C null-terminated strings are not terminated with a null character if truncation occurs.
- The FOR UPDATE clause is required for all columns to be updated in a positioned UPDATE.
- A searched UPDATE or DELETE will not require SELECT privilege on the object table of the UPDATE or DELETE statement unless a fullselect in the statement references the object table.
- A column function that can be resolved using an index (for example MIN or MAX) will not check for nulls and warning SQLSTATE 01003 is not returned.
- A warning is returned and the duplicate unique constraint is ignored.
- An error is returned when no privilege is granted.

### SQL92E

Defines the ISO/ANS SQL92 rules as follows:

- To support checking of SQLCODE or SQLSTATE values, variables by this name may be declared in the host variable declare section (if neither is declared, SQLCODE is assumed during precompilation).
- The behaviors listed under option MIA.

### LEVEL *consistency-token*

Defines the level of a module using the consistency token. The consistency token is any alphanumeric value up to 8 characters in length. The RDB package consistency token verifies that the requester's application and the relational database package are synchronized. This DRDA precompile option is not supported by DB2.

**Note:** This option is not recommended for general use.

### MESSAGES *message-file*

Specifies the destination for warning, error, and completion status messages. A message file is created whether the bind is successful or not. If a message file name is not specified, the messages are written to standard output. If the complete path to the file is not specified, the current directory is used. If the name of an existing file is specified, the contents of the file are overwritten.

### NOLINEMACRO

Suppresses the generation of the #line macros in the output .c file. Useful when the file is used with development tools which require source line information such as profiles, cross-reference utilities, and debuggers.

**Note:** This precompile option is used for the C/C++ programming languages only.

## PRECOMPILE PROGRAM

### OPTLEVEL

Indicates whether the C/C++ precompiler is to optimize initialization of internal SQLDAs when host variables are used in SQL statements. Such optimization can increase performance when a single SQL statement (such as FETCH) is used inside a tight loop.

0

Instructs the precompiler not to optimize SQLDA initialization.

1

Instructs the precompiler to optimize SQLDA initialization. This value should not be specified if the application uses:

- pointer host variables, as in the following example:

```
exec sql begin declare section;
char (*name)[20];
short *id;
exec sql end declare section;
```
- C++ data members directly in SQL statements.

For more information, see the *Embedded SQL Programming Guide*.

### OUTPUT *filename*

Overrides the default name of the modified source file produced by the compiler. It can include a path.

### OWNER *authorization-id*

Designates an 8-character authorization identifier for the package owner. The owner must have the privileges required to execute the SQL statements in the package. The default is the primary authorization ID of the precompile/bind process if this option has not been explicitly specified. This DRDA precompile/bind option is not supported by DB2.

### PACKAGE

Creates a package. If neither **package**, **bindfile**, nor **syntax** is specified, a package is created in the database by default.

#### *USING package-name*

The name of the package that is to be generated by the precompiler. If a name is not entered, the name of the application program source file (minus extension and folded to uppercase) is used. Maximum length is 8 characters.

### QUALIFIER *qualifier-name*

Provides an 18-character implicit qualifier for unqualified table names, views, indexes, and aliases contained in the package. The default is the owner's authorization ID, whether or not **owner** is explicitly specified. This DRDA precompile/bind option is not supported by DB2.

### QUERYOPT *optimization-level*

Indicates the desired level of optimization for all static SQL statements contained in the package. The default value is 5. For the complete range of

## PRECOMPILE PROGRAM

optimization levels available, see the SET CURRENT QUERY OPTIMIZATION statement in the *SQL Reference*. This DB2 precompile/bind option is not supported by DRDA.

### RELEASE

Indicates whether resources are released at each COMMIT point, or when the application terminates. This DRDA precompile/bind option is not supported by DB2.

#### COMMIT

Release resources at each COMMIT point. Used for dynamic SQL statements.

#### DEALLOCATE

Release resources only when the application terminates.

### SQLCA

For FORTRAN applications only. This option is ignored if it is used with other languages.

#### NONE

Specifies that the modified source code is not consistent with the SAA definition.

#### SAA

Specifies that the modified source code is consistent with the SAA definition.

### SQLERROR

Indicates whether to create a package or a bind file if an error is encountered.

#### CHECK

Specifies that the target system performs all syntax and semantic checks on the SQL statements being bound. A package will not be created as part of this process. If, while creating a package, an existing package with the same name and version is encountered, the existing package is neither dropped nor replaced if **action replace** was specified.

#### CONTINUE

A package or a bind file is created even when SQL errors are encountered. This option is not supported by DB2.

#### NOPACKAGE

A package or a bind file is not created if an error is encountered.

### SQLFLAG

Identifies and reports on deviations from the SQL language syntax specified in this option.

A bind file or a package is created only if the **bindfile** or the **package** option is specified, in addition to the **sqlflag** option.

Local syntax checking is performed only if one of the following options is specified:

## PRECOMPILE PROGRAM

- **bindfile**
- **package**
- **sqlerror check**
- **syntax**

If **sqlflag** is not specified, the flagger function is not invoked, and the bind file or the package is not affected.

### *SQL92E SYNTAX*

The SQL statements will be checked against ANSI or ISO SQL92 Entry level SQL language format and syntax with the exception of syntax rules that would require access to the database catalog. Any deviation is reported in the precompiler listing.

### *MVSDB2V23 SYNTAX*

The SQL statements will be checked against MVS DB2 Version 2.3 SQL language syntax. Any deviation from the syntax is reported in the precompiler listing.

### *MVSDB2V31 SYNTAX*

The SQL statements will be checked against MVS DB2 Version 3.1 SQL language syntax. Any deviation from the syntax is reported in the precompiler listing.

### *MVSDB2V41 SYNTAX*

The SQL statements will be checked against MVS DB2 Version 4.1 SQL language syntax. Any deviation from the syntax is reported in the precompiler listing.

## SQLRULES

Specifies whether type 2 CONNECTs are to be processed according to the DB2 rules or the Standard (STD) rules based on ISO/ANS SQL92.

### *DB2*

Allow the use of the SQL CONNECT statement to switch the current connection to another established (*dormant*) connection.

### *STD*

Allow the use of the SQL CONNECT statement to establish a new connection only. The SQL SET CONNECTION statement must be used to switch to a dormant connection.

## SQLWARN

Indicates whether warnings will be returned from the compilation of dynamic SQL statements (via PREPARE or EXECUTE IMMEDIATE), or from describe processing (via PREPARE...INTO or DESCRIBE). This DB2 precompile/bind option is not supported by DRDA.

### *NO*

Warnings will not be returned from the SQL compiler.

### *YES*

Warnings will be returned from the SQL compiler.

## PRECOMPILE PROGRAM

**Note:** SQLCODE +238 is an exception. It is returned regardless of the **sqlwarn** option value.

### STRDEL

Designates whether an apostrophe (') or double quotation marks (") will be used as the string delimiter within SQL statements. This DRDA precompile/bind option is not supported by DB2. The DRDA server will use a system defined default value if this option is not specified.

#### *APOSTROPHE*

Use an apostrophe (') as the string delimiter.

#### *QUOTE*

Use double quotation marks (") as the string delimiter.

### SYNCPOINT

Specifies how commits or rollbacks are to be coordinated among multiple database connections.

#### *NONE*

Specifies that no Transaction Manager (TM) is to be used to perform a two-phase commit, and does not enforce single updater, multiple reader. A COMMIT is sent to each participating database. The application is responsible for recovery if any of the commits fail.

#### *ONEPHASE*

Specifies that no TM is to be used to perform a two-phase commit. A one-phase commit is to be used to commit the work done by each database in multiple database transactions.

#### *TWOPHASE*

Specifies that the TM is required to coordinate two-phase commits among those databases that support this protocol.

### SYNTAX

Suppresses the creation of a package or a bind file during precompilation. This option can be used to check the validity of the source file without modifying or altering existing packages or bind files. **Syntax** is a synonym for **sqlerror check**.

If **syntax** is used together with the **package** option, **package** is ignored.

### TARGET

Instructs the precompiler to produce modified code tailored to one of the supported compilers on the current platform.

#### *IBMCOB*

On AIX, code is generated for the IBM COBOL Set for AIX compiler. On OS/2, code is generated for the IBM VisualAge for COBOL compiler.

#### *MFCOB*

Code is generated for the Micro Focus COBOL compiler. On OS/2 this refers to the 32-bit Micro Focus COBOL compiler. This is the default if a **target** value is not specified with the COBOL precompiler on all UNIX platforms and Windows NT.

## PRECOMPILE PROGRAM

### *MFCOB16*

Code is generated for the 16-bit Micro Focus COBOL compiler. This value is only valid on OS/2, and is the default if a **target** value is not specified with the COBOL precompiler.

### *ANSI\_COBOL*

Code compatible with the ANS X3.23-1985 standard is generated.

### *C*

Code compatible with the C compilers supported by DB2 on the current platform is generated.

### *CPLUSPLUS*

Code compatible with the C++ compilers supported by DB2 on the current platform is generated.

### *BORLAND\_C*

C code is generated for the Borland C/C++ compiler. This value is only valid on OS/2.

### *BORLAND\_CPLUSPLUS*

C++ code is generated for the Borland C/C++ compiler. This value is only valid on OS/2.

### *FORTTRAN*

Code compatible with the Fortran compilers supported by DB2 on the current platform is generated.

### **TEXT** *label*

The description of a package. Maximum length is 255 characters. The default value is blanks. This DRDA precompile/bind option is not supported by DB2.

### **VALIDATE**

Determines when the database manager checks for authorization errors and object not found errors. The package owner authorization ID is used for validity checking. This DRDA precompile/bind option is not supported by DB2.

### *BIND*

Validation is performed at precompile/bind time. If all objects do not exist, or all authority is not held, error messages are produced. If **sqlerror continue** is specified, a package/bind file is produced despite the error message, but the statements in error are not executable.

### *RUN*

Validation is attempted at bind time. If all objects exist, and all authority is held, no further checking is performed at execution time.

If all objects do not exist, or all authority is not held at precompile/bind time, warning messages are produced, and the package is successfully bound, regardless of the **sqlerror continue** option setting. However, authority checking and existence checking for SQL statements that failed these

## PRECOMPILE PROGRAM

checks during the precompile/bind process may be redone at execution time.

### **VERSION** *version-id*

Defines the version identifier for a package. The version identifier is any alphanumeric value, \$, #, @, \_, -, or ., up to 254 characters in length. This DRDA precompile option is not supported by DB2.

### **WCHARTYPE**

For details and restrictions on the use and applicability of **wchartype**, see the *Embedded SQL Programming Guide*.

#### *CONVERT*

Host variables declared using the `wchar_t` base type will be treated as containing data in `wchar_t` format. Since this format is not directly compatible with the format of graphic data stored in the database (DBCS format), input data in `wchar_t` host variables is implicitly converted to DBCS format on behalf of the application, using the ANSI C function `wcstombs()`. Similarly, output DBCS data is implicitly converted to `wchar_t` format, using `mbstowcs()`, before being stored in host variables.

#### *NOCONVERT*

Host variables declared using the `wchar_t` base type will be treated as containing data in DBCS format. This is the format used within the database for graphic data; it is, however, different from the native `wchar_t` format implemented in the C language. Using **noconvert** means that graphic data will not undergo conversion between the application and the database, which can improve efficiency. The application is, however, responsible for ensuring that data in `wchar_t` format is not passed to the database manager. When this option is used, `wchar_t` host variables should not be manipulated with the C wide character string functions, and should not be initialized with wide character literals (*L-literals*).

## Usage Notes

A modified source file is produced, which contains host language equivalents to the SQL statements. By default, a package is created in the database to which a connection has been established. The name of the package is the same as the file name (minus the extension and folded to uppercase), up to a maximum of 8 characters.

Following connection to a database, PREP executes under the transaction that was started. PREP then issues a COMMIT or a ROLLBACK operation to terminate the current transaction and start another one.

Creating a package with a schema name that does not already exist will result in the implicit creation of that schema. The schema owner is SYSIBM. The CREATEIN privilege on the schema is granted to PUBLIC.

## PRECOMPILE PROGRAM

During precompilation, an Explain Snapshot is not taken unless a package is created and **explsnap** has been specified. The snapshot is put into the Explain tables of the user creating the package. Similarly, Explain table information is only captured when **explain** is specified, and a package is created.

Precompiling stops if a fatal error or more than 100 errors occur. If a fatal error does occur, PREP stops precompiling, attempts to close all files, and discards the package.

### See Also

“BIND” on page 98.



---

## PRUNE HISTORY

Deletes entries from the recovery history file.

### Authorization

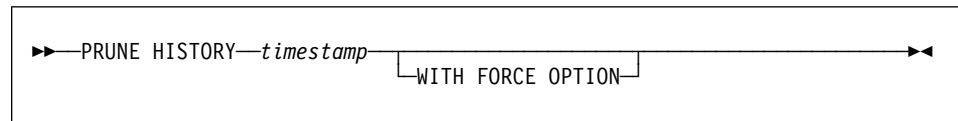
One of the following:

*sysadm*  
*sysctrl*  
*sysmaint*  
*dbadm*

### Required Connection

Database

### Command Syntax



### Command Parameters

*timestamp*

Identifies a range of entries in the recovery history file that will be deleted. A complete time stamp (in the form *yyyymmddhhnnss*), or an initial prefix (minimum *yyyy*) can be specified. All entries with time stamps equal to or less than the time stamp provided are deleted from the recovery history file.

#### WITH FORCE OPTION

Specifies that the entries will be pruned according to the time stamp specified, even if some entries from the most recent restore set are deleted from the file.

### Example

To remove the entries for all restores, loads, table space backups, and full database backups taken before and including December 1, 1994 from the recovery history file, enter:

```
db2 prune history 199412
```

## QUERY CLIENT

---

### QUERY CLIENT

Returns current connection settings for an application process.

#### Authorization

None

#### Required Connection

None

#### Command Syntax

```
▶—QUERY CLIENT—▶
```

#### Command Parameters

None

#### Example

The following is sample output from QUERY CLIENT:

```
The current connection settings of the application process are:
```

```
          CONNECT = 1
          DISCONNECT = EXPLICIT
MAX_NETBIOS_CONNECTIONS = 1
          SQLRULES = DB2
          SYNCPOINT = ONEPHASE
```

#### Usage Notes

The connection settings for an application process can be queried at any time during execution.

For information about distributed unit of work (DUOW), see the *Administration Guide*.

#### See Also

“SET CLIENT” on page 353.

## QUIESCE TABLESPACES FOR TABLE

Quiesces table spaces for a table. There are three valid quiesce modes: share, intent to update, and exclusive. There are three possible states resulting from the quiesce function: QUIESCED SHARE, QUIESCED UPDATE, and QUIESCED EXCLUSIVE.

### Scope

In a single-node environment, this command quiesces all table spaces involved in a load operation in exclusive mode for the duration of the load. In an MPP environment, this command acts locally on a node. It quiesces only that portion of table spaces belonging to the node on which the load is performed.

### Authorization

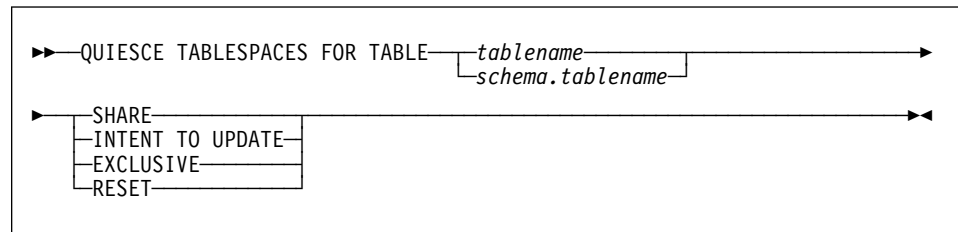
One of the following:

*sysadm*  
*sysctrl*  
*sysmaint*  
*dbadm*

### Required Connection

Database

### Command Syntax



### Command Parameters

#### TABLE

*tablename*

Specifies the unqualified table name. The table cannot be a system catalog table.

*schema.tablename*

Specifies the qualified table name. If *schema* is not provided, the authorization ID used for the database connection will be used as the *schema*. The table cannot be a system catalog table.

#### SHARE

Specifies that quiesce is in share mode.

## QUIESCE TABLESPACES FOR TABLE

### INTENT TO UPDATE

Specifies that quiesce is in intent to update mode.

### EXCLUSIVE

Specifies that quiesce is in exclusive mode.

### RESET

Specifies that the state of the table spaces is reset to normal.

## Examples

```
db2 quiesce tablespaces for table staff share
```

```
db2 quiesce tablespaces for table boss.org intent to update
```

## Usage Notes

When the quiesce share request is received, the transaction requests intent share locks for the table spaces and a share lock for the table. When the transaction obtains the locks, the state of the table spaces is changed to QUIESCED SHARE. The state is granted to the quiescer only if there is no conflicting state held by other users. The state of the table spaces is recorded in the table space table, along with the authorization ID and the database agent ID of the quiescer, so that the state is persistent.

The table cannot be changed while the table spaces for the table are in QUIESCED SHARE state. Other share mode requests to the table and table spaces will be allowed. When the transaction commits or rolls back, the locks are released, but the table spaces for the table remain in QUIESCED SHARE state until the state is explicitly reset.

When the quiesce exclusive request is made, the transaction requests super exclusive locks on the table spaces, and a super exclusive lock on the table. When the transaction obtains the locks, the state of the table spaces changes to QUIESCED EXCLUSIVE. The state of the table spaces, along with the authorization ID and the database agent ID of the quiescer, are recorded in the table space table. Since the table spaces are held in super exclusive mode, no other access to the table spaces is allowed. The user who invokes the quiesce function (the quiescer), however, has exclusive access to the table and the table spaces.

When a quiesce update request is made, the table spaces are locked in intent exclusive (IX) mode, and the table is locked in update (U) mode. The state of the table spaces with the quiescer is recorded in the table space table.

There is a limit of five quiescers on a table space at any given time. Since QUIESCED EXCLUSIVE is incompatible with any other state, and QUIESCED UPDATE is incompatible with another QUIESCED UPDATE, the five quiescer limit, if reached, must have at least four QUIESCED SHARE and at most one QUIESCED UPDATE.

A quiescer can upgrade the state of a table space from a less restrictive state to a more restrictive one (for example, S to U, or U to X). If a user requests a state lower than one that is already held, the original state is returned. States are not downgraded.

## QUIESCE TABLESPACES FOR TABLE

### See Also

“LOAD” on page 262.

## QUIT

---

### QUIT

Exits the command line processor interactive input mode and returns to the operating system command prompt. If a batch file is being used to input commands to the command line processor, commands are processed until QUIT, TERMINATE, or the end-of-file is encountered.

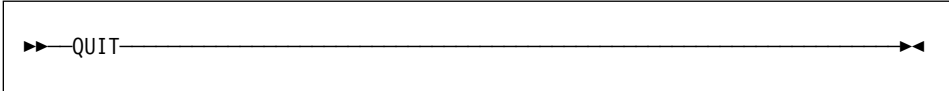
#### Authorization

None

#### Required Connection

None

#### Command Syntax



A diagram showing the command syntax for QUIT. It consists of a horizontal line with a right-pointing arrowhead on the left and a left-pointing arrowhead on the right. The word "QUIT" is positioned on the line, centered between the two arrowheads.

#### Command Parameters

None

#### Usage Notes

QUIT does not terminate the command line processor back-end process or break a database connection. CONNECT RESET breaks a connection, but does not terminate the back-end process. "TERMINATE" on page 368 does both.

---

## REBIND

Allows the user to recreate a package stored in the database without the need for a bind file.

### Authorization

One of the following:

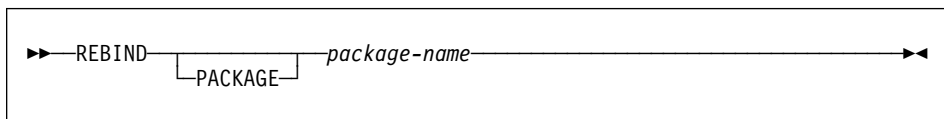
- *sysadm* or *dbadm* authority
- ALTERIN privilege on the schema
- BIND privilege on the package.

The authorization ID logged in the BOUNDBY column of the SYSCAT.PACKAGES system catalog table, which is the ID of the most recent binder of the package, is used as the binder authorization ID for the rebind, and for the default *schema* for table references in the package. Note that this default qualifier may be different from the authorization ID of the user executing the rebind request. REBIND will use the same bind options that were specified when the package was created.

### Required Connection

Database. If no database connection exists, and if implicit connect is enabled, a connection to the default database is made.

### Command Syntax



### Command Parameters

**PACKAGE** *package-name*

The qualified or unqualified name that designates the package to be rebound. An unqualified package name is implicitly qualified by the current authorization ID.

### Usage Notes

REBIND does not automatically commit the transaction following a successful rebind. The user must explicitly commit the transaction. This enables "what if" analysis, in which the user updates certain statistics, and then tries to rebind the package to see what changes. It also permits multiple rebinds within a unit of work.

**Note:** The REBIND command *will* commit the transaction if auto-commit is enabled.

This command:

- Provides a quick way to recreate a package. This enables the user to take advantage of a change in the system without a need for the original bind file. For

## REBIND

example, if it is likely that a particular SQL statement can take advantage of a newly created index, the REBIND command can be used to recreate the package. REBIND can also be used to recreate packages after “RUNSTATS” on page 350 has been executed, thereby taking advantage of the new statistics.

- Provides a method to recreate inoperative packages. Inoperative packages must be explicitly rebound by invoking either the bind utility or the rebind utility. A package will be marked inoperative (the VALID column of the SYSCAT.PACKAGES system catalog will be set to X) if a function instance on which the package depends is dropped.
- Gives users control over the rebinding of invalid packages. Invalid packages will be automatically (or implicitly) rebound by the database manager when they are executed. This may result in a noticeable delay in the execution of the first SQL request for the invalid package. It may be desirable to explicitly rebind invalid packages, rather than allow the system to automatically rebind them, in order to eliminate the initial delay and to prevent unexpected SQL error messages which may be returned in case the implicit rebind fails. For example, following migration, all packages stored in the database will be invalidated by the DB2 Version 2.1 migration process. Given that this may involve a large number of packages, it may be desirable to explicitly rebind all of the invalid packages at one time. This explicit rebinding can be accomplished using BIND, REBIND, or the **db2rbind** tool (see “db2rbind - Rebind all Packages” on page 48).

The choice of whether to use BIND or REBIND to explicitly rebind a package depends on the circumstances. It is recommended that REBIND be used whenever the situation does not specifically require the use of BIND, since the performance of REBIND is significantly better than that of BIND. BIND *must* be used, however:

- When there have been modifications to the program (for example, when SQL statements have been added or deleted, or when the package does not match the executable for the program).
- When the user wishes to modify any of the bind options as part of the rebind. REBIND does not support any bind options. For example, if the user wishes to have privileges on the package granted as part of the bind process, BIND must be used, since it has a **grant** option.
- When the package does not currently exist in the database.
- When detection of *all* bind errors is desired. REBIND only returns the first error it detects, whereas the BIND command returns the first 100 errors that occur during binding.

REBIND is supported by DDCS.

If REBIND is executed on a package that is in use by another user, the rebind will not occur until the other user's logical unit of work ends, because an exclusive lock is held on the package's record in the SYSCAT.PACKAGES system catalog table during the rebind.

When REBIND is executed, the database manager recreates the package from the SQL statements stored in the SYSCAT.STATEMENTS system catalog table.



## REBIND

If REBIND encounters an error, processing stops, and an error message is returned.

REBIND will re-explain packages that were created with the **explsnap** bind option set to YES or ALL (indicated in the EXPLAIN\_SNAPSHOT column in the SYSCAT.PACKAGES catalog table entry for the package) or with the **explain** bind option set to YES or ALL (indicated in the EXPLAIN\_MODE column in the SYSCAT.PACKAGES catalog table entry for the package). The Explain tables used are those of the REBIND requester, not the original binder.

### See Also

“BIND” on page 98

“db2rbind - Rebind all Packages” on page 48

“RUNSTATS” on page 350.

# REDISTRIBUTE NODEGROUP

---

## REDISTRIBUTE NODEGROUP

Redistributes data across the nodes in a nodegroup. The current data distribution, whether it is uniform or skewed, can be specified. The redistribution algorithm selects the partitions to be moved based on the current data distribution.

This command can only be issued from the catalog node. Use “LIST DATABASE DIRECTORY” on page 231 to determine which node is the catalog node for each database.

### Scope

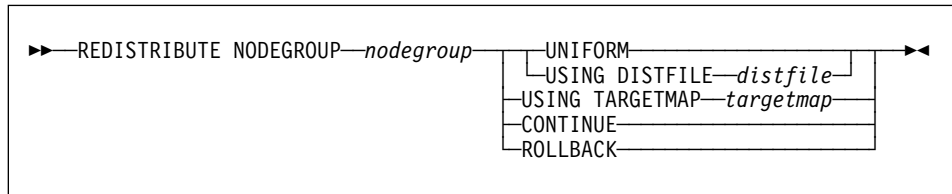
This command affects all nodes in the nodegroup.

### Authorization

One of the following:

*sysadm*  
*sysctrl*  
*sysmaint*  
*dbadm*

### Command Syntax



### Command Parameters

#### **NODEGROUP** *nodegroup*

The name of the nodegroup. This one-part name identifies a nodegroup described in the SYSNODEGROUPS catalog table. The nodegroup cannot currently be undergoing redistribution.

**Note:** Tables in the IBMCATGROUP nodegroup cannot be redistributed.

#### **UNIFORM**

Specifies that the data is uniformly distributed across hash partitions (that is, every hash partition is assumed to have the same number of rows), but the same number of hash partitions do not map to each node. After redistribution, all nodes in the nodegroup have approximately the same number of hash partitions.

#### **USING DISTFILE** *distfile*

If the distribution of partitioning key values is skewed, use this option to achieve a uniform redistribution of data across the nodes of a nodegroup.

## REDISTRIBUTE NODEGROUP

Use the *distfile* to indicate the current distribution of data across the 4096 hash partitions.

Use row counts, byte volumes, or any other measure to indicate the amount of data represented by each hash partition. The utility reads the integer value associated with a partition as the weight of that partition. When a *distfile* is specified, the utility generates a target partitioning map that it uses to redistribute the data across the nodes in the nodegroup as uniformly as possible. After the redistribution, the weight of each node in the nodegroup is approximately the same (the weight of a node is the sum of the weights of all partitions that map to that node).

For example, the input distribution file may contain entries as follows:

```
10223
1345
112000
0
100
...
```

In the example, hash partition 2 has a weight of 112000, and partition 3 (with a weight of 0) has no data mapping to it at all.

The *distfile* should contain 4096 positive integer values in character format. The sum of the values should be less than or equal to 4294967295.

If the path for *distfile* is not specified, the current directory is used.

### USING TARGETMAP *targetmap*

The file specified in *targetmap* is used as the target partitioning map. Data redistribution is done according to this file. If the path is not specified, the current directory is used.

If a node included in the target map is not in the nodegroup, an error is returned. Issue ALTER NODEGROUP ADD NODE before running REDISTRIBUTE NODEGROUP.

If a node excluded from the target map *is* in the nodegroup, that node will not be included in the partitioning. Such a node can be dropped using ALTER NODEGROUP DROP NODE either before or after REDISTRIBUTE NODEGROUP.

### CONTINUE

Continues a previously failed REDISTRIBUTE NODEGROUP operation. If none occurred, an error is returned.

### ROLLBACK

Rolls back a previously failed REDISTRIBUTE NODEGROUP operation. If none occurred, an error is returned.

## REDISTRIBUTE NODEGROUP

### Usage Notes

When a redistribution operation is done, a message file is written to the \$HOME/sqllib/redist directory. The file name has the following format:

*database-name.nodegroup-name.timestamp*

The time stamp value is the time when the command was issued.

This utility performs intermittent COMMITs during processing.

Use the ALTER NODEGROUP statement to add nodes to a nodegroup. This statement permits one to define the containers for the table spaces associated with the nodegroup. See the *SQL Reference* for details.

**Note:** DB2 Parallel Edition for AIX Version 1 syntax, with ADD NODE and DROP NODE options, is supported for users with *sysadm* or *sysctrl* authority. For ADD NODE, containers are created like the containers on the lowest node number of the existing nodes within the nodegroup.

All packages having a dependency on a table that has undergone redistribution are invalidated. It is recommended to explicitly rebind such packages after the redistribute nodegroup operation has completed. Explicit rebinding eliminates the initial delay in the execution of the first SQL request for the invalid package. The redistribute message file contains a list of all the tables that have undergone redistribution.

It is also recommended to update statistics by issuing "RUNSTATS" on page 350 after the redistribute nodegroup operation has completed.

### See Also

"REBIND" on page 309.

---

**REGISTER**

Registers the DB2 server on the network server. Adds the DB2 server's network address in a specified location on the network server.

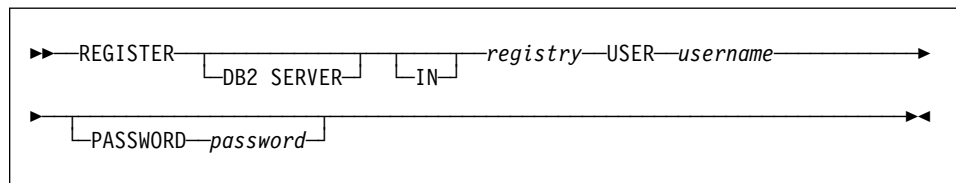
This command is not available on the Windows operating system.

**Authorization**

None

**Required Connection**

None

**Command Syntax****Command Parameters**

**IN** *registry*

Indicates where on the network server to register the DB2 server. In this release, the only supported value is NWBINDERY (NetWare bindery).

**USER** *username*

User ID to log into the network server. The user ID must have SUPERVISOR or Workgroup Manager security equivalence.

**PASSWORD** *password*

Password used to log into the network server. The password must have SUPERVISOR or Workgroup Manager security equivalence.

**Usage Notes**

This command *must* be issued locally from a DB2 server. It is not supported remotely.

This command is only relevant when using file server addressing to connect a client and server.

Install DB2 and configure the database manager IPX/SPX parameters for each server instance. Then issue the REGISTER command at each DB2 server instance once, before invoking "START DATABASE MANAGER" on page 361 for the first time at each DB2 server instance. After that, if the IPX/SPX fields are reconfigured, or the server network address changes, deregister the DB2 server on the network server before making the changes, and then register it again after the changes have been made.

## REGISTER

### See Also

“DEREGISTER” on page 151.

---

## REORGANIZE TABLE

Reorganizes a table by reconstructing the rows to eliminate fragmented data, and by compacting information.

### Scope

This command affects all nodes in the nodegroup.

### Authorization

One of the following:

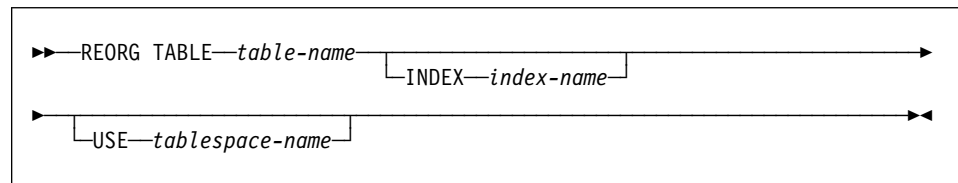
- sysadm*
- sysctrl*
- sysmaint*
- dbadm*

CONTROL privilege on the table.

### Required Connection

Database

### Command Syntax



### Command Parameters

#### **TABLE** *table-name*

Specifies the table to reorganize. The table can be in a local or a remote database. The fully qualified name or alias in the form: *schema.table-name* must be used. The *schema* is the user name under which the table was created.

#### **INDEX** *index-name*

Specifies the index to use when reorganizing the table. The fully qualified name in the form: *schema.index-name* must be used. The *schema* is the user name under which the index was created. The database manager uses the index to physically reorder the records in the table it is reorganizing. If the name of an index is not provided, the records are reorganized without regard to order.

#### **USE** *tablespace-name*

Specifies the name of a temporary table space where the database manager can temporarily store the table being reconstructed. If a table space name is not entered, the database manager stores a working copy

## REORGANIZE TABLE

of the table in the table space(s) in which the table being reorganized resides.

### Example

To reorganize the EMPLOYEE table using the temporary table space TEMPSPACE1 as a work area, enter:

```
db2 reorg table homer.employee using tempSPACE1
```

### Usage Notes

Tables that have been modified so many times that data is fragmented and access performance is noticeably slow are candidates for reorganization. Use “REORGCHK” on page 320 to determine whether a table needs reorganizing. Be sure to complete all database operations and release all locks before invoking REORGANIZE TABLE. This may be done by issuing a COMMIT after closing all cursors opened WITH HOLD, or by issuing a ROLLBACK. After reorganizing a table, use “RUNSTATS” on page 350 to update the table statistics, and “REBIND” on page 309 to rebind the packages that use this table.

If the table is partitioned onto several nodes, and the table reorganization fails on any of the affected nodes, only the failing nodes will have the table reorganization rolled back.

**Note:** If the reorganization is not successful, temporary files should not be deleted. The database manager uses these files to recover the database.

If the name of an index is specified, the database manager reorganizes the data according to the order in the index. To maximize performance, specify an index that is often used in SQL queries.

REORGANIZE TABLE cannot be used on views.

REORGANIZE TABLE cannot be used on a DMS table while an online backup of a table space in which the table resides is being performed.

To complete a table space roll-forward recovery following a table reorganization, both data and LONG table spaces must be roll-forward enabled.

If the table contains LOB columns that do not use the COMPACT option, the LOB DATA storage object can be significantly larger following table reorganization. This can be a result of the order in which the rows were reorganized, and the types of table spaces used (SMS/DMS).

DB2 Version 2 servers do not support down-level client requests to reorganize a table. Since pre-Version 2 servers do not support table spaces, the *tablespace-name* parameter is treated as the Version 1 *path* parameter, when Version 2 clients are used with a down-level server.

If a Version 2 client requests to reorganize a table on a Version 2 server, and that request includes a path instead of a temporary table space in the *tablespace-name*



## REORGANIZE TABLE

parameter (for example, an old application, specifying a temporary file path, being executed on Version 2 clients), REORG chooses a temporary table space in which to place the work files on behalf of the user. A valid temporary table space name containing a path separator character (/ or \) should not be specified, because it will be interpreted as a temporary path (pre-Version 2 request), and REORG will choose a temporary table space on behalf of the user.

### See Also

“REBIND” on page 309

“REORGCHK” on page 320

“RUNSTATS” on page 350.

# REORGCHK

---

## REORGCHK

Calculates statistics on the database to determine if tables need to be reorganized.

### Scope

This command can be issued from any node in the `db2nodes.cfg` file. It can be used to update table and index statistics in the catalogs.

### Authorization

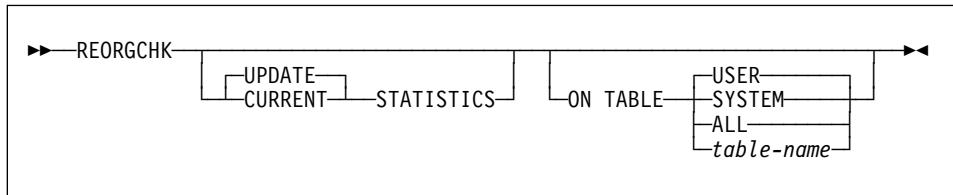
One of the following:

- *sysadm* or *dbadm* authority
- CONTROL privilege on the table.

### Required Connection

Database

### Command Syntax



### Command Parameters

#### UPDATE STATISTICS

Calls the RUNSTATS routine to update table statistics, and then uses the updated statistics to determine if table reorganization is required.

If a table partition exists on the node where REORGCHK has been issued, RUNSTATS executes on this node. If a table partition does not exist on this node, the request is sent to the first node in the nodegroup that holds a partition for the table. RUNSTATS then executes on that node.

#### CURRENT STATISTICS

Uses the current table statistics to determine if table reorganization is required.

#### ON TABLE

*USER*

Checks the tables that are owned by the run time authorization ID.

*SYSTEM*

Checks the system tables.

*ALL*

Checks all user and system tables.

*table-name*

Specifies the table to check. The fully qualified name or alias in the form: *schema.table-name* must be used. The *schema* is the user name under which the table was created. If the table specified is a system catalog table, the *schema* is SYSIBM.

**Example**

The following shows sample output from the command

```
db2 reorgchk update statistics on table system
```

run against the SAMPLE database:

```
Doing RUNSTATS ....
```

Table statistics:

```
F1: 100*OVERFLOW/CARD < 5
F2: 100*TSIZE / ((FPAGES-1) * 4020) > 70
F3: 100*NPAGES/FPAGES > 80
```

CREATOR	NAME	CARD	OV	NP	FP	TSIZE	F1	F2	F3	REORG
SYSIBM	SYSCHECKS	-	-	-	-	-	-	-	-	----
SYSIBM	SYSCOLAUTH	-	-	-	-	-	-	-	-	----
SYSIBM	SYSCOLCHECKS	-	-	-	-	-	-	-	-	----
SYSIBM	SYSCOLDIST	-	-	-	-	-	-	-	-	----
SYSIBM	SYSCOLUMNS	735	0	25	25	92610	0	95	100	---
SYSIBM	SYSCONSTDEP	-	-	-	-	-	-	-	-	----
SYSIBM	SYSDATATYPES	13	0	1	1	1027	0	-	100	---
SYSIBM	SYSDBAUTH	3	0	1	1	90	0	-	100	---
SYSIBM	SYSEVENTMONITORS	-	-	-	-	-	-	-	-	----
SYSIBM	SYSEVENTS	-	-	-	-	-	-	-	-	----
SYSIBM	SYSFUNCPARMS	254	0	6	6	21590	0	100	100	---
SYSIBM	SYSFUNCTIONS	104	0	8	8	728	0	2	100	-*-
SYSIBM	SYSINDEXAUTH	2	0	1	1	112	0	-	100	---
SYSIBM	SYSINDEXES	57	17	3	5	9063	29	56	60	***
SYSIBM	SYSKEYCOLUSE	4	0	1	1	268	0	-	100	---
SYSIBM	SYSPLAN	22	0	2	2	154	0	3	100	-*-
SYSIBM	SYSPLANAUTH	41	0	1	1	1804	0	-	100	---
SYSIBM	SYSPLANDEP	-	-	-	-	-	-	-	-	----
SYSIBM	SYSRELS	-	-	-	-	-	-	-	-	----
SYSIBM	SYSSECTION	4	0	1	1	260	0	-	100	---
SYSIBM	SYSSTMT	4	0	1	1	268	0	-	100	---
SYSIBM	SYSTABAUTH	68	0	2	2	3944	0	98	100	---
SYSIBM	SYSTABCONST	2	0	1	1	132	0	-	100	---
SYSIBM	SYSTABLES	69	0	6	6	483	0	2	100	-*-
SYSIBM	SYSTABLESPACES	3	0	1	1	225	0	-	100	---
SYSIBM	SYSTRIGDEP	-	-	-	-	-	-	-	-	----
SYSIBM	SYSTRIGGERS	-	-	-	-	-	-	-	-	----
SYSIBM	SYSVIEWDEP	42	0	1	1	2646	0	-	100	---
SYSIBM	SYSVIEWS	32	0	5	5	3168	0	19	100	-*-

Index statistics:

# REORGCHK

F4: CLUSTERRATIO or normalized CLUSTERFACTOR > 80  
 F5: 100\*(KEYS\*(ISIZE+10)+(CARD-KEYS)\*4) / (NLEAF\*4096) > 50  
 F6: 90\*(4000/(ISIZE+10)\*\*(NLEVELS-2))\*4096/ (KEYS\*(ISIZE+10)+(CARD-KEYS)\*4)<100

CREATOR	NAME	CARD	LEAF	LVLS	ISIZE	KEYS	F4	F5	F6	REORG
-----										
Table:	SYSIBM.SYSCHECKS									
SYSIBM	IBM37	-	-	-	-	-	-	-	-	----
Table:	SYSIBM.SYSCOLAUTH									
SYSIBM	IBM42	-	-	-	-	-	-	-	-	----
SYSIBM	IBM43	-	-	-	-	-	-	-	-	----
Table:	SYSIBM.SYSCOLCHECKS									
SYSIBM	IBM38	-	-	-	-	-	-	-	-	----
SYSIBM	IBM39	-	-	-	-	-	-	-	-	----
Table:	SYSIBM.SYSCOLDIST									
SYSIBM	IBM46	-	-	-	-	-	-	-	-	----
Table:	SYSIBM.SYSCOLUMNS									
SYSIBM	IBM01	735	12	2	33	735	97	64	11	---
SYSIBM	IBM24	735	1	1	20	10	85	-	-	----
Table:	SYSIBM.SYSCONSTDEP									
SYSIBM	IBM44	-	-	-	-	-	-	-	-	----
SYSIBM	IBM45	-	-	-	-	-	-	-	-	----
Table:	SYSIBM.SYSDATATYPES									
SYSIBM	IBM40	13	1	1	20	13	100	-	-	----
SYSIBM	IBM41	13	1	1	2	13	100	-	-	----
Table:	SYSIBM.SYSDBAUTH									
SYSIBM	IBM12	3	1	1	17	3	100	-	-	----
Table:	SYSIBM.SYSEVENTMONITORS									
SYSIBM	IBM47	-	-	-	-	-	-	-	-	----
Table:	SYSIBM.SYSEVENTS									
SYSIBM	IBM48	-	-	-	-	-	-	-	-	----
Table:	SYSIBM.SYSFUNCPARMS									
SYSIBM	IBM31	254	2	2	30	104	100	58	77	---
SYSIBM	IBM32	254	3	2	51	154	96	79	37	---
SYSIBM	IBM33	254	1	1	6	1	100	-	-	----
Table:	SYSIBM.SYSFUNCTIONS									
SYSIBM	IBM25	104	1	1	30	104	100	-	-	----
SYSIBM	IBM26	104	1	1	27	104	86	-	-	----
SYSIBM	IBM27	104	1	1	18	50	86	-	-	----
SYSIBM	IBM28	104	1	1	16	2	99	-	-	----
SYSIBM	IBM29	104	1	1	4	104	100	-	-	----
SYSIBM	IBM30	104	2	2	53	104	86	79	56	----
Table:	SYSIBM.SYSINDEXAUTH									
SYSIBM	IBM17	2	1	1	47	2	100	-	-	----
SYSIBM	IBM18	2	1	1	30	2	100	-	-	----
Table:	SYSIBM.SYSINDEXES									
SYSIBM	IBM02	57	1	1	17	57	100	-	-	----
SYSIBM	IBM03	57	1	1	25	57	100	-	-	----
Table:	SYSIBM.SYSKEYCOLUSE									
SYSIBM	IBM35	4	1	1	57	4	100	-	-	----
SYSIBM	IBM36	4	1	1	44	2	100	-	-	----
Table:	SYSIBM.SYSPLAN									
SYSIBM	IBM07	22	1	1	16	22	100	-	-	----
SYSIBM	IBM19	22	1	1	8	1	100	-	-	----
Table:	SYSIBM.SYSPLANAUTH									
SYSIBM	IBM13	41	1	1	33	41	100	-	-	----
SYSIBM	IBM14	41	1	1	16	22	100	-	-	----
Table:	SYSIBM.SYSPLANDEP									
SYSIBM	IBM08	-	-	-	-	-	-	-	-	----

## REORGCHK

SYSIBM	IBM09	-	-	-	-	-	-	-	-	----
Table: SYSIBM.SYSRELS										
SYSIBM	IBM20	-	-	-	-	-	-	-	-	----
Table: SYSIBM.SYSSECTION										
SYSIBM	IBM10	4	1	1	20	4	100	-	-	----
Table: SYSIBM.SYSSTMT										
SYSIBM	IBM11	4	1	1	20	4	100	-	-	----
Table: SYSIBM.SYSTABAUTH										
SYSIBM	IBM15	68	1	1	38	68	100	-	-	----
SYSIBM	IBM16	68	1	1	21	68	100	-	-	----
Table: SYSIBM.SYSTABCONST										
SYSIBM	IBM34	2	1	1	44	2	100	-	-	----
Table: SYSIBM.SYSTABLES										
SYSIBM	IBM00	69	1	1	21	69	95	-	-	----
SYSIBM	IBM21	69	1	1	12	3	100	-	-	----
SYSIBM	IBM22	69	1	1	6	1	100	-	-	----
SYSIBM	IBM23	69	1	1	6	1	100	-	-	----
Table: SYSIBM.SYSTABLESPACES										
SYSIBM	IBM49	3	1	1	14	3	100	-	-	----
SYSIBM	IBM50	3	1	1	8	1	100	-	-	----
Table: SYSIBM.SYSTRIGDEP										
SYSIBM	IBM51	-	-	-	-	-	-	-	-	----
SYSIBM	IBM52	-	-	-	-	-	-	-	-	----
Table: SYSIBM.SYSTRIGGERS										
SYSIBM	IBM53	-	-	-	-	-	-	-	-	----
SYSIBM	IBM54	-	-	-	-	-	-	-	-	----
Table: SYSIBM.SYSVIEWDEP										
SYSIBM	IBM05	42	1	1	42	42	100	-	-	----
SYSIBM	IBM06	42	1	1	20	32	100	-	-	----
Table: SYSIBM.SYSVIEWS										
SYSIBM	IBM04	32	1	1	20	32	100	-	-	----

CLUSTERRATIO or normalized CLUSTERFACTOR (F4) will indicate REORG is necessary for indexes that are not in the same sequence as the base table. When multiple indexes are defined on a table, one or more indexes may be flagged as needing REORG. Specify the most important index for REORG sequencing.

The report headings for the table statistics (formulas 1-3) mean:

- CARD**      Number of rows in base table.
- OV**        (OVERFLOW) Number of overflow rows.
- NP**        (NPAGES) Number of pages that contain data.
- FP**        (FPAGES) Total number of pages.
- TSIZE**     Table size in bytes. Calculated as the product of the number of rows in the table (CARD) and the average row length. The average row length is computed as the sum of the average column lengths (AVGCOLLEN in SYSCOLUMNS) plus 10 bytes of row overhead. For long fields and LOBs only the approximate length of the descriptor is used. The actual long field or LOB data is not counted in TSIZE.
- F1**        Results of Formula 1.

## REORGCHK

**F2** Results of Formula 2.

**F3** Results of Formula 3.

**REORG** Each hyphen (-) displayed in this column indicates that the calculated results were within the set bounds of the corresponding formula, and each asterisk (\*) indicates that the calculated results exceeded the set bounds of its corresponding formula.

- - or \* on the left side of the column corresponds to F1 (Formula 1)
- - or \* in the middle of the column corresponds to F2 (Formula 2)
- - or \* on the right side of the column corresponds to F3 (Formula 3).

Table reorganization is suggested when the results of the calculations exceed the bounds set by the formula.

For example, --- indicates that, since the formula results of F1, F2, and F3 are within the set bounds of the formula, no table reorganization is suggested. The notation \*-\* indicates that the results of F1 and F3 suggest table reorganization, even though F2 is still within its set bounds. The notation \*-- indicates that F1 is the only formula exceeding its bounds.

The report headings for the index statistics (formulas 4-6) mean:

**CARD** Number of rows in base table.

**LEAF** Total number of index leaves (pages).

**LVLS** (LEVELS) Number of index levels.

**ISIZE** Index size, calculated from the average column length of all columns participating in the index.

**KEYS** (FULLKEYCARD) Number of unique index entries.

**F4** Results of Formula 4.

**F5** Results of Formula 5. The notation +++ indicates that the result exceeds 999, and is invalid. Rerun REORGCHK with the UPDATE STATISTICS option, or issue "RUNSTATS" on page 350, followed by the REORGCHK command.

**F6** Results of Formula 6. The notation +++ indicates that the result exceeds 999, and is invalid. Rerun REORGCHK with the UPDATE STATISTICS option, or issue "RUNSTATS" on page 350, followed by the REORGCHK command.

**REORG** Each hyphen (-) displayed in this column indicates that the calculated results were within the set bounds of the corresponding formula, and each asterisk (\*) indicates that the calculated result exceeded the set bounds of its corresponding formula.

- - or \* on the left side of the column corresponds to F4 (Formula 4)
- - or \* in the middle of the column corresponds to F5 (Formula 5)
- - or \* on the right side of the column corresponds to F6 (Formula 6).

Table reorganization is suggested when the results of the calculations exceed the bounds set by the formula.

## Usage Notes

REORGCHK calculates statistics obtained from six different formulas to determine if performance has deteriorated or can be improved by reorganizing a table.

**Attention:** These statistics should not be used to determine if empty tables (TSIZE=0) need reorganization. If TSIZE=0 and FPAGE>0, the table needs to be reorganized. If TSIZE=0 and FPAGE=0, no reorganization is necessary.

REORGCHK uses the following formulas to analyze the physical location of rows and the size of the table:

- Formula F1:

$$100 * \text{OVERFLOW} / \text{CARD} < 5$$

The total number of overflow rows in the table should be less than 5 percent of the total number of rows. Overflow rows can be created when rows are updated and the new rows contain more bytes than the old ones (VARCHAR fields), or when columns are added to existing tables.

- Formula F2:

$$100 * \text{TSIZE} / ((\text{FPAGES} - 1) * 4020) > 70$$

The table size in bytes (TSIZE) should be more than 70 percent of the total space allocated for the table. (There should be less than 30% free space.) Because the last page allocated is not usually filled, 1 is subtracted from FPAGES.

- Formula F3:

$$100 * \text{NPAGES} / \text{FPAGES} > 80$$

The number of pages that contain no rows at all should be less than 20 percent of the total number of pages. (Pages can become empty after rows are deleted.)

REORGCHK uses the following formulas to analyze the relationship of the indexes to the table data:

- Formula F4:

$$\text{CLUSTERRATIO or normalized CLUSTERFACTOR} > 80\%$$

The clustering ratio of an index should be greater than 80 percent. When multiple indexes are defined on one table, some of these indexes have a low cluster ratio. (The index sequence is not the same as the table sequence.) This cannot be avoided. Be sure to specify the most important index when reorganizing the table. The cluster ratio is usually not optimal for indexes that contain many duplicate keys and many entries.

- Formula F5:

$$100 * (\text{KEYS} * (\text{ISIZE} + 10) + (\text{CARD} - \text{KEYS}) * 4) / (\text{NLEAF} * 4096) > 50$$

## REORGCHK

Less than 50 percent of the space reserved for index entries should be empty (only checked when NLEAF>1).

- Formula F6:

$$90 * (4000 / (ISIZE + 10)) ** (NLEVELS - 2) * 4096 / \\ (KEYS * (ISIZE + 10) + (CARD - KEYS) * 4) < 100$$

The actual number of index entries should be more than 90% of the number of entries NLEVELS-1 can handle (only checked if NLEVELS>1).

**Note:** Running statistics on many tables can take time, especially if the tables are large.

### See Also

“REORGANIZE TABLE” on page 317

“RUNSTATS” on page 350.



---

### RESET ADMIN CONFIGURATION

Resets the parameters in the database manager configuration file that are relevant to the DB2 Administration Server to the system defaults. The values are reset by node type, which is always a server with remote clients. The DB2 Administration Server is a special DB2 instance that enables remote administration of DB2 servers. The following database manager configuration parameters are reset:

- AGENT\_STACK\_SZ
- AUTHENTICATION
- DIAGLEVEL
- DIAGPATH
- DISCOVER
- DISCOVER\_COMM
- FILESERVER
- IPX\_SOCKET
- NNAME
- OBJECTNAME
- QUERY\_HEAP\_SZ
- SYSADM\_GROUP
- SYSCTRL\_GROUP
- SYSMANT\_GROUP
- TPNAME
- TRUST\_ALLCLNTS
- TRUST\_CLNTAUTH

**Note:** It is not recommended that the SVCENAME parameter, set by the installation program, be modified by the user. The administration server service name is set to use the DB2 registered TCP/IP port (523).

For more information about these parameters, see “GET DATABASE MANAGER CONFIGURATION” on page 184.

### Scope

This command resets the database manager configuration file, `$HOME/sqllib/db2system`. It affects all nodes that are listed in the `$HOME/sqllib/db2nodes.cfg` file.

### Authorization

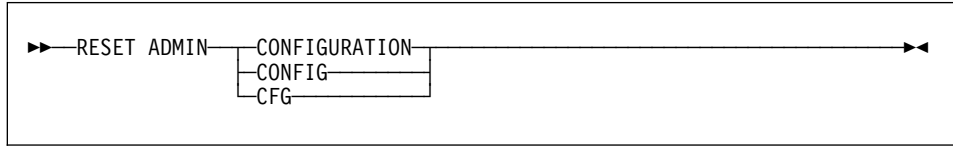
*sysadm*

### Required Connection

None or instance. An instance attachment is not required to perform local DBM configuration operations, but is required to perform remote DBM configuration operations. To reset the database manager configuration for a remote instance, it is necessary to first attach to that instance.

# RESET ADMIN CONFIGURATION

## Command Syntax



## Command Parameters

None

## Usage Notes

To view or print a list of the admin configuration parameters, use “GET ADMIN CONFIGURATION” on page 169.

To change the value of an admin parameter, use “UPDATE ADMIN CONFIGURATION” on page 375.

For more information about these parameters, see the *Administration Guide*.

Changes to the database manager configuration file become effective only after they are loaded into memory. This occurs during execution of **db2start**.

If an error occurs, the database manager configuration file does not change.

The database manager configuration file cannot be reset if the checksum is invalid. This may occur if the database manager configuration file is changed without using the appropriate command. If this happens, the database manager must be installed again to reset the database manager configuration file.

## See Also

“GET ADMIN CONFIGURATION” on page 169

“UPDATE ADMIN CONFIGURATION” on page 375.

# RESET DATABASE CONFIGURATION

---

## RESET DATABASE CONFIGURATION

Resets the configuration of a specific database to the system defaults.

### Scope

This command only affects the node on which it is executed.

### Authorization

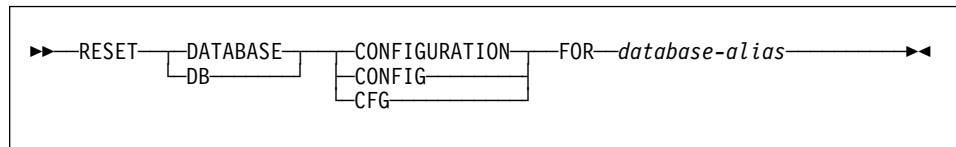
One of the following:

*sysadm*  
*sysctrl*  
*sysmaint*

### Required Connection

Instance. An explicit attachment is not required. If the database is listed as remote, an instance attachment to the remote node is established for the duration of the command.

### Command Syntax



### Command Parameters

**FOR** *database-alias*

Specifies the alias of the database whose configuration is to be reset to the system defaults.

### Usage Notes

To view or print a list of the database configuration parameters, use "GET DATABASE CONFIGURATION" on page 175.

To change the value of a configurable parameter, use "UPDATE DATABASE CONFIGURATION" on page 379.

For more information about these parameters, see the *Administration Guide*.

Changes to the database configuration file become effective only after they are loaded into memory. All applications must disconnect from the database before this can occur.

If an error occurs, the database configuration file does not change.

The database configuration file cannot be reset if the checksum is invalid. This may occur if the database configuration file is changed without using the appropriate

## RESET DATABASE CONFIGURATION

command. If this happens, the database must be restored to reset the database configuration file.

### See Also

“GET DATABASE CONFIGURATION” on page 175

“RESTORE DATABASE” on page 337

“UPDATE DATABASE CONFIGURATION” on page 379.

# RESET DATABASE MANAGER CONFIGURATION

---

## RESET DATABASE MANAGER CONFIGURATION

Resets the parameters in the database manager configuration file to the system defaults. The values are reset by node type, which is always a server with remote clients.

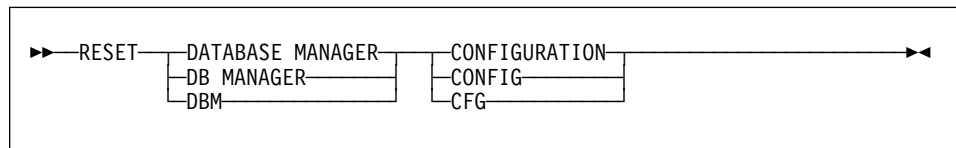
### Authorization

`sysadm`

### Required Connection

None or instance. An instance attachment is not required to perform local DBM configuration operations, but is required to perform remote DBM configuration operations. To reset the database manager configuration for a remote instance, it is necessary to first attach to that instance.

### Command Syntax



### Command Parameters

None

### Usage Notes

To view or print a list of the database manager configuration parameters, use “GET DATABASE MANAGER CONFIGURATION” on page 184.

To change the value of a configurable parameter, use “UPDATE DATABASE MANAGER CONFIGURATION” on page 381.

For more information about these parameters, see the *Administration Guide*.

Changes to the database manager configuration file become effective only after they are loaded into memory. For a server configuration parameter, this occurs during execution of **db2start**. For a client configuration parameter, this occurs when the application is restarted. If the client is the command line processor, it is necessary to invoke “TERMINATE” on page 368.

If an error occurs, the database manager configuration file does not change.

The database manager configuration file cannot be reset if the checksum is invalid. This may occur if the database manager configuration file is changed without using the

## RESET DATABASE MANAGER CONFIGURATION

appropriate command. If this happens, the database manager must be installed again to reset the database manager configuration file.

### See Also

“GET DATABASE MANAGER CONFIGURATION” on page 184

“UPDATE DATABASE MANAGER CONFIGURATION” on page 381.

---

## RESET MONITOR

Resets the internal Database System Monitor data areas of a specified database, or of all active databases, to zero. The internal database system monitor data areas include the data areas for all applications connected to the database, as well as the data areas for the database itself.

### Authorization

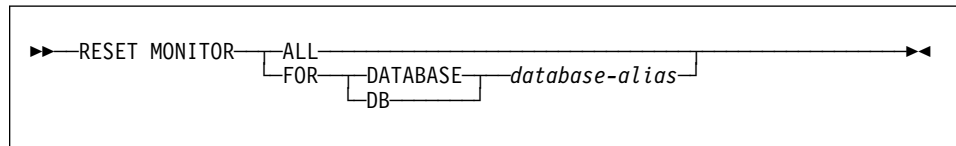
One of the following:

*sysadm*  
*sysctrl*  
*sysmaint*

### Required Connection

Instance. To reset the monitor switches for a remote instance (or a different local instance), it is necessary to first attach to that instance.

### Command Syntax



### Command Parameters

#### ALL

This option indicates that the internal counters should be reset for all databases.

#### FOR DATABASE *database-alias*

This option indicates that only the database with alias *database-alias* should have its internal counters reset.

### Usage Notes

Each process (attachment) has its own private view of the monitor data. If one user resets, or turns off a monitor switch, other users are not affected. Change the setting of the monitor switch configuration parameters to make global changes to the monitor switches (see "UPDATE DATABASE MANAGER CONFIGURATION" on page 381).

If ALL is specified, some database manager information is also reset to maintain consistency of the returned data, and some node-level counters are reset.

To see a list of the data items that can be reset, see the *System Monitor Guide and Reference*.

## RESET MONITOR

### See Also

“GET SNAPSHOT” on page 199

“GET MONITOR SWITCHES” on page 197.



## RESTART DATABASE

Restarts a database that has been abnormally terminated and left in an inconsistent state. At the successful completion of RESTART DATABASE, the application remains connected to the database if the user has CONNECT privilege.

### Scope

This command affects only the node on which it is executed.

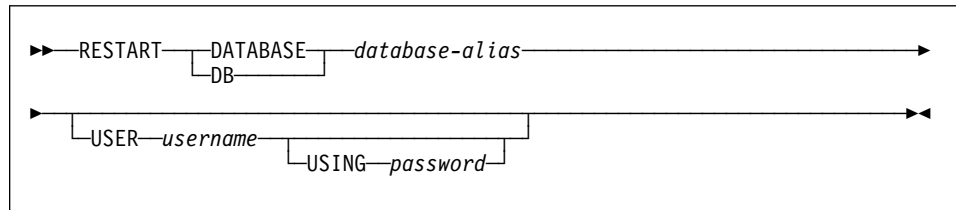
### Authorization

None

### Required Connection

This command establishes a database connection.

### Command Syntax



### Command Parameters

**DATABASE** *database-alias*

Identifies the database to restart.

**USER** *username*

Identifies the user name under which the database is to be restarted.

**USING** *password*

The password used to authenticate *username*. If the password is omitted, the user is prompted to enter it.

### Usage Notes

Execute this command if an attempt to connect to a database returns an error message, indicating that the database must be restarted. This action occurs only if the previous session with this database terminated abnormally (due to power failure, for example).

At the completion of RESTART DATABASE, a shared connection to the database is maintained if the user has CONNECT privilege, and an SQL warning is issued if any indoubt transactions exist. In this case, the database is still usable, but if the indoubt transactions are not resolved before the last connection to the database is dropped, another RESTART DATABASE must be issued before the database can be used again. Use "LIST INDOUBT TRANSACTIONS" on page 241 to generate a list of indoubt

## RESTART DATABASE

transactions. For more information about indoubt transactions, see the *Administration Guide*.

If the database is only restarted on a single node within an MPP system, a message may be returned on a subsequent database query indicating that the database needs to be restarted. This occurs because the database partition on a node on which the query depends must also be restarted. Restarting the database on all nodes solves the problem.

### See Also

CONNECT TO statement in the *SQL Reference*.

---

### RESTORE DATABASE

Rebuilds a damaged or corrupted database that has been backed up using BACKUP DATABASE. The restored database is in the same state it was in when the backup copy was made. This utility can also restore to a database with a name different from the database name in the backup image (in addition to being able to restore to a new database).

The utility can also be used to restore previous versions of DB2 databases.

If, at the time of the backup operation, the database was enabled for roll-forward recovery, the database can be brought to the state it was in prior to the occurrence of the damage or corruption by issuing ROLLFORWARD DATABASE after successful execution of RESTORE DATABASE.

This utility can also restore from a table space level backup.

### Scope

This command only affects the node on which it is executed.

### Authorization

To restore to an existing database, one of the following:

*sysadm*  
*sysctrl*  
*sysmaint*

To restore to a new database, one of the following:

*sysadm*  
*sysctrl*

### Required Connection

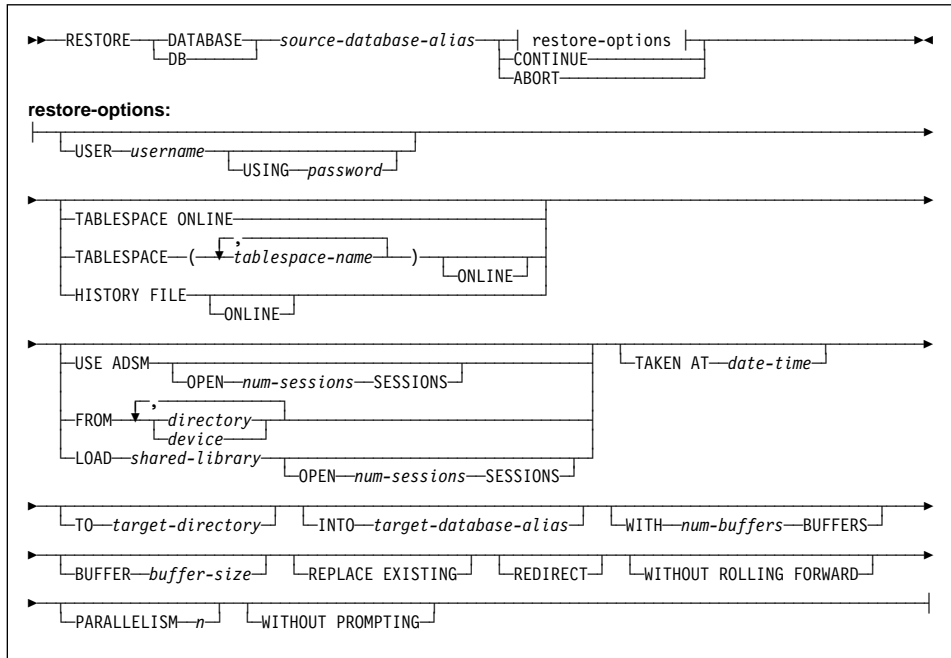
Database, to restore to an existing database.

Instance and database, to restore to a new database. The instance attachment is required to create the database.

To restore to a new remote database, it is necessary to first attach to the instance where the new database will reside.

# RESTORE DATABASE

## Command Syntax



## Command Parameters

**DATABASE** *source-database-alias*

Alias of the source database from which the backup was taken.

**CONTINUE**

Indicates that the containers have been redefined, and that the final step in the redirected restore should be performed.

**ABORT**

Stops the redirected restore. Useful when an error has occurred that would require one or more steps to be repeated. After RESTORE DATABASE with the ABORT option has been issued, each step of a redirected restore must be repeated, including RESTORE DATABASE with the REDIRECT option.

**USER** *username*

Identifies the user name under which the database is to be restored.

**USING** *password*

The password used to authenticate the user name. If the password is omitted, the user is prompted to enter it.

**TABLESPACE** *tablespace-name*

A list of names used to specify the table spaces that are to be restored.

## RESTORE DATABASE

### ONLINE

This keyword, applicable only when doing a table space level restore, is specified to allow the backup to be restored online. This means that other agents can connect while the backup is being restored.

### HISTORY FILE

This keyword is specified to restore the history file from the backup only.

### USE ADSM

Indicates that the database is to be restored from ADSM-managed output.

### OPEN *num-sessions* SESSIONS

The number of I/O sessions to be used with ADSM or the vendor product.

### FROM *directory/device*

The directory or device on which the backup images reside. If USE ADSM, FROM, and LOAD are omitted, the default is the current directory.

If several items are specified, and the last item is a tape device, the user is prompted for another tape. Valid response options are:

- c** Continue. Continue using the device that generated the warning message (for example, when a new tape has been mounted)
- d** Device terminate. Stop using **only** the device that generated the warning message (for example, when there are no more tapes)
- t** Terminate. Abort the restore or backup utility.

Tape is not supported on OS/2. On OS/2, 0 or 0: can be specified to call the user exit program (see the *Administration Guide*). This option is invalid on all other platforms.

**Note:** Redirected restore is not allowed when a user exit program is used to perform the restore.

### LOAD *shared-library*

The name of the shared library (DLL on OS/2 or the Windows operating system) containing the vendor backup and restore I/O functions to be used. It may contain the full path. If the full path is not given, it will default to the path where the user exit programs reside.

### TAKEN AT *date-time*

The time stamp of the database backup. The backup image file name includes the time stamp.

### TO *target-directory*

Directory of the target database. This parameter is ignored if the utility is restoring to an existing database.

### INTO *target-database-alias*

Alias of the target database. If the target database does not exist, it will be created.

### WITH *num-buffers* BUFFERS

The number of buffers to be used.

### BUFFER *buffer-size*

The size, in pages, of the buffer used for the restore. The minimum value for this parameter is 16 pages; the default value is 1024 pages. If a buffer-size of 0 is specified, the value in the database manager configuration parameter *restbufsz* will be used.

## RESTORE DATABASE

The specified value is compared to the value specified during the backup. The actual restore buffer size will be an even multiple of the backup buffer size, which is equal to or greater than the backup buffer size. For example, if a backup buffer size of 1024 pages were specified, and an attempt were made to restore this backup with a buffer size of 16 pages, the actual restore buffer size would be 1024. If the specified restore buffer size were 2049, the actual restore buffer size would be 2048.

### **REPLACE EXISTING**

If a database with the same alias as the target database alias already exists, this parameter tells the restore utility to replace the existing database with the restored database. This is useful in scripts containing the RESTORE DATABASE command, because the CLP will not prompt the user to verify deletion of the existing database. If the WITHOUT PROMPTING parameter is specified, it is not necessary to specify REPLACE EXISTING, but in this case the command will fail if events occur that normally require user intervention.

### **REDIRECT**

Specifies a redirected restore. To complete a redirected restore, this command should be followed by one or more SET TABLESPACE CONTAINERS commands, and then by a RESTORE DATABASE command with the CONTINUE option.

### **WITHOUT ROLLING FORWARD**

Specifies not to place the database in roll-forward pending state after it has been successfully restored.

If, following a successful restore, the database is in roll-forward pending state, ROLLFORWARD DATABASE must be executed before the database can be used.

### **PARALLELISM *n***

Specifies the number of buffer manipulators to be spawned during the restore process. The default value is 1.

### **WITHOUT PROMPTING**

Specifies that the restore will run unattended, and that any actions which normally require user intervention will instead return an error message.

## Examples

Following is a typical redirected restore scenario for a database whose alias is MYDB:

1. Issue a RESTORE DATABASE command with the REDIRECT option.

```
db2 restore db mydb replace existing redirect
```

After successful completion of step 1, and before completing step 3, the restore can be aborted by issuing:

```
db2 restore db mydb abort
```

2. Issue a SET TABLESPACE CONTAINERS command for each table space whose containers must be redefined. For example, on OS/2:

```
db2 set tablespace containers for 5 using  
(file 'f:\ts3con1' 20000, file 'f:\ts3con2' 20000)
```

## RESTORE DATABASE

To verify that the containers of the restored database are the ones specified in this step, issue the LIST TABLESPACE CONTAINERS command.

3. After successful completion of steps 1 and 2, issue:

```
db2 restore db mydb continue
```

This is the final step of the redirected restore.

4. If step 3 fails, or if the restore has been aborted, the redirected restore can be restarted, beginning at step 1.

### Usage Notes

#### Database Level Restore

If restoring to an existing database, the current database configuration file is not replaced by the backup copy unless the configuration file is corrupt.

If WITHOUT ROLLING FORWARD is not specified, and the database was enabled for roll-forward recovery at the time it was backed up, the database is in roll-forward pending state after it has been successfully restored. Use “GET DATABASE CONFIGURATION” on page 175 to check the database state. If the database is in roll-forward pending state, “ROLLFORWARD DATABASE” on page 344 must be issued against the database before it can be used.

BACKUP and RESTORE can also be used to copy a database to another file system or node.

If the backup file being restored was created during an online backup, it is imperative that forward recovery be invoked at the completion of the restore. Forward recovery (using ROLLFORWARD DATABASE) will ensure that any changes which occurred during the course of the backup operation are captured to bring the database into a stable state.

For offline restore, this utility connects to the database in exclusive mode. The utility fails if any application, including the calling application, is already connected to the database that is being restored to.

If an interrupt occurs during a restore, it will not be possible to successfully connect to the database until a successful restore has been performed.

The backup image must be an image that was created by the BACKUP DATABASE command, and may reside on disk, diskette (on OS/2 or the Windows operating system), tape, at the ADSM utility, or on other vendor product-managed media. Tape is not supported on OS/2.

When a database backup is restored to an existing database, the database inherits the alias and database names of the existing database. When restoring to a nonexistent database, the new database will be created with an alias and database name specified by the *target-database-alias* parameter. If a target database alias is not specified, the database will inherit the alias and database name of the backed up database.

## RESTORE DATABASE

Although a remote client may initiate a restore, the source and target always refer to entities that exist at the server.

Restoring databases may have prerequisite requirements and restrictions that are beyond the scope of this manual. For more detailed information about these conditions, see the *Administration Guide*.

### Table Space Level Restore

To ensure that restored table spaces are synchronized with the rest of the database, the table spaces must be rolled forward to the end of the log (or to the point where the table spaces were last used). For this reason, table space level backup and restore can only be performed if roll-forward recovery is enabled. If roll-forward recovery is disabled at any time after a table space level backup is executed, it will not be possible to restore from the backup, and then to roll the table space forward to the current point in time. In this case, all table space level backups taken prior to that time are no longer restorable. The restore operation will fail if the user tries to restore from such a backup. In cases where it cannot be determined that the backup is invalid (if, for instance, the database has been restored and rolled forward, thus creating a new log sequence), the restore may be successful, and the broken restore set will be detected during roll-forward recovery.

Each component of a table may be backed up and restored with the table space in which it resides, independently of the other components of the table.

Table space level backup and restore cannot be run concurrently.

### See Also

“BACKUP DATABASE” on page 93

“GET DATABASE CONFIGURATION” on page 175

“MIGRATE DATABASE” on page 282

“ROLLFORWARD DATABASE” on page 344.



---

## REWIND TAPE

DB2 for Windows NT supports backup and restore to streaming tape devices. Use this command for tape rewinding.

This command is available on Windows NT only.

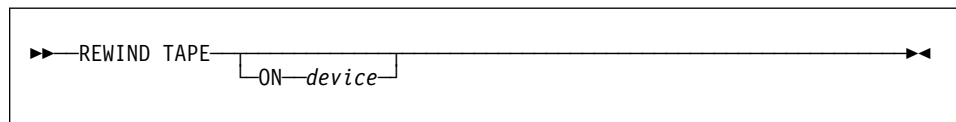
### Authorization

None

### Required Connection

None

### Command Syntax



### Command Parameters

**ON** *device*

Specifies a valid tape device name. The default value is `\\.\TAPE0`.

### See Also

“INITIALIZE TAPE” on page 221  
“SET TAPE POSITION” on page 360.

## ROLLFORWARD DATABASE

---

### ROLLFORWARD DATABASE

Recovers a database by applying transactions recorded in the database log files. Invoked after a database or a table space backup has been restored, or if any table spaces have been taken offline by the database due to a media error. The database must be recoverable (that is, either *logretain*, *userexit*, or both of these database configuration parameters must be set on) before the database can be recovered with roll-forward recovery.

#### Scope

In a multi-node environment, this command can only be issued from the catalog node. A database or table space rollforward command specifying a point-in-time affects all nodes that are listed in the `db2nodes.cfg` file. A database or table space rollforward command specifying end of logs affects the nodes that are specified. If no nodes are specified, it affects all nodes that are listed in the `db2nodes.cfg` file.

#### Authorization

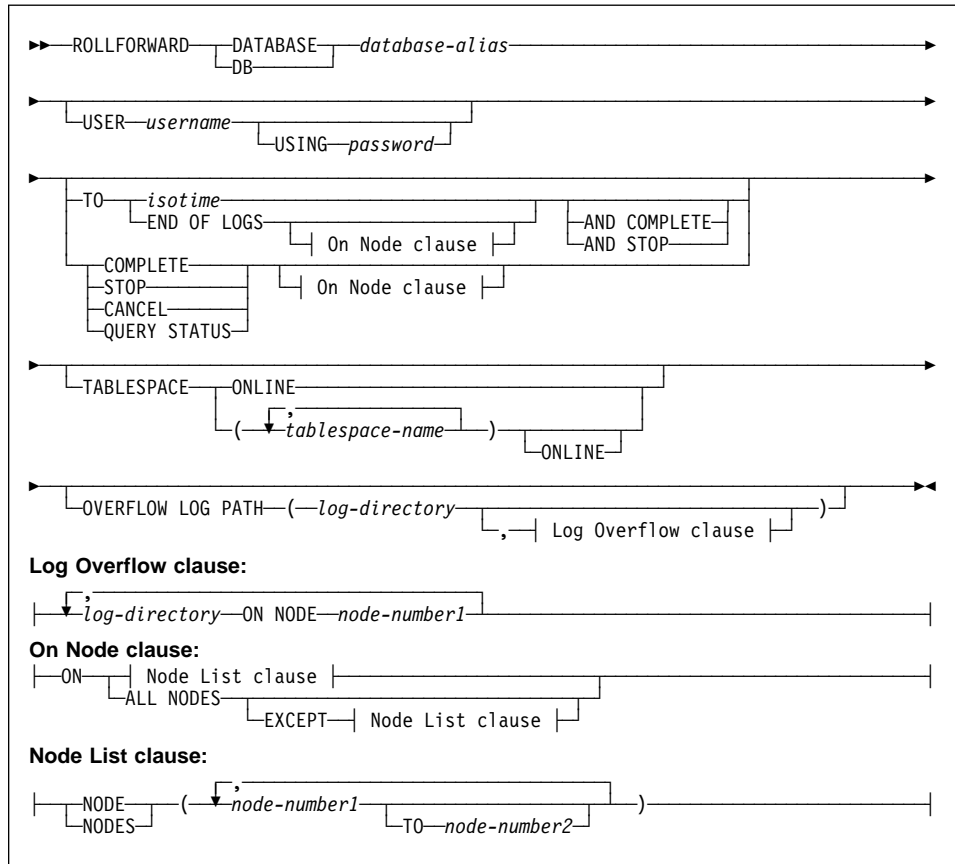
One of the following:

- sysadm*
- sysctrl*
- sysmaint*

#### Required Connection

None. This command establishes a database connection.

## Command Syntax



## Command Parameters

**DATABASE** *database-alias*

The alias of the database to roll forward.

**USER** *username*

Identifies the user name under which the database is to be rolled forward.

**USING** *password*

The password used to authenticate the user name. If the password is omitted, the user is prompted to enter it.

**TO**

*isotime*

The point in time to which all committed transactions are to be rolled forward (including the transaction committed precisely at that time, as well as all transactions committed previously).

This value is specified as a time stamp, a 7-part character string that identifies a combined date and time. The format is

## ROLLFORWARD DATABASE

*yyyy-mm-dd-hh.mm.ss.nnnnnn* (year, month, day, hour, minutes, seconds, microseconds), expressed in Coordinated Universal Time (CUT).

The environment variable **TZ** indicates the difference between CUT and local time. For example, a **TZ** value of EST5EDT indicates that the local time zone is EST; that there is a 5-hour difference between this time zone and CUT; and that daylight savings time is observed. This observance reduces the difference from 5 to 4 hours when daylight savings time is in effect, and CUT = current time + 4.

**Note:** In a multi-node environment, if *isotime* is specified, roll forward recovery is performed on all nodes.

### *END OF LOGS*

Specifies that all committed transactions from all online archive log files listed in the database configuration parameter *logpath* are to be applied.

### **ALL NODES**

Specifies that transactions are to be rolled forward on all nodes specified in the *db2nodes.cfg* file. This is the default if a node clause is not specified.

### **EXCEPT**

Specifies that transactions are to be rolled forward on all nodes specified in the *db2nodes.cfg* file, except those specified in the node list.

### **ON NODE**

### **ON NODES**

Roll forward the database on a set of nodes.

*node-number1*

Specifies a node number in the node list.

*node-number2*

Specifies the second node number, so that all nodes from *node-number1* up to and including *node-number2* are included in the node list.

### **AND COMPLETE**

### **AND STOP**

Completes the roll-forward recovery process by rolling back any incomplete transactions and turning off the roll-forward pending state of the database. This allows access to the database or table spaces that are being rolled forward.

**Note:** When rolling table spaces forward to a point-in-time, the table spaces are placed in backup pending state.

### **COMPLETE STOP**

Does not roll forward any more log records, and completes the roll-forward recovery process by rolling back any incomplete transactions. This allows access to the database or table spaces that are being rolled forward.

**Note:** When rolling table spaces forward to a point-in-time, the table spaces are placed in backup pending state.

### CANCEL

Cancels the roll-forward recovery process. This leaves the database or table space(s) on all nodes on which forward recovery has been started in the restore-pending state. If the database roll-forward is not in progress (that is, the database is in rollforward-pending state), this option will change the database to restore-pending state. If a table space roll-forward is not in progress (that is, the table spaces are in rollforward-pending state), a table space list must be specified. All table spaces in the list will be changed to restore-pending state. If a table space roll-forward is in progress (that is, at least one table space is in rollforward-in-progress state), all table spaces in rollforward-in-progress state will be changed to restore-pending state. If a table space list is specified, it must include all table spaces in rollforward-in-progress state. If rolling forward to a point in time, any table space passed in will be ignored, and all table spaces that are rollforward-in-progress state will be put in restore pending state. If rolling forward to the end of logs with a table space list, only those table spaces will be put in restore-pending state.

**Note:** Use this option with caution.

### QUERY STATUS

Lists the log files that the database manager has rolled forward, the next archive file required, and the time stamp (in CUT) of the last committed transaction since roll-forward processing began. In a multi-node environment, this status information is returned for each node. The information returned contains the following fields:

*Node number*

*Rollforward status*

Status may be database or table space rollforward pending, database or table space rollforward in progress, database or table space rollforward processing STOP, or no rollforward pending.

*Next log file to be read*

A string containing the name of the next required log file. In a multi-node environment, use this information if the rollforward utility fails with a return code indicating that a log file is missing or a log information mismatch has occurred.

*Log files processed*

A string containing the names of the processed log files that are no longer needed for recovery, and that can be removed from the directory.

*Last committed transaction*

A string containing a time stamp in ISO format (yyyy-mm-dd-hh.mm.ss). This time stamp marks the last transaction committed after the completion of roll-forward recovery. The time stamp applies to the database. For table

## ROLLFORWARD DATABASE

space roll-forward, it is the time stamp of the last transaction committed to the database.

**Note:** QUERY STATUS is the default if the TO, STOP, COMPLETE, and CANCEL clauses are omitted.

If TO, STOP, or COMPLETE are specified, this information will be displayed if the command ran successfully.

### TABLESPACE

This keyword is specified for table space level roll-forward.

*tablespace-name*

Mandatory for table space level roll-forward to a point in time. Also allows a subset of table spaces to be specified for a roll-forward to the end of logs. In a multi-node environment, each table space in the list does not have to exist at each node that is rolling forward. If it *does* exist at the node, it must be in the correct state.

### ONLINE

This keyword is specified to allow the table space level roll-forward recovery to be done online. This means that other agents are allowed to connect while roll-forward recovery is in progress.

### OVERFLOW LOG PATH *log-directory*

Specifies an alternate log path to be searched for archived logs during recovery. In a multi-node environment, this is the default overflow log path for all nodes.

In a single-node environment, a relative overflow log path can be specified, but in a multi-node environment, the path must be fully qualified.

*log-directory* **ON NODE**

In a multi-node environment, allows a different log path to override the default overflow log path for a specific node.

## Usage Notes

The database manager uses the information stored in the archived and the active log files to reconstruct the transactions performed on the database since its last backup.

If the database is in roll-forward pending state when ROLLFORWARD DATABASE is invoked, the database will be rolled forward. Table spaces are returned to normal state after a successful database roll-forward, unless an abnormal state causes one or more table spaces to go offline.

If the database is not in roll-forward pending state and no point in time is specified, any table spaces that are in rollforward-in-progress state will be rolled forward to the end of logs. If no table spaces are in rollforward-in-progress state, any table spaces that are in rollforward pending state will be rolled forward to the end of logs.

The roll-forward operation can be performed on a subset of table spaces by specifying table space names.

## ROLLFORWARD DATABASE

If rolling forward table spaces to a point in time, a subset of table spaces must be specified. Only those table spaces specified will be rolled forward. Each table space must be in roll-forward pending state or, if continuing a table space roll-forward that is already in progress, in rollforward-in-progress state.

If enabling an existing database for roll-forward recovery, change the number of primary log files to the sum of the number of primary log files and secondary log files +1. More information will be logged for LONG VARCHAR fields and LOB data in a database enabled for roll-forward recovery.

Rolling databases forward may require a load recovery using tape devices. If prompted for another tape, the user can respond with one of the following:

- c** Continue. Continue using the device that generated the warning message (for example, when a new tape has been mounted)
- d** Device terminate. Stop using the device that generated the warning message (for example, when there are no more tapes)
- t** Terminate. Terminate all devices.

Rolling databases forward may involve prerequisites and restrictions that are beyond the scope of this manual. For more detailed information, see the *Administration Guide*.

### See Also

“LOAD” on page 262

“RESTORE DATABASE” on page 337.

# RUNSTATS

---

## RUNSTATS

Updates statistics about the physical characteristics of a table and the associated indexes. These characteristics include number of records, number of pages, and average record length. The optimizer uses these statistics when determining access paths to the data.

This utility should be called when a table has had many updates, or after reorganizing a table.

### Scope

This command can be issued from any node in the `db2nodes.cfg` file. It can be used to update the catalogs on the catalog node.

The command collects statistics for a table on the node from which it is invoked. If the table does not exist on that node, the first node in the nodegroup is selected.

### Authorization

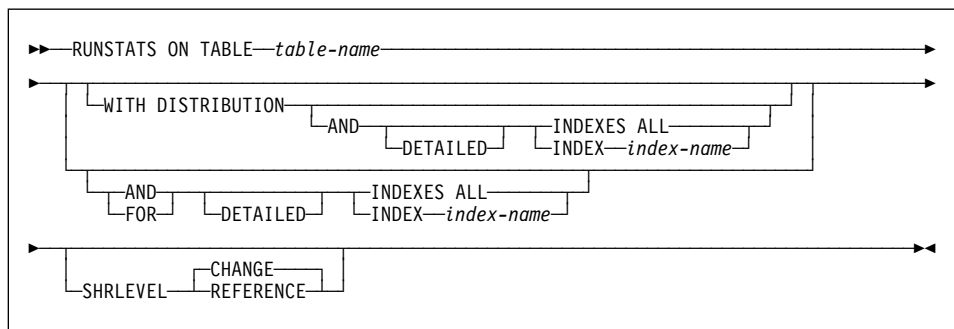
One of the following:

- sysadm*
- sysctrl*
- sysmaint*
- dbadm*
- CONTROL privilege on the table.

### Required Connection

Database

### Command Syntax



### Command Parameters

**TABLE** *table-name*

The table on which to update statistics. The fully qualified name or alias in the form: `schema.table-name` must be used. The `schema` is the user name



under which the table was created. If no options are specified, only table statistics will be updated. Other users will have access to the table while the statistics are being gathered.

## **WITH DISTRIBUTION**

Specifies that distribution statistics are requested. The number of most frequent values collected is defined by the *num\_freqvalues* database configuration parameter. The number of quantiles collected is defined by the *num\_quantiles* database configuration parameter. For information about nonuniform distribution statistics, see the *Administration Guide*.

## **AND INDEXES ALL**

Update statistics on both the table and its indexes.

## **AND INDEX** *index-name*

Update statistics on both the table and the specified index, where *index-name* is a fully qualified name in the form: *schema.index-name*.

## **FOR INDEXES ALL**

Update statistics on the indexes only. If statistics on the table have never been generated, the database manager calculates statistics on the table as well as on the indexes.

## **FOR INDEX** *index-name*

Update statistics on the specified index only. If table statistics have never been generated, the database manager calculates statistics on the table as well as on the index. The index-name is a fully qualified name in the form: *schema.index-name*.

## **DETAILED**

Calculate extended index statistics.

## **SHRLEVEL**

### *CHANGE*

Specifies that other users can read from and write to the table while statistics are calculated.

### *REFERENCE*

Specifies that other users can have read-only access to the table while statistics are calculated.

## **Examples**

Collect statistics on table only, without distribution statistics:

```
db2 runstats on table smith.table1
```

Collect statistics on table only, with distribution statistics:

```
db2 runstats on table smith.table1 with distribution
```

Collect basic statistics on indexes only:

```
db2 runstats on table smith.table1 for indexes all
```

Collect statistics on table and all indexes (basic level):

```
db2 runstats on table smith.table1 and indexes all
```

## RUNSTATS

Collect statistics on table, with distribution statistics and index statistics:

```
db2 runstats on table smith.table1 with distribution and indexes all
```

Collect all possible statistics (distribution and extended index):

```
db2 runstats on table smith.table1 with distribution and detailed index
```

Collect distribution statistics on index INDEX1 only:

```
db2 runstats on table smith.table1 with distribution for index smith.index1
```

### Usage Notes

Use RUNSTATS to update the statistics on tables:

- That have been modified many times (for example, if a large number of updates have been made, or if a significant amount of data has been inserted or deleted)
- That have been reorganized.

After statistics have been updated, new access paths to the table can be created by rebinding the packages using “BIND” on page 98.

Statistics for tables only should be collected before any indexes are created. This will ensure that statistics gathered during index creation are not overlaid by estimates gathered during the calculation of table statistics.

Statistics are collected based on the table partition that is resident on the node where the command executes. Global table statistics are derived by multiplying the values obtained at a node by the number of nodes on which the table is completely stored. The global statistics are stored in the catalog tables.

The node from which the command is issued does not have to contain a partition for the table:

- If the command is issued from a node that contains a partition for the table, the utility executes at this node.
- If the command is issued from a node that does not contain a table partition, the request is sent to the first node in the nodegroup that holds a partition for the table. The utility then executes at this node.

### See Also

“GET DATABASE CONFIGURATION” on page 175

“REORGANIZE TABLE” on page 317

“REORGCHK” on page 320.

**SET CLIENT**

Specifies connection settings for the back-end process.

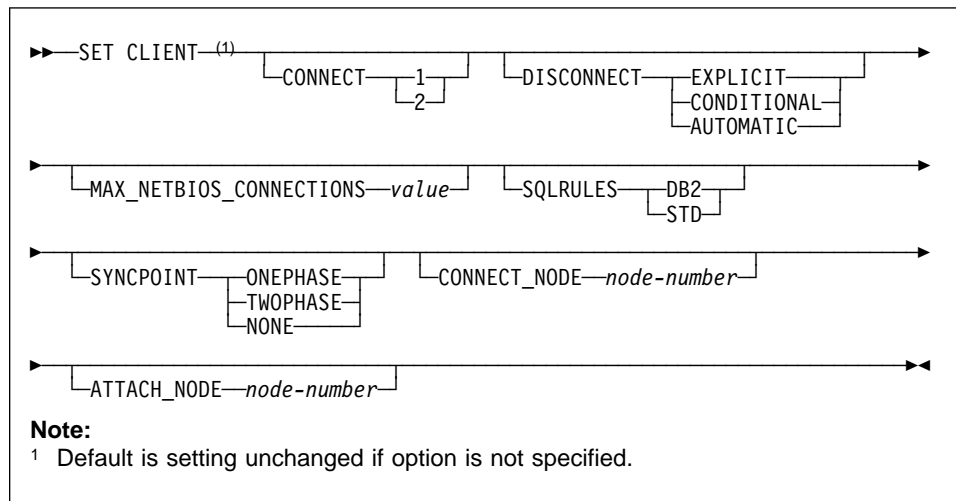
**Authorization**

None

**Required Connection**

None

**Command Syntax**



**Command Parameters**

**CONNECT**

1

Specifies that a CONNECT statement is to be processed as a type 1 CONNECT.

2

Specifies that a CONNECT statement is to be processed as a type 2 CONNECT.

**DISCONNECT**

*EXPLICIT*

Specifies that only database connections that have been explicitly marked for release by the RELEASE statement are to be disconnected at commit.

## SET CLIENT

### *CONDITIONAL*

Specifies that the database connections that have been marked **RELEASE** or have no open **WITH HOLD** cursors are to be disconnected at commit.

### *AUTOMATIC*

Specifies that all database connections are to be disconnected at commit.

### **MAX\_NETBIOS\_CONNECTIONS** *value*

Specifies the maximum number of concurrent connections that can be made in an application using a NetBIOS adapter. Maximum value is 254. This parameter must be set before the first NetBIOS connection is made. Changes subsequent to the first connection are ignored.

### **SQLRULES**

#### *DB2*

Specifies that a type 2 **CONNECT** is to be processed according to the **DB2** rules.

#### *STD*

Specifies that a type 2 **CONNECT** is to be processed according to the Standard (**STD**) rules based on **ISO/ANS SQL92**.

### **SYNCPOINT**

Specifies how commits or rollbacks are to be coordinated among multiple database connections.

#### *ONEPHASE*

Specifies that no Transaction Manager (**TM**) is to be used to perform a two-phase commit. A one-phase commit is to be used to commit the work done by each database in multiple database transactions.

#### *TWOPHASE*

Specifies that the **TM** is required to coordinate two-phase commits among those databases that support this protocol.

#### *NONE*

Specifies that no **TM** is to be used to perform a two-phase commit, and does not enforce single updater, multiple reader. A **COMMIT** is sent to each participating database. The application is responsible for recovery if any of the commits fail.

### **CONNECT\_NODE (MPP only)** *node-number*

Specifies the node to which a connect is to be made. A value between zero and 999, inclusive. Overrides the value of the environment variable **DB2NODE**.

### **ATTACH\_NODE (MPP only)** *node-number*

Specifies the node to which an attach is to be made. A value between zero and 999, inclusive. Overrides the value of the environment variable **DB2NODE**.

For example, if nodes 1, 2, and 3 are defined, the client only needs to be able to access one of these nodes. If only node 1 containing databases has been cataloged, and this parameter is set to 3, then the next attach

attempt will result in an attachment at node 3, after an initial attachment at node 1.

### Examples

To set specific values:

```
db2 set client connect 2 disconnect automatic sqlrules std
syncpoint twophase
```

To change SQLRULES back to DB2, but keep the other settings:

```
db2 set client sqlrules db2
```

**Note:** The connection settings revert to default values after “TERMINATE” on page 368 is issued.

### Usage Notes

SET CLIENT cannot be issued if one or more connections are active.

If SET CLIENT is successful, the connections in the subsequent units of work will use the connection settings specified. If SET CLIENT is unsuccessful, the connection settings of the back-end process are unchanged.

For more information about distributed unit of work (DUOW), see the *Administration Guide*.

### See Also

“QUERY CLIENT” on page 304.

# SET RUNTIME DEGREE

---

## SET RUNTIME DEGREE

Sets the maximum run time degree of parallelism for SQL statements for specified active applications.

### Scope

This command affects all nodes that are listed in the `$HOME/sqllib/db2nodes.cfg` file.

### Authorization

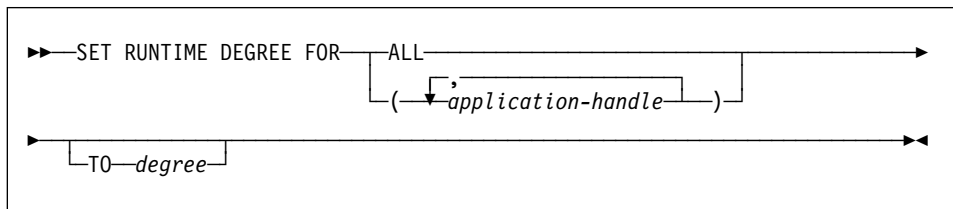
One of the following:

*sysadm*  
*sysctrl*

### Required Connection

Instance. To change the maximum run time degree of parallelism on a remote server, it is first necessary to attach to that server. If no attachment exists, the SET RUNTIME DEGREE command fails.

### Command Syntax



### Command Parameters

#### FOR

*ALL*

The specified degree will apply to all applications.

*application-handle*

Specifies the agent to which the new degree applies. List the values using "LIST APPLICATIONS" on page 225.

#### TO *degree*

The maximum run time degree of parallelism.

### Example

The following example sets the maximum run time degree of parallelism for two users, with *application-handle* values of 41408 and 55458, to 4:

```
db2 SET RUNTIME DEGREE FOR ( 41408, 55458 ) TO 4
```

### Usage Notes

This command provides a mechanism to modify the maximum degree of parallelism for active applications. It can be used to override the value that was determined at SQL statement compilation time.

The run time degree of parallelism specifies the maximum number of parallel operations that will be used when the statement is executed. The degree of parallelism for an SQL statement can be specified at statement compilation time using the CURRENT DEGREE special register or the **degree** bind option. The maximum run time degree of parallelism for an active application can be specified using the SET RUNTIME DEGREE command. The *max\_querydegree* database manager configuration parameter specifies the maximum run time degree for any SQL statement executing on this instance of the database manager.

The actual run time degree will be the lowest of:

- the *max\_querydegree* configuration parameter
- the application run time degree
- the SQL statement compilation degree.

## SET TABLESPACE CONTAINERS

---

### SET TABLESPACE CONTAINERS

A *redirected restore* is a restore in which the set of table space containers for the restored database is different from the set of containers for the original database at the time the backup was done. This command permits the addition, change, or removal of table space containers for a database that is to be restored. If, for example, one or more containers become inaccessible for any reason, the restore fails if it is not redirected to different containers.

**Note:** A redirected restore is not allowed when a user exit program is used to perform the restore.

#### Authorization

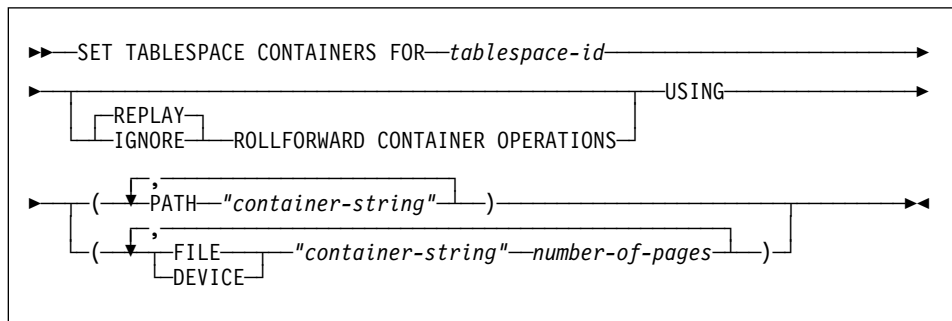
One of the following:

*sysadm*  
*sysctrl*

#### Required Connection

Database

#### Command Syntax



#### Command Parameters

**FOR** *tablespace-id*

An integer that uniquely represents a table space used by the database being restored.

**REPLAY ROLLFORWARD CONTAINER OPERATIONS**

Specifies that any ALTER TABLESPACE operation issued against this table space since the database was backed up is to be redone during a subsequent roll forward of the database.

**IGNORE ROLLFORWARD CONTAINER OPERATIONS**

Specifies that ALTER TABLESPACE operations in the log are to be ignored when performing a roll forward.



## SET TABLESPACE CONTAINERS

### **USING PATH** "*container-string*"

For an SMS table space, identifies one or more containers that will belong to the table space and into which the table space data will be stored. It is an absolute or relative directory name. If the directory name is not absolute, it is relative to the database directory. The string cannot exceed 240 bytes in length.

### **USING FILE/DEVICE** "*container-string*" *number-of-pages*

For a DMS table space, identifies one or more containers that will belong to the table space and into which the table space data will be stored. The container type (either FILE or DEVICE) and its size (in 4KB pages) are specified. A mixture of file and device containers can be specified. The string cannot exceed 254 bytes in length.

For a file container, the string must be an absolute or relative file name. If the file name is not absolute, it is relative to the database directory.

For a device container, the string must be a device name. The device must already exist. Device containers are not supported on OS/2.

### **Example**

See the example in "RESTORE DATABASE" on page 337.

### **Usage Notes**

This command is used in conjunction with "RESTORE DATABASE" on page 337.

A backup of a database, or one or more table spaces, keeps a record of all the table space containers in use by the table spaces being backed up. During a restore, all containers listed in the backup are checked to see if they currently exist and are accessible. If one or more of the containers is inaccessible for any reason, the restore will fail. In order to allow a restore in such a case, the redirecting of table space containers is supported during the restore. This support includes adding, changing, or removing of table space containers. It is this command that allows the user to add, change or remove those containers. For more information, see the *Administration Guide*.

### **See Also**

"BACKUP DATABASE" on page 93

"RESTORE DATABASE" on page 337

"ROLLFORWARD DATABASE" on page 344.

## SET TAPE POSITION

---

### SET TAPE POSITION

DB2 for Windows NT supports backup and restore to streaming tape devices. Use this command for tape positioning.

This command is available on Windows NT only.

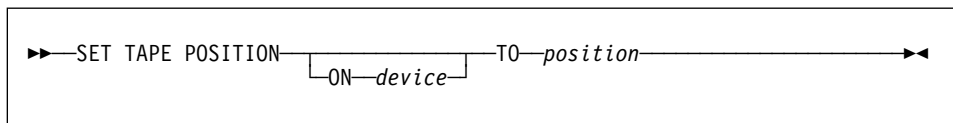
#### Authorization

None

#### Required Connection

None

#### Command Syntax



#### Command Parameters

**ON** *device*

Specifies a valid tape device name. The default value is `\\.\TAPE0`.

**TO** *position*

Specifies the mark at which the tape is to be positioned. DB2 for Windows NT writes a tape mark after every backup. A value of 1 specifies the first position, 2 specifies the second position, and so on. If the tape is positioned at tape mark 1, archive 2 is positioned to be restored.

#### See Also

“INITIALIZE TAPE” on page 221

“REWIND TAPE” on page 343.

## START DATABASE MANAGER

Starts the current database manager instance background processes on a single node or on all the nodes defined in a multi-node environment.

This command is not valid on a client.

### Scope

In a multi-node environment, this command affects all nodes that are listed in the `$HOME/sql1lib/db2nodes.cfg` file, unless the `nodenum` parameter is used.

### Authorization

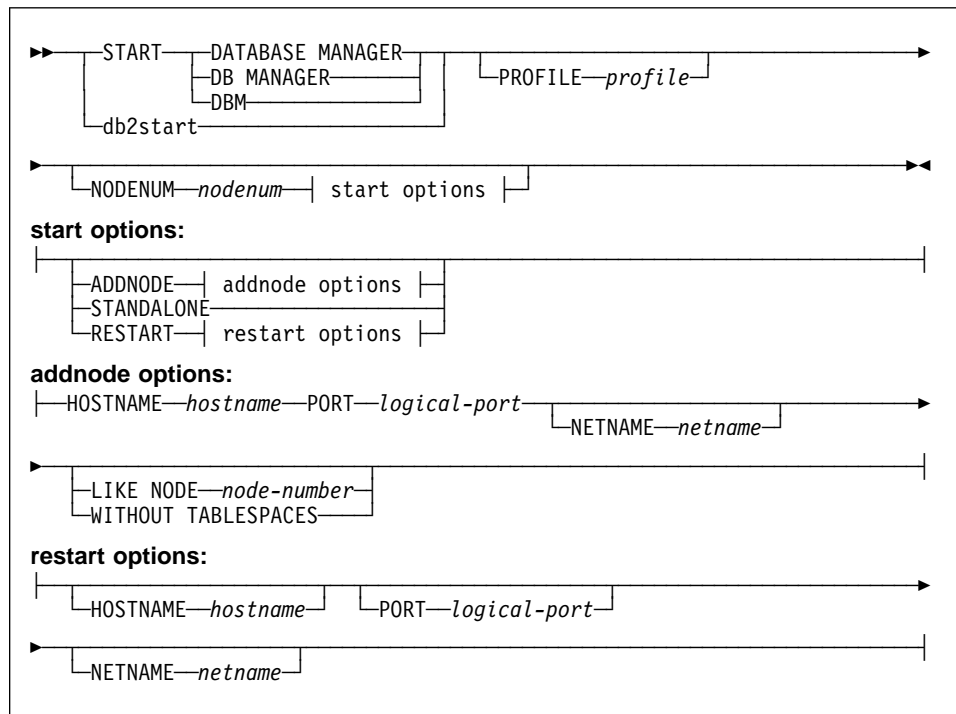
One of the following:

`sysadm`  
`sysctrl`  
`sysmaint`

### Required Connection

None

### Command Syntax



# START DATABASE MANAGER

## Command Parameters

**Note:** All of the following parameters are valid in an MPP environment only.

### **PROFILE** *profile*

Specifies the name of the profile file to be executed at each node to define the DB2 environment. This file is executed before the nodes are started. The profile file must reside in the `sql1lib` directory of the instance owner.

**Note:** The environment variables in the profile file are not necessarily all defined in the user session.

### **NODENUM** *nodenum*

Specifies the node to be started. If no other options are specified, a normal startup is done at this node.

Valid values are from 0 to 999 inclusive. If **ADDNODE** is not specified, the value must already exist in the `db2nodes.cfg` file of the instance owner. If no node number is specified, all nodes defined in the node configuration file are started.

### **ADDNODE**

Specifies that the new node is added to the `db2nodes.cfg` file of the instance owner with the *hostname* and *logical-port* values.

Ensure that the combination of *hostname* and *logical-port* is unique.

The add node utility is executed internally to create all existing databases on the node being added. After a node is added, the `db2nodes.cfg` file is not updated with the new node until a **db2stop** is issued. The node is not part of the MPP system until the next **db2start** following the **db2stop**.

**Note:** When the database partitions are created on the new node, their configuration parameters are set to the default.

### **HOSTNAME** *hostname*

With **ADDNODE**, specifies the host name to be added to the `db2nodes.cfg` file.

### **PORT** *logical-port*

With **ADDNODE**, specifies the logical port to be added to the `db2nodes.cfg` file. Valid values are from 0 to 999.

### **NETNAME** *netname*

Specifies the *netname* to be added to the `db2nodes.cfg` file. If not specified, this parameter defaults to the value specified for *hostname*.

### **LIKE NODE** *node-number*

Specifies that the containers for the temporary table spaces will be the same as the containers on the specified *node-number* for each database in the instance. The node specified must be a node that is already in the `db2nodes.cfg` file.

## START DATABASE MANAGER

### WITHOUT TABLESPACES

Specifies that containers for the temporary table spaces are not created for any of the databases. The ALTER TABLESPACE statement must be used to add temporary table space containers to each database before the database can be used.

### STANDALONE

Specifies that the node is to be started in stand-alone mode. FCM does not attempt to establish a connection to any other node. This option is used when adding a node.

### RESTART

Starts the database manager after a failure. Other nodes are still operating, and this node attempts to connect to the others. If neither the *hostname* nor the *logical-port* parameter is specified, the database manager is restarted using the *hostname* and *logical-port* values specified in *db2nodes.cfg*. If either parameter is specified, the new values are sent to the other nodes when a connection is established. The *db2nodes.cfg* file is updated with this information.

#### HOSTNAME *hostname*

With RESTART, specifies the host name to be used to override that in the node configuration file.

#### PORT *logical-port*

With RESTART, specifies the logical port number to be used to override that in the node configuration file. If not specified, this parameter defaults to the *logical-port* value that corresponds to the *nodenum* value in the *db2nodes.cfg* file. Valid values are from 0 to 999.

#### NETNAME *netname*

Specifies the *netname* to override that specified in the *db2nodes.cfg* file. If not specified, this parameter defaults to the *netname* value that corresponds to the *nodenum* value in the *db2nodes.cfg* file.

## Example

The following is sample output from **db2start** issued on a three node system with nodes 10, 20, and 30:

```
04-07-1997 10:33:05 10 0 SQL1063N DB2START processing was successful.
04-07-1997 10:33:07 20 0 SQL1063N DB2START processing was successful.
04-07-1997 10:33:07 30 0 SQL1063N DB2START processing was successful.
SQL1063N DB2START processing was successful.
```

## Usage Notes

It is not necessary to issue this command on a client node. It is provided for compatibility with older clients, but it has no effect on the database manager.

## START DATABASE MANAGER

Once started, the database manager instance runs until the user stops it, even if all application programs that were using it have ended.

If the database manager starts successfully, a successful completion message is sent to the standard output device. If an error occurs, processing stops, and an error message is sent to the standard output device. In a multi-node environment, messages are returned on the node that issued the START DATABASE MANAGER command.

If no parameters are specified in a multi-node database environment, the database manager is started on all parallel nodes using the parameters specified in the node configuration file.

If a START DATABASE MANAGER command is in progress, ensure that the applicable nodes have started *before* issuing a request to the database.

The db2cshrc file is not supported and cannot be used to define the environment.

On UNIX platforms, the START DATABASE MANAGER command supports the SIGINT and SIGALRM signals. The SIGINT signal is issued if CTRL+C is pressed. The SIGALRM signal is issued if the value specified for the *start\_stop\_time* database manager configuration parameter is reached. If either signal occurs, all in-progress startups are interrupted and a message (SQL1044N for SIGINT and SQL6037N for SIGALRM) is returned from each interrupted node to the `$HOME/sql1lib/log/db2start.timestamp.log` error log file. Nodes that are already started are not affected. If CTRL+C is pressed on a node that is starting, **db2stop** must be issued on that node before an attempt is made to start it again.

### See Also

“ADD NODE” on page 89

“STOP DATABASE MANAGER” on page 365.

STOP DATABASE MANAGER

Stops the current database manager instance. Unless explicitly stopped, the database manager continues to be active. This command does not stop the database manager instance if any applications are connected to databases. If there are no database connections, but there are instance attachments, it forces the instance attachments and stops the database manager. This command also deactivates any outstanding database activations before stopping the database manager.

On an MPP system, this command stops the current database manager instance on a node or on all nodes. When it stops the database manager on all nodes, it uses the node configuration file db2nodes.cfg to obtain information about each node.

This command can also be used to drop a node from the db2nodes.cfg file (MPP systems only).

This command is not valid on a client.

Scope

By default, and in a multi-node environment, this command affects all nodes that are listed in the db2nodes.cfg file.

Authorization

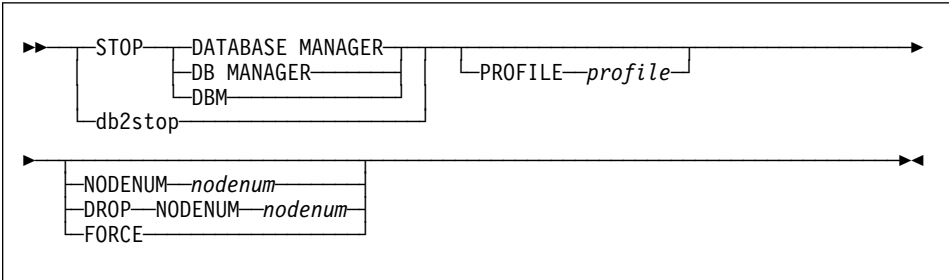
One of the following:

- sysadm
sysctrl
sysmaint

Required Connection

None

Command Syntax



# STOP DATABASE MANAGER

## Command Parameters

### **PROFILE** *profile*

MPP only. Specifies the name of the profile file that was executed at startup to define the DB2 environment for those nodes that were started. If a profile for "START DATABASE MANAGER" on page 361 was specified, the same profile must be specified here. The profile file must reside in the `sql1lib` directory of the instance owner.

### **NODENUM** *nodenum*

MPP only. Specifies the node to be stopped.

Valid values are from 0 to 999 inclusive, and must be in the `db2nodes.cfg` file. If no node number is specified, all nodes defined in the node configuration file are stopped.

### **DROP NODENUM** *nodenum*

MPP only. Specifies the node to be dropped from the `db2nodes.cfg` file.

Before using this parameter, run "DROP NODE VERIFY" on page 159 to ensure that there is no user data on this node.

When this option is specified, all nodes in the `db2nodes.cfg` file are stopped.

### **FORCE**

Specifies to use (ALL) when stopping the database manager at each node.

## Example

The following is sample output from **db2stop** issued on a three node system with nodes 10, 20, and 30:

```
04-07-1997 10:32:53    10  0  SQL1064N  DB2STOP processing was successful.
04-07-1997 10:32:54    20  0  SQL1064N  DB2STOP processing was successful.
04-07-1997 10:32:55    30  0  SQL1064N  DB2STOP processing was successful.
SQL1064N  DB2STOP processing was successful.
```

## Usage Notes

It is not necessary to issue this command on a client node. It is provided for compatibility with older clients, but it has no effect on the database manager.

Once started, the database manager instance runs until the user stops it, even if all application programs that were using it have ended.

If the database manager is stopped, a successful completion message is sent to the standard output device. If an error occurs, processing stops, and an error message is sent to the standard output device.

If the database manager cannot be stopped because application programs are still connected to databases, use "FORCE APPLICATION" on page 167 to disconnect all users first, or reissue the STOP DATABASE MANAGER command with the FORCE option.



## STOP DATABASE MANAGER

The following information currently applies to multi-node environments only:

- If no parameters are specified, the database manager is stopped on each node listed in the node configuration file. The `db2diag.log` file may contain messages to indicate that other nodes are shutting down.
- Any nodes added to the MPP system since the previous STOP DATABASE MANAGER command was issued will be updated in the `db2nodes.cfg` file.
- On UNIX platforms, this command supports the SIGALRM signal, which is issued if the value specified for the `start_stop_time` database manager configuration parameter is reached. If this signal occurs, all in-progress stops are interrupted, and message SQL6037N is returned from each interrupted node to the `$HOME/sqllib/log/db2stop.timestamp.log` error log file. Nodes that are already stopped are not affected.
- The `db2cshrc` file is not supported and cannot be specified as the value for the PROFILE parameter.

**Attention:** The UNIX `kill` command should *not* be used to terminate the database manager because it will abruptly end database manager processes without controlled termination and cleanup processing.

### See Also

“DEACTIVATE DATABASE” on page 149

“DROP NODE VERIFY” on page 159

“FORCE APPLICATION” on page 167

“START DATABASE MANAGER” on page 361.

## TERMINATE

---

### TERMINATE

Explicitly terminates the command line processor's back-end process. For more information about back-end and front-end processes, see “Command Line Processor Design” on page 79.


#### Authorization

None

#### Required Connection

None

#### Command Syntax



```
▶—TERMINATE—▶
```

#### Command Parameters

None

#### Usage Notes

If an application is connected to a database, or a process is in the middle of a unit of work, **TERMINATE** causes the database connection to be lost. An internal commit is then performed.

Although **TERMINATE** and **CONNECT RESET** both break the connection to a database, only **TERMINATE** results in termination of the back-end process.

It is recommended that **TERMINATE** be issued if **db2start** and **db2stop** were executed while the back-end process was active. This prevents the back-end process from maintaining an attachment to a database manager instance that is no longer available.

Back-end processes in MPP systems must also be terminated when the **DB2NODE** environment variable is updated in the session. This environment variable is used to specify the coordinator node number within an MPP multiple logical node configuration.

---

## UNCATALOG DATABASE

Deletes a database entry from the system database directory.

### Authorization

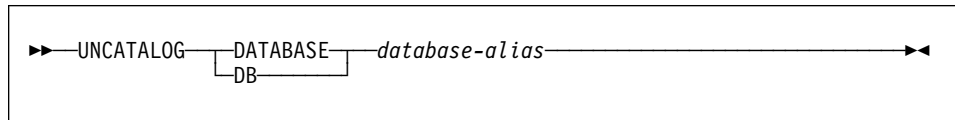
One of the following:

*sysadm*  
*sysctrl*

### Required Connection

None. Directory operations affect the local directory only.

### Command Syntax



### Command Parameters

**DATABASE** *database-alias*  
Specifies the alias of the database to uncatalog.

### Usage Notes

Only entries in the system database directory can be uncataloged. Entries in the local database directory can be deleted using “DROP DATABASE” on page 157.

To recatalog the database, use “CATALOG DATABASE” on page 118. To list the databases that are cataloged on a node, use “LIST DATABASE DIRECTORY” on page 231.

The authentication type of a database, used when communicating with a down-level server, can be changed by first uncataloging the database, and then cataloging it again with a different type.

**Note:** If directory caching is enabled (see the configuration parameter *dir\_cache* in “GET DATABASE MANAGER CONFIGURATION” on page 184), database, node, and DCS directory files are cached in memory. An application's directory cache is created during its first directory lookup. Since the cache is only refreshed when the application modifies any of the directory files, directory changes made by other applications may not be effective until the application has restarted.

To refresh the CLP's directory cache, use “TERMINATE” on page 368. To refresh DB2's shared cache, stop (**db2stop**) and then restart (**db2start**) the database. To

## UNCATALOG DATABASE

refresh the directory cache for another application, stop and then restart that application.

---

## UNCATALOG DCS DATABASE

Deletes an entry from the Database Connection Services (DCS) directory.

### Authorization

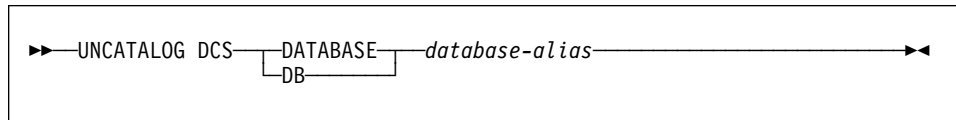
One of the following:

*sysadm*  
*sysctrl*

### Required Connection

None. Directory operations affect the local directory only.

### Command Syntax



### Command Parameters

**DATABASE** *database-alias*

Specifies the alias of the DCS database to uncatalog.

### Usage Notes

DCS databases are also cataloged in the system database directory as remote databases that can be uncataloged using “UNCATALOG DATABASE” on page 369.

To recatalog a database in the DCS directory, use “CATALOG DCS DATABASE” on page 121. To list the DCS databases that are cataloged on a node, use “LIST DCS DIRECTORY” on page 237.

**Note:** If directory caching is enabled (see the configuration parameter *dir\_cache* in “GET DATABASE MANAGER CONFIGURATION” on page 184), database, node, and DCS directory files are cached in memory. An application's directory cache is created during its first directory lookup. Since the cache is only refreshed when the application modifies any of the directory files, directory changes made by other applications may not be effective until the application has restarted.

To refresh the CLP's directory cache, use “TERMINATE” on page 368. To refresh DB2's shared cache, stop (**db2stop**) and then restart (**db2start**) the database. To refresh the directory cache for another application, stop and then restart that application.

## UNCATALOG NODE

---

### UNCATALOG NODE

Deletes an entry from the node directory.

#### Authorization

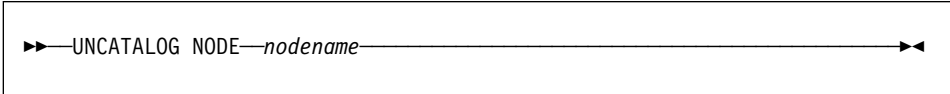
One of the following:

*sysadm*  
*sysctrl*

#### Required Connection

None. Directory operations affect the local directory only.

#### Command Syntax



A rectangular box containing the command syntax: `▶▶—UNCATALOG NODE—nodename————▶▶`. The text is centered and flanked by double arrowheads on both sides.

#### Command Parameters

**NODE** *nodename*

Specifies the node entry being uncataloged.

#### Usage Notes

UNCATALOG NODE can be executed on any type of node, but only the local directory is affected, even if there is an attachment to a remote instance, or a different local instance.

**Note:** If directory caching is enabled (see the configuration parameter *dir\_cache* in “GET DATABASE MANAGER CONFIGURATION” on page 184), database, node, and DCS directory files are cached in memory. An application’s directory cache is created during its first directory lookup. Since the cache is only refreshed when the application modifies any of the directory files, directory changes made by other applications may not be effective until the application has restarted.

To refresh the CLP’s directory cache, use “TERMINATE” on page 368. To refresh DB2’s shared cache, stop (**db2stop**) and then restart (**db2start**) the database. To refresh the directory cache for another application, stop and then restart that application.

#### See Also

“CATALOG APPC NODE” on page 111  
“CATALOG APPCLU NODE” on page 114  
“CATALOG APPN NODE” on page 116  
“CATALOG IPX/SPX NODE” on page 126

## UNCATALOG NODE

"CATALOG LOCAL NODE" on page 129  
"CATALOG NETBIOS NODE" on page 133  
"CATALOG NAMED PIPE NODE" on page 131  
"CATALOG TCP/IP NODE" on page 136.

## UNCATALOG ODBC DATA SOURCE

---

### UNCATALOG ODBC DATA SOURCE

Uncatalogs a user or system ODBC data source.

A *data source*, in ODBC (Open Database Connectivity) terminology, is a user-defined name for a specific database or file system. That name is used to access the database or file system through ODBC APIs. On Windows NT and Windows 95, either user or system data sources can be uncataloged. A user data source is only visible to the user who cataloged it, whereas a system data source is visible to and can be used by all other users.

This command is available on Windows NT, Windows 95, and Windows 3.1 only.

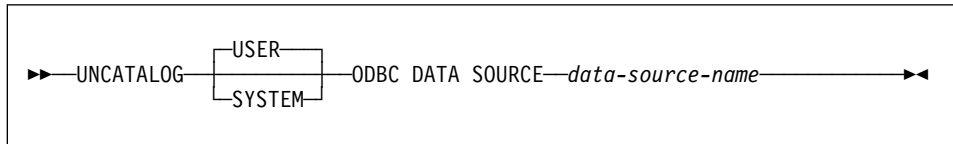
#### Authorization

None to uncatalog a user ODBC data source; *sysadm* to uncatalog a system ODBC data source.

#### Required Connection

None

#### Command Syntax



#### Command Parameters

##### USER

Uncatalog a user data source. This is the default if no keyword is specified.

##### SYSTEM

Uncatalog a system data source.

##### ODBC DATA SOURCE *data-source-name*

Specifies the name of the data source to be uncataloged. Maximum length is 32 characters.

#### See Also

“CATALOG ODBC DATA SOURCE” on page 135

“LIST ODBC DATA SOURCES” on page 251.



---

### UPDATE ADMIN CONFIGURATION

Modifies individual entries in the database manager configuration file that are relevant to the DB2 Administration Server. The DB2 Administration Server is a special DB2 instance that enables remote administration of DB2 servers. The following database manager configuration parameters can be modified:

- AGENT\_STACK\_SZ
- AUTHENTICATION
- DIAGLEVEL
- DIAGPATH
- DISCOVER
- DISCOVER\_COMM
- FILESERVER
- IPX\_SOCKET
- NNAME
- OBJECTNAME
- QUERY\_HEAP\_SZ
- SYSADM\_GROUP
- SYSCTRL\_GROUP
- SYSMANT\_GROUP
- TPNAME
- TRUST\_ALLCLNTS
- TRUST\_CLNTAUTH

**Note:** It is not recommended that the SVCENAME parameter, set by the installation program, be modified by the user. The administration server service name is set to use the DB2 registered TCP/IP port (523).

For more information about these parameters, see “GET DATABASE MANAGER CONFIGURATION” on page 184.

### Scope

This command can be issued from any node listed in the `db2nodes.cfg` file. It affects all nodes that are listed in this file.

### Authorization

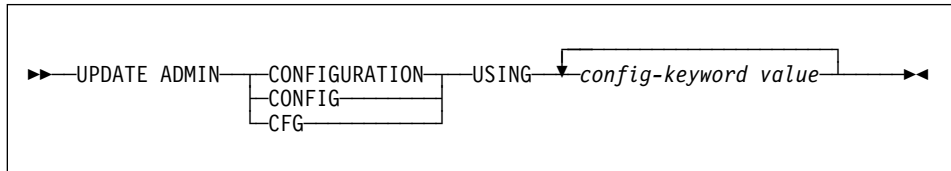
*sysadm*

### Required Connection

None or instance. An instance attachment is not required to perform local DBM configuration operations, but is required to perform remote DBM configuration operations. To update the database manager configuration for a remote instance, it is necessary to first attach to that instance.

# UPDATE ADMIN CONFIGURATION

## Command Syntax



## Command Parameters

**USING** *config-keyword value*

Specifies the admin configuration parameter to be updated.

## Usage Notes

To view or print a list of the admin configuration parameters, use “GET ADMIN CONFIGURATION” on page 169.

To reset the admin configuration parameters to the recommended database manager defaults, use “RESET ADMIN CONFIGURATION” on page 327.

For more information about admin configuration parameters, see the *Administration Guide*.

The values of these parameters differ for each type of database node configured (server, client, or server with remote clients). See the *Administration Guide*, or one of the *Quick Beginnings* books for the ranges and the default values that can be set on each node type.

Changes to the database manager configuration file become effective only after they are loaded into memory. This occurs during execution of **db2start**.

If an error occurs, the database manager configuration file does not change.

The database manager configuration file cannot be updated if the checksum is invalid. This may occur if the database manager configuration file is changed without using the appropriate command. If this happens, the database manager must be reinstalled to reset the database manager configuration file.

## See Also

“GET ADMIN CONFIGURATION” on page 169

“RESET ADMIN CONFIGURATION” on page 327.

---

## UPDATE COMMAND OPTIONS

Sets one or more command options during an interactive session, or from a batch input file. The settings revert to system defaults (or the system default overrides in **DB2OPTIONS**) when the interactive session or batch input file ends.

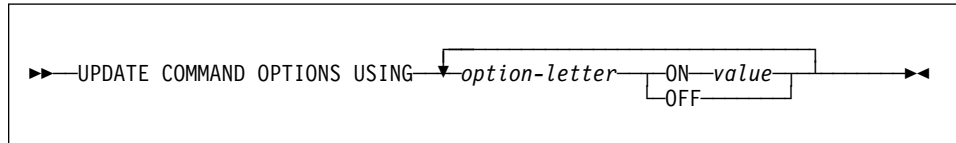
### Authorization

None

### Required Connection

None

### Command Syntax



### Command Parameters

#### **USING** *option-letter*

The following option-letters can be set:

- a** Display SQLCA
- c** Auto-commit SQL statements
- e** Display SQLCODE/SQLSTATE
- l** Log commands in a history file
- o** Display to standard output
- p** Display DB2 interactive prompt
- r** Save output report to a file
- s** Stop execution on command error
- v** Echo current command
- w** Show SQL statement warning messages
- z** Redirect all output to a file.

#### **ON** *value*

The e, l, r, and z options require a value if they are turned on. For the e option, *value* can be c to display the SQLCODE, or s to display the SQLSTATE. For the l, r, and z options, *value* represents the name to be used for the history file or the report file. No other options accept a value.

## UPDATE COMMAND OPTIONS

### Usage Notes

These settings override system defaults, settings in **DB2OPTIONS**, and options specified using the command line option flags.

The file input option (-f) and the statement termination option (-t) cannot be updated using this command.

To view the current option settings, use “LIST COMMAND OPTIONS” on page 229.

For detailed information about these options, see “Command Line Processor Invocation and Options” on page 69.

---

## UPDATE DATABASE CONFIGURATION

Modifies individual entries in a specific database configuration file.

A database configuration file resides on every node on which the database has been created.

### Scope

This command only affects the node on which it is executed.

### Authorization

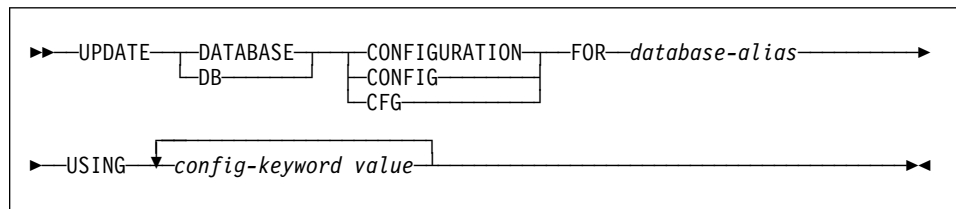
One of the following:

*sysadm*  
*sysctrl*  
*sysmaint*

### Required Connection

Instance. An explicit attachment is not required. If the database is listed as remote, an instance attachment to the remote node is established for the duration of the command.

### Command Syntax



### Command Parameters

**FOR** *database-alias*

Specifies the alias of the database whose configuration is to be updated.

**USING** *config-keyword value*

Specifies the database configuration parameter to be updated. For a brief description of configurable parameters, see “GET DATABASE CONFIGURATION” on page 175.

### Usage Notes

To view or print a list of the database configuration parameters, use “GET DATABASE CONFIGURATION” on page 175.

To reset the database configuration parameters to the recommended database manager defaults, use “RESET DATABASE CONFIGURATION” on page 329.

## UPDATE DATABASE CONFIGURATION

For more information about DB2's configuration parameters, see the *Administration Guide*.

The values of these parameters differ for each type of database node configured (server, client, or server with remote clients). See the *Administration Guide* for the ranges and the default values that can be set on each node type.

Not all parameters can be updated.

Changes to the database configuration file become effective only after they are loaded into memory. All applications must disconnect from the database before this can occur.

If an error occurs, the database configuration file does not change.

The database configuration file cannot be updated if the checksum is invalid. This may occur if the database configuration file is changed without using the appropriate command. If this happens, the database must be restored to reset the database configuration file.

### See Also

"GET DATABASE CONFIGURATION" on page 175

"RESET DATABASE CONFIGURATION" on page 329.

# UPDATE DATABASE MANAGER CONFIGURATION

---

## UPDATE DATABASE MANAGER CONFIGURATION

Modifies individual entries in the database manager configuration file.

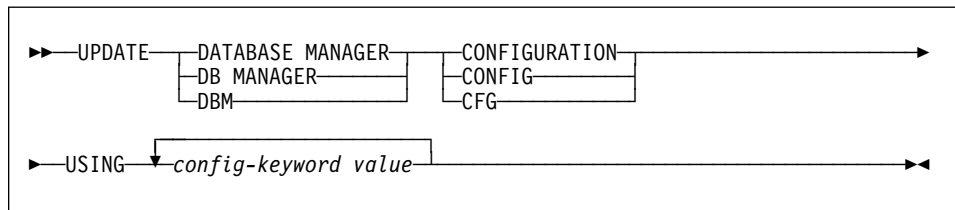
### Authorization

*sysadm*

### Required Connection

None or instance. An instance attachment is not required to perform local DBM configuration operations, but is required to perform remote DBM configuration operations. To update the database manager configuration for a remote instance, it is necessary to first attach to that instance.

### Command Syntax



### Command Parameters

**USING** *config-keyword value*

Specifies the database manager configuration parameter to be updated. For a brief description of configurable parameters, see “GET DATABASE MANAGER CONFIGURATION” on page 184.

### Usage Notes

To view or print a list of the database manager configuration parameters, use “GET DATABASE MANAGER CONFIGURATION” on page 184.

To reset the database manager configuration parameters to the recommended database manager defaults, use “RESET DATABASE MANAGER CONFIGURATION” on page 331.

For more information about database manager configuration parameters, see the *Administration Guide*.

The values of these parameters differ for each type of database node configured (server, client, or server with remote clients). See the *Administration Guide* for the ranges and the default values that can be set on each node type.

Not all parameters can be updated.

## UPDATE DATABASE MANAGER CONFIGURATION

Changes to the database manager configuration file become effective only after they are loaded into memory. For a server configuration parameter, this occurs during execution of **db2start**. For a client configuration parameter, this occurs when the application is restarted. If the client is the command line processor, it is necessary to invoke "TERMINATE" on page 368.

If an error occurs, the database manager configuration file does not change.

The database manager configuration file cannot be updated if the checksum is invalid. This may occur if the database manager configuration file is changed without using the appropriate command. If this happens, the database manager must be reinstalled to reset the database manager configuration file.

### See Also

"GET DATABASE MANAGER CONFIGURATION" on page 184

"RESET DATABASE MANAGER CONFIGURATION" on page 331.



### UPDATE MONITOR SWITCHES

Turns one or more database monitor recording switches on or off. When the database manager starts, the settings of the six switches are determined by the *dft\_mon* database manager configuration parameters (see “GET DATABASE MANAGER CONFIGURATION” on page 184).

The database monitor records a base set of information at all times. Users who require more than this basic information can turn on the appropriate switches, but at a cost to system performance. The amount of information available in output from “GET SNAPSHOT” on page 199 reflects which, if any, switches are on.

### Authorization

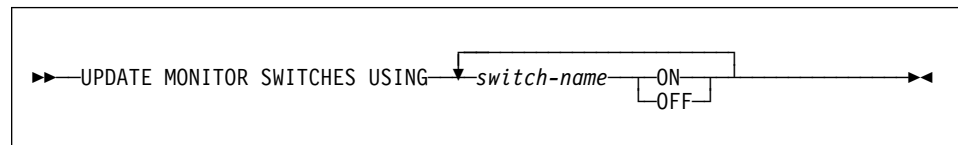
One of the following:

*sysadm*  
*sysctrl*  
*sysmaint*

### Required Connection

Instance. To update the monitor switches at a remote instance (or a different local instance), it is necessary to first attach to that instance.

### Command Syntax



### Command Parameters

**USING** *switch-name*

The following switch names are available:

<b>BUFFERPOOL</b>	Buffer pool activity information
<b>LOCK</b>	Lock information
<b>SORT</b>	Sorting information
<b>STATEMENT</b>	SQL statement information
<b>TABLE</b>	Table activity information
<b>UOW</b>	Unit of work information.

## UPDATE MONITOR SWITCHES

### Usage Notes

Information is collected by the database manager only after a switch is turned on. The switches remain set until **db2stop** is issued. To clear the information related to a particular switch, set the switch off, then on.

Updating switches in one application does not affect other applications.

To view the switch settings, use "GET MONITOR SWITCHES" on page 197.

---

## UPDATE RECOVERY HISTORY FILE

Updates the location, device type, or comment in a recovery history file entry.

### Authorization

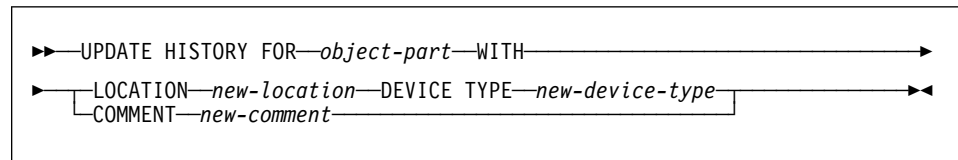
One of the following:

*sysadm*  
*sysctrl*  
*sysmaint*  
*dbadm*

### Required Connection

Database

### Command Syntax



### Command Parameters

**FOR** *object-part*

Specifies the identifier for the backup or copy image. It is a time stamp with a sequence number from 001 to 999.

**LOCATION** *new-location*

Specifies the new physical location of a backup. The interpretation of this parameter depends on the device type.

**DEVICE TYPE** *new-device-type*

Specifies a new device type for storing the backup. Valid device types are:

**D** Disk  
**K** Diskette  
**T** Tape  
**A** ADSM  
**U** User exit  
**O** Other.

**COMMENT** *new-comment*

Specifies a new comment to describe the entry.

### Example

To update the history file entry for the full database backup taken on April 13, 1997 at 10:00 a.m., enter:

```

db2 update history for 19970413100000001 with
location /backup/dbbackup.1 device type d
  
```

## UPDATE RECOVERY HISTORY FILE

### Usage Notes

The recovery history file is used for record keeping purposes only. It is not used during database recovery.

### See Also

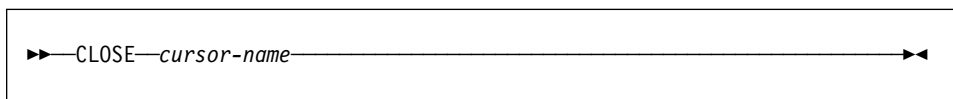
“PRUNE HISTORY” on page 303.

## Chapter 4. Using Command Line SQL Statements

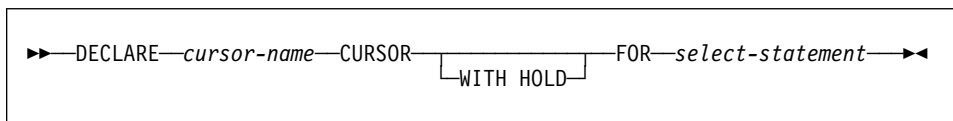
This section provides information about using Structured Query Language (SQL) statements from the command line. These statements can be executed directly from an operating system command prompt, and can be used to define and manipulate information stored in a database table, index, or view in much the same way as if the commands were written into an application program. Information can be added, deleted, or updated, and reports can be generated from the contents of tables.

All SQL statements that can be executed through the command line processor are listed in the CLP column of Table 10 on page 390. The syntax of all the SQL statements, whether executed from the command line or embedded in a source program, is described in the *SQL Reference*. The syntax of many embedded SQL statements and CLP SQL statements is identical. However, host variables, parameter markers, descriptor-names, and statement-names are applicable only to embedded SQL. The syntax of CLOSE, DECLARE CURSOR, FETCH, OPEN, and SELECT *does* depend on whether these statements are embedded or executed through the CLP. The CLP syntax of these statements is provided below:

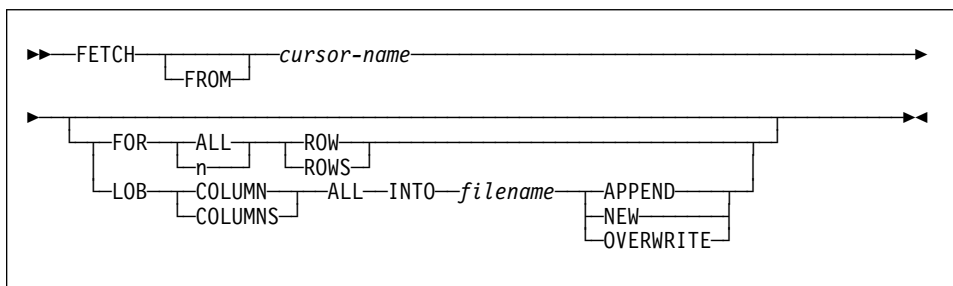
### CLOSE



### DECLARE CURSOR



### FETCH





**Notes:**

1. When FETCH or SELECT is issued through the command line processor, decimal and floating-point numbers are displayed with the country's decimal delimiter, that is, a period (.) in the U.S., Canada, and the U.K.; a comma (,) in most other countries. However, when INSERT, UPDATE, and other SQL statements are issued through the command line processor to update tables, a period must be used as the decimal delimiter, even in countries that use a comma for that purpose.
2. When FETCH or SELECT is issued through the command line processor, null values are typically displayed as a hyphen (-). For databases configured with DFT\_SQLMATHWARN YES, expressions that result in an arithmetic error are processed as null values. Such arithmetic error nulls are displayed as a plus (+).

For example, create and populate table t1 as follows:

```
create table t1 (i1 int , i2 int);
insert into t1 values (1,1),(2,0),(3,null);
```

The statement: select i1/i2 from t1 generates the following result:

```
1
---
1
+
-
3 records selected
```

3. A new LOB option has been added to FETCH. If the LOB clause is specified, only the next row is fetched:
  - Each LOB column value is fetched into a file with the name *filename.xxx*, where *filename* is specified in the LOB clause, and *xxx* is a file extension from 001 to 999 (001 is the first LOB column in the select list of the corresponding DECLARE CURSOR statement, 002 is the second LOB column, and 999 is the 999th column). The maximum number of LOB columns that can be fetched into files is 999.
  - Names of the files containing the data are displayed in the LOB columns.

Change the way that the CLP displays data (when querying databases using SQL statements through the CLP) by rebinding the CLP bind files against the database being queried. For example, to display date and time in ISO format, do the following:

1. Create a text file containing the names of the CLP bind files. This file is used as the list file for binding multiple files with one BIND command. In this example the file is named *clp.1st*, and its contents are:

```

db2c1pcs.bnd +
db2c1prp.bnd +
db2c1pur.bnd +
db2c1prs.bnd +
db2c1pns.bnd

```

2. Connect to the database.
3. Issue the following command:

```
db2 bind @c1p.1st collection nullid datetime iso
```

For detailed information about the command line processor, see Chapter 2, “Command Line Processor (CLP)” on page 69. For more information about the syntax of SQL statements and the function provided by SQL statements, see the *SQL Reference*. For information about reading syntax diagrams, see Appendix A, “How to Read the Syntax Diagrams” on page 393.

Table 10 (Page 1 of 3). SQL Statements (DB2 Universal Database)

SQL Statement	Dynamic <sup>1</sup>	Command Line Processor (CLP)	Call Level Interface <sup>3</sup> (CLI)
ALTER { BUFFERPOOL, NODEGROUP, TABLE, TABLESPACE }	X	X	X
BEGIN DECLARE SECTION <sup>2</sup>			
CALL			X <sup>4</sup>
CLOSE		X	SQLCloseCursor(), SQLFreeStmt()
COMMENT ON	X	X	X
COMMIT	X	X	SQLEndTran, SQLTransact()
Compound SQL			X <sup>4</sup>
CONNECT (Type 1)		X	SQLBrowseConnect(), SQLConnect(), SQLDriverConnect()
CONNECT (Type 2)		X	SQLBrowseConnect(), SQLConnect(), SQLDriverConnect()
CREATE { ALIAS, BUFFERPOOL, DISTINCT TYPE, EVENT MONITOR, FUNCTION, INDEX, NODEGROUP, PROCEDURE, SCHEMA, TABLE, TABLESPACE, TRIGGER, VIEW }	X	X	X
DECLARE CURSOR <sup>2</sup>		X	SQLAllocStmt()
DELETE	X	X	X
DESCRIBE <sup>8</sup>		X	SQLColAttributes(), SQLDescribeCol(), SQLDescribeParam() <sup>6</sup>
DISCONNECT		X	SQLDisconnect()
DROP	X	X	X



Table 10 (Page 2 of 3). SQL Statements (DB2 Universal Database)

SQL Statement	Dynamic <sup>1</sup>	Command Line Processor (CLP)	Call Level Interface <sup>3</sup> (CLI)
END DECLARE SECTION <sup>2</sup>			
EXECUTE			SQLExecute()
EXECUTE IMMEDIATE			SQLExecDirect()
EXPLAIN	X	X	X
FETCH		X	SQLExtendedFetch() <sup>7</sup> , SQLFetch(), SQLFetchScroll() <sup>7</sup>
FREE LOCATOR			X <sup>4</sup>
GRANT	X	X	X
INCLUDE <sup>2</sup>			
INSERT	X	X	X
LOCK TABLE	X	X	X
OPEN		X	SQLExecute(), SQLExecDirect()
PREPARE			SQLPrepare()
RELEASE		X	
RENAME TABLE	X	X	X
REVOKE	X	X	X
ROLLBACK	X	X	SQLEndTran(), SQLTransact()
select-statement	X	X	X
SELECT INTO			
SET CONNECTION		X	SQLSetConnection()
SET CONSTRAINTS	X	X	X
SET CURRENT DEGREE	X	X	X
SET CURRENT EXPLAIN MODE	X	X	X, SQLSetConnectAttr()
SET CURRENT EXPLAIN SNAPSHOT	X	X	X, SQLSetConnectAttr()
SET CURRENT FUNCTION PATH	X	X	X
SET CURRENT PACKAGESET			
SET CURRENT QUERY OPTIMIZATION	X	X	X
SET EVENT MONITOR STATE	X	X	X
SET transition-variable <sup>5</sup>	X	X	X
SIGNAL SQLSTATE <sup>5</sup>	X	X	X
UPDATE	X	X	X
VALUES INTO			
WHENEVER <sup>2</sup>			

Table 10 (Page 3 of 3). SQL Statements (DB2 Universal Database)

SQL Statement	Dynamic <sup>1</sup>	Command Line Processor (CLP)	Call Level Interface <sup>3</sup> (CLI)
---------------	----------------------	---------------------------------------	---

**Notes:**

1. You can code all statements in this list as static SQL, but only those marked with X as dynamic SQL.
2. You cannot execute this statement.
3. An X indicates that you can execute this statement using either `SQLExecDirect()` or `SQLPrepare()` and `SQLExecute()`. If there is an equivalent DB2 CLI function, the function name is listed.
4. Although this statement is not dynamic, with DB2 CLI you can specify this statement when calling either `SQLExecDirect()`, or `SQLPrepare()` and `SQLExecute()`.
5. You can only use this within CREATE TRIGGER statements.
6. You can only use the SQL DESCRIBE statement to describe output, whereas with DB2 CLI you can also describe input (using the `SQLDescribeParam()` function).
7. You can only use the SQL FETCH statement to fetch one row at a time in one direction, whereas with the DB2 CLI `SQLExtendedFetch()` and `SQLFetchScroll()` functions, you can fetch into arrays. Furthermore, you can fetch in any direction, and at any position in the result set.
8. The DESCRIBE SQL statement has a different syntax than that of the CLP DESCRIBE command. For information on the DESCRIBE SQL statement, refer to the *SQL Reference*. For information on the DESCRIBE CLP command, refer to the *Command Reference*.

## Appendix A. How to Read the Syntax Diagrams

A syntax diagram shows how a command should be specified so that the operating system can correctly interpret what is typed.

Read a syntax diagram from left to right, and from top to bottom, following the horizontal line (the main path). If the line ends with an arrowhead, the command syntax is continued, and the next line starts with an arrowhead. A vertical bar marks the end of the command syntax.

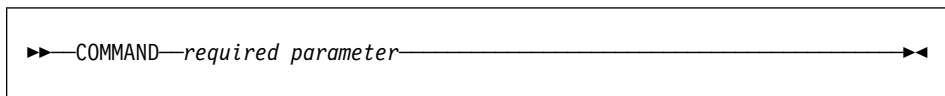
When typing information from a syntax diagram, be sure to include punctuation, such as quotation marks and equal signs.

Parameters are classified as keywords or variables:

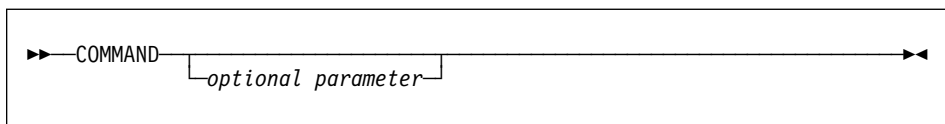
- Keywords represent constants, and are shown in uppercase letters; at the command prompt, however, keywords can be entered in upper, lower, or mixed case. A command name is an example of a keyword.
- Variables represent names or values that are supplied by the user, and are shown in lowercase letters; at the command prompt, however, variables can be entered in upper, lower, or mixed case, unless case restrictions are explicitly stated. A file name is an example of a variable.

A parameter can be a combination of a keyword and a variable.

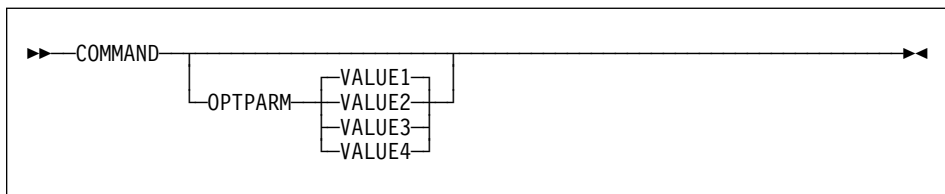
Required parameters are displayed on the main path:



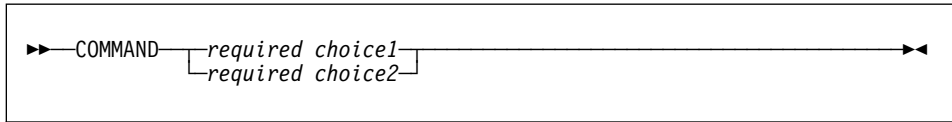
Optional parameters are displayed below the main path:



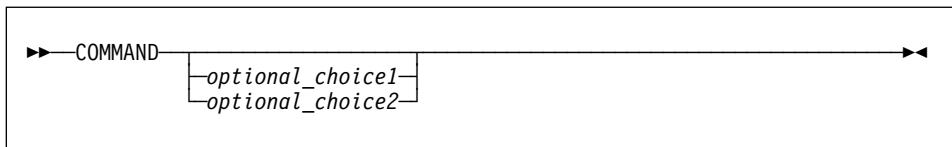
A parameter's default value is displayed above the path:



A stack of parameters, with the first parameter displayed on the main path, indicates that one of the parameters must be selected:

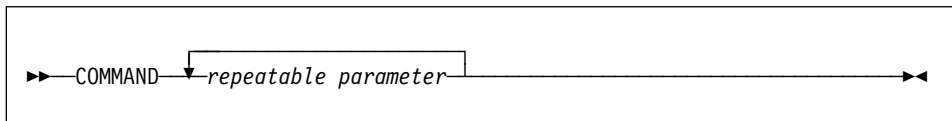


A stack of parameters, with the first parameter displayed below the main path, indicates that one of the parameters can be selected:

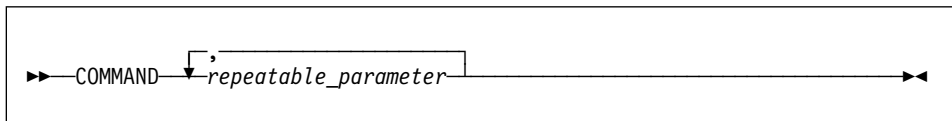


An arrow returning to the left, above the path, indicates that items can be repeated in accordance with the following conventions:

- If the arrow is uninterrupted, the item can be repeated in a list with the items separated by blank spaces:

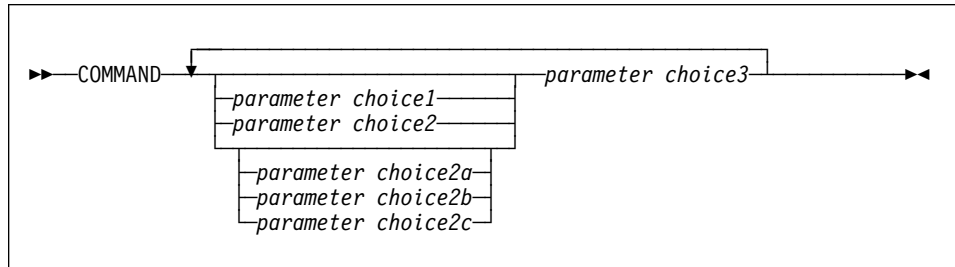


- If the arrow contains a comma, the item can be repeated in a list with the items separated by commas:

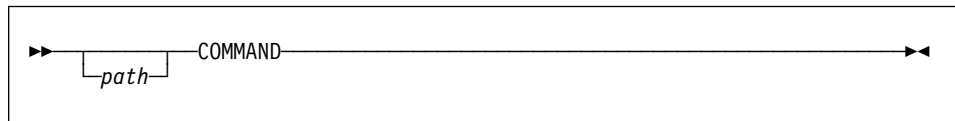


Items from parameter stacks can be repeated in accordance with the stack conventions for required and optional parameters discussed previously.

Some syntax diagrams contain parameter stacks within other parameter stacks. Items from stacks can only be repeated in accordance with the conventions discussed previously. That is, if an inner stack does not have a repeat arrow above it, but an outer stack does, only one parameter from the inner stack can be chosen and combined with any parameter from the outer stack, and that combination can be repeated. For example, the following diagram shows that one could combine parameter *choice2a* with parameter *choice2*, and then repeat that combination again (*choice2* plus *choice2a*):

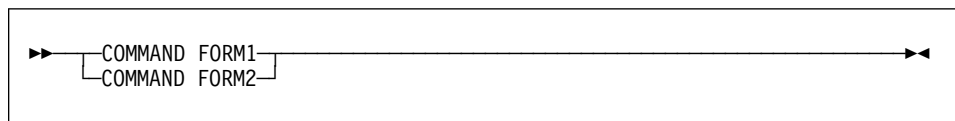


Some commands are preceded by an optional path parameter:



If this parameter is not supplied, the system searches the current directory for the command. If it cannot find the command, the system continues searching for the command in all the directories on the paths listed in the `.profile`.

Some commands have syntactical variants that are functionally equivalent:





---

## Appendix B. Naming Conventions

This section provides information about the conventions that apply when naming database manager objects, such as databases and tables, and authentication IDs.

- Character strings that represent names of database manager objects can contain any of the following: a-z, A-Z, 0-9, @, #, and \$.
- The first character in the string must be an alphabetic character, @, #, or \$; it cannot be a number or the letter sequences SYS, DBM, or IBM.
- Unless otherwise noted, names can be entered in lowercase letters; however, the database manager processes them as if they were uppercase.

The exception to this is character strings that represent names under the systems network architecture (SNA). Many values, such as logical unit names (partner\_lu and local\_lu), are case sensitive. The name must be entered exactly as it appears in the SNA definitions that correspond to those terms.

- A database name or database alias is a unique character string containing from one to eight letters, numbers, or keyboard characters from the set described above.

Databases are cataloged in the system and local database directories by their aliases in one field, and their original name in another. For most functions, the database manager uses the name entered in the alias field of the database directories. (The exceptions are CHANGE DATABASE COMMENT and CREATE DATABASE, where a directory path must be specified.)

- The long identifier or alias for a database table or view, and the name of a column within a table or a view, are unique character strings 1 to 18 characters in length.

A fully qualified table name consists of the *schema.tablename*. The schema is the unique user ID under which the table was created.

- Authentication IDs (both user IDs and group IDs) cannot exceed eight characters in length.

For more information about naming conventions, see the *Administration Guide*.





## Appendix C. IMPORT/EXPORT/LOAD Utility File Formats

Three operating system file formats supported by the database manager for IMPORT, EXPORT, and LOAD are described. They are:

**DEL** Delimited ASCII, for exchange with many database managers and file managers.

**PC/IXF** PC version of IXF, an exchange format used by the database manager.

**ASC** Non-delimited ASCII (IMPORT only).

**Note:** Throughout this section, all comments that pertain to IMPORT are also applicable to LOAD.

### Delimited ASCII (DEL) File Format

A Delimited ASCII (DEL) file is a sequential ASCII file with row and column delimiters. Each DEL file is a stream of ASCII characters consisting of cell values ordered by row, and then by column. Rows in the data stream are separated by row delimiters; within each row, individual cell values are separated by column delimiters.

The following table describes the format of DEL files that can be imported into, or that can be generated as the result of an export action.

```

DEL file ::= Row 1 data || Row delimiter ||
           Row 2 data || Row delimiter ||
           .
           .
           Row n data || Optional row delimiter

Row i data ::= Cell value(i,1) || Column delimiter ||
              Cell value(i,2) || Column delimiter ||
              .
              .
              Cell value(i,m)

Column delimiter ::= Default value ASCII comma (,)a

Row delimiter ::= ASCII line feed sequenceb
    
```

## Delimited ASCII (DEL) File Format

```
Cell value(i,j) ::= Leading spaces
                  || ASCII representation of a numeric value
                  || (integer, decimal, or float)
                  || Character string enclosed by character string delimiters
                  || Character string
                  || Trailing spaces

Character string delimiter ::= Default value ASCII double quotation
                              marks (")c

End-of-file character ::= Hex '1A' (OS/2 or the Windows operating system only)

ASCII representation of a numeric valued ::= Optional sign '+' or '-'
      || 1 to 31 decimal digits with an optional decimal point before,
      || after, or between two digits
      || Optional exponent

Exponent ::= Character 'E' or 'e'
      || Optional sign '+' or '-'
      || 1 to 3 decimal digits with no decimal point

Decimal digit ::= Any one of the characters '0', '1', ... '9'

Decimal point ::= Default value ASCII period (.)e
```

<sup>a</sup> The column delimiter can be specified with the COLDEL option.

<sup>b</sup> The record delimiter is assumed to be a new line character, ASCII x0A. Data generated on OS/2 or the Windows operating system can use the carriage return/line feed 2-byte standard of 0x0D0A. Data in EBCDIC code pages should use the EBCDIC LF character (0x25) as the record delimiter (EBCDIC data can be loaded using the CODEPAGE option on the LOAD command).

<sup>c</sup> The character string delimiter can be specified with the CHARDEL option.

<sup>d</sup> If the ASCII representation of a numeric value contains an exponent, it is a FLOAT constant. If it has a decimal point but no exponent, it is a DECIMAL constant. If it has no decimal point and no exponent, it is an INTEGER constant.

<sup>e</sup> The decimal point character can be specified with the DECPT option.

### Sample DEL File

Following is an example of a DEL file. Each line ends with a line feed sequence (on OS/2 or the Windows operating system, each line ends with a carriage return/line feed sequence).

```
"Smith, Bob",4973,15.46
"Jones, Bill",12345,16.34
"Williams, Sam",452,193.78
```

The following example illustrates the use of non-delimited character strings. The column delimiter has been changed to a semicolon, because the character data contains a comma.

## Delimited ASCII (DEL) File Format

Smith, Bob;4973;15.46  
Jones, Bill;12345;16.34  
Williams, Sam;452;193.78

### Notes:

1. A space (X'20') is never a valid delimiter.
2. Spaces that precede the first character, or that follow the last character of a cell value, are discarded during import. Spaces that are embedded in a cell value are not discarded.
3. A period (.) is not a valid character string delimiter, because it conflicts with periods in time stamp values.
4. For pure DBCS (graphic), mixed DBCS, and EUC, delimiters are restricted to the range of x00 to x3F, inclusive.
5. For DEL data specified in an EBCDIC code page, the delimiters may not coincide with the shift-in and shift-out DBCS characters.
6. On OS/2 or the Windows operating system, the first occurrence of an end-of-file character (X'1A') that is not within character delimiters indicates the end-of-file. Any subsequent data is not imported.
7. If the first character of a cell value is the character string delimiter, the cell value is imported as a *delimited* character string. The string is terminated by the next occurrence of the character string delimiter; characters that follow this second delimiter but precede the next column delimiter are discarded during import.

Import of a delimited character string which contains a character string delimiter thus produces erroneous results. An attempt to export character data containing a character string delimiter causes a warning message. The import and export utilities permit the user to change the character string delimiter, thereby offering flexibility in resolving such problems.

A *non-delimited* character string is a cell value whose first character is not the character string delimiter. A non-delimited character string is terminated by the next occurrence of a column or a row delimiter, and thus may contain the character string delimiter. However, a non-delimited character string should not contain a column delimiter, a carriage return (on OS/2 or the Windows operating system), or the line feed sequence.

8. A null value is indicated by the absence of a cell value where one would normally occur, or by a string of spaces.
9. Since some products restrict character fields to 254 or 255 bytes, the export utility generates a warning message whenever a character column of maximum length greater than 254 bytes is selected for export. The import utility accommodates fields that are as long as the longest LONG VARCHAR and LONG VARGRAPHIC columns.

## Delimited ASCII (DEL) File Format

### DEL Data Type Descriptions

<i>Table 11 (Page 1 of 3). Acceptable Data Type Forms for the DEL File Format</i>		
<b>Data Type</b>	<b>Form in Files Created by the Export Utility</b>	<b>Form Acceptable to the Import Utility</b>
BLOB, CLOB	Character data enclosed by character delimiters (for example, double quotation marks).	A delimited or non-delimited character string. The character string is used as the database column value.
BLOB_FILE, CLOB_FILE	The character data for each BLOB/CLOB column is stored in individual files, and the file name is enclosed by character delimiters.	The delimited or non-delimited name of the file that holds the data.
CHAR	Character data enclosed by character delimiters (for example, double quotation marks).	A delimited or non-delimited character string. The character string is truncated or padded with spaces (X'20'), if necessary, to match the width of the database column.
DATE	<i>yyyymmdd</i> (year month day) with no character delimiters. For example: 19931029  Alternatively, the DATESISO option can be used to specify that all date values are to be exported in ISO format.	A delimited or non-delimited character string containing a date value in an ISO format consistent with the country code of the target database, or a non-delimited character string of the form <i>yyyymmdd</i> .
DBCLOB (DBCS only)	Graphic data is exported as a delimited character string.	A delimited or non-delimited character string, an even number of bytes in length. The character string is used as the database column value.
DBCLOB_FILE (DBCS only)	The character data for each DBCLOB column is stored in individual files, and the file name is enclosed by character delimiters.	The delimited or non-delimited name of the file that holds the data.

## Delimited ASCII (DEL) File Format

Table 11 (Page 2 of 3). Acceptable Data Type Forms for the DEL File Format

Data Type	Form in Files Created by the Export Utility	Form Acceptable to the Import Utility
DECIMAL	A DECIMAL constant with the precision and scale of the field being exported. The DECPLUSBLANK option can be used to specify that positive decimal values are to be prefixed with a blank space instead of a plus sign (+).	ASCII representation of a numeric value that does not overflow the range of the database column into which the field is being imported. If the input value has more digits after the decimal point than can be accommodated by the database column, the excess digits are truncated.
FLOAT(long)	A FLOAT constant in the range -10E307 to 10E307.	ASCII representation of a numeric value in the range -10E307 to 10E307.
GRAPHIC (DBCS only)	Graphic data is exported as a delimited character string.	A delimited or non-delimited character string, an even number of bytes in length. The character string is truncated or padded with double-byte spaces (for example, X'8140'), if necessary, to match the width of the database column.
INTEGER	An INTEGER constant in the range -2 147 483 648 to 2 147 483 647.	ASCII representation of a numeric value in the range -2 147 483 648 to 2 147 483 647. Decimal and float numbers are truncated to integer values.
LONG VARCHAR	Character data enclosed by character delimiters (for example, double quotation marks).	A delimited or non-delimited character string. The character string is used as the database column value.
LONG VARGRAPHIC (DBCS only)	Graphic data is exported as a delimited character string.	A delimited or non-delimited character string, an even number of bytes in length. The character string is used as the database column value.
SMALLINT	An INTEGER constant in the range -32 768 to 32 767.	ASCII representation of a numeric value in the range -32 768 to 32 767. Decimal and float numbers are truncated to integer values.

## PC Version of IXF File Format

Data Type	Form in Files Created by the Export Utility	Form Acceptable to the Import Utility
TIME	<i>hh.mm.ss</i> (hour minutes seconds). A time value in ISO format enclosed by character delimiters. For example: "09.39.43"	A delimited or non-delimited character string containing a time value in a format consistent with the country code of the target database.
TIMESTAMP	<i>yyyy-mm-dd-hh.mm.ss.nnnnnn</i> (year month day hour minutes seconds microseconds). A character string representing a date and time enclosed by character delimiters.	A delimited or non-delimited character string containing a time stamp value acceptable for storage in a database.
VARCHAR	Character data enclosed by character delimiters (for example, double quotation marks).	A delimited or non-delimited character string. The character string is truncated, if necessary, to match the maximum width of the database column.
VARGRAPHIC (DBCS only)	Graphic data is exported as a delimited character string.	A delimited or non-delimited character string, an even number of bytes in length. The character string is truncated, if necessary, to match the maximum width of the database column.

## PC Version of IXF File Format

The PC version of IXF (PC/IXF) file format is a database manager adaptation of the Integration Exchange Format (IXF) data interchange architecture. The IXF architecture was specifically designed to enable the exchange of relational database structures and data. The PC/IXF architecture allows the database manager to export a database without having to anticipate the requirements and idiosyncrasies of a receiving product. Similarly, a product importing a PC/IXF file need only understand the PC/IXF architecture; the characteristics of the product which exported the file are not relevant. The PC/IXF file architecture maintains the independence of both the exporting and the importing database systems.

The IXF architecture is a generic relational database exchange format that supports a rich set of relational data types, including some types that may not be supported by specific relational database products. The PC/IXF file format preserves this flexibility; for example, the PC/IXF architecture supports both single-byte character string (SBCS) and double-byte character string (DBCS) data types. Not all implementations support all PC/IXF data types; however, even restricted implementations provide for the detection and disposition of unsupported data types during import.

## PC Version of IXF File Format

In general, a PC/IXF file consists of an unbroken sequence of variable-length records. The file contains the following record types in the order shown:

- One header record of record type H
- One table record of record type T
- Multiple column descriptor records of record type C (one record for each column in the table)
- Multiple data records of record type D (each row in the table is represented by one or more D records).

A PC/IXF file may also contain application records of record type A, anywhere after the H record. These records are permitted in PC/IXF files to enable an application to include additional data, not defined by the PC/IXF format, in a PC/IXF file. A records are ignored by any program reading a PC/IXF file that does not have particular knowledge about the data format and content implied by the application identifier in the A record.

Every record in a PC/IXF file begins with a record length indicator. This is a 6-byte right justified character representation of an integer value specifying the length, in bytes, of the portion of the PC/IXF record that follows the record length indicator; that is, the total record size minus 6 bytes. Programs reading PC/IXF files should use these record lengths to locate the end of the current record and the beginning of the next record. H, T, and C records must be sufficiently large to include all of their defined fields, and, of course, their record length fields must agree with their actual lengths. However, if extra data (for example, a *new* field), is added to the end of one of these records, pre-existing programs reading PC/IXF files should ignore the extra data, and generate no more than a warning message. Programs writing PC/IXF files, however, should write H, T and C records that are the precise length needed to contain all of the defined fields.

PC/IXF file records are composed of fields which contain character data. The import and export utilities interpret this character data using the CPGID of the target database, with two exceptions:

- The IXFADATA field of A records.

The code page environment of character data contained in an IXFADATA field is established by the application which creates and processes a particular A record; that is, the environment varies by implementation.

- The IXFDCOLS field of D records.

The code page environment of character data contained in an IXFDCOLS field is a function of information contained in the C record which defines a particular column and its data.

Numeric fields in H, T, and C records, and in the prefix portion of D and A records should be right justified single-byte character representations of integer values, filled with leading zeros or blanks. A value of zero should be indicated with at least one (right justified) zero character, not blanks. Whenever one of these numeric fields is not used, for example IXFCLENG, where the length is implied by the data type, it should be filled with blanks. These numeric fields are:

## PC Version of IXF File Format

IXFHRECL, IXFTRECL, IXFCRECL, IXFDRECL, IXFARECL,  
IXFHHCNT, IXFHBCP, IXFHDBCP, IXFTCNT, IXFTNAML,  
IXFCLENG, IXFCDRID, IXFCPOSN, IXFCNAML, IXFCTYPE,  
IXFCSBCP, IXFCBCP, IXFCNDIM, IXFCDSIZ, IXFDRID

**Note:** The database manager PC/IXF file format is not identical to the System/370 IXF format (see “Differences between Version 1 PC/IXF and Version 0 System/370 IXF” on page 435).

## PC/IXF Record Types

There are five PC/IXF record types:

- header
- table
- column descriptor
- data
- application

Each PC/IXF record type is defined as a sequence of fields; these fields are required, and must appear in the order shown.

HEADER RECORD			
FIELD NAME	LENGTH	TYPE	COMMENTS
-----	-----	-----	-----
IXFHRECL	06-BYTE	CHARACTER	record length
IXFHRECT	01-BYTE	CHARACTER	record type = 'H'
IXFHID	03-BYTE	CHARACTER	IXF identifier
IXFHVERS	04-BYTE	CHARACTER	IXF version
IXFHPROD	12-BYTE	CHARACTER	product
IXFHDATE	08-BYTE	CHARACTER	date written
IXFHTIME	06-BYTE	CHARACTER	time written
IXFHHCNT	05-BYTE	CHARACTER	heading record count
IXFHBCP	05-BYTE	CHARACTER	single byte code page
IXFHDBCP	05-BYTE	CHARACTER	double byte code page
IXHFIL1	02-BYTE	CHARACTER	reserved

The following fields are contained in the header record:

- IXFHRECL** The record length indicator. A 6-byte character representation of an integer value specifying the length, in bytes, of the portion of the PC/IXF record that follows the record length indicator; that is, the total record size minus 6 bytes. The H record must be sufficiently long to include all of its defined fields.
- IXFHRECT** The IXF record type, which is set to H for this record.
- IXFHID** The file format identifier, which is set to IXF for this file.



## PC Version of IXF File Format

<b>IXFHVERS</b>	The PC/IXF format level used when the file was created, which is set to '0001'.
<b>IXFHPROD</b>	A field that can be used by the program creating the file to identify itself. If this field is filled in, the first six bytes are used to identify the product creating the file, and the last six bytes are used to indicate the version or release of the creating product. The database manager uses this field to signal the existence of database manager-specific data.
<b>IXFHDATE</b>	The date on which the file was written, in the form <i>yyyymmdd</i> .
<b>IXFHTIME</b>	The time at which the file was written, in the form <i>hhmmss</i> . This field is optional and can be left blank.
<b>IXFHHCNT</b>	The number of H, T, and C records in this file that precede the first data record. A records are not included in this count.
<b>IXFHSBCP</b>	Single-byte code page field, containing a single-byte character representation of a SBCS CPGID or '00000'.  The export utility sets this field equal to the SBCS CPGID of the exported database table. For example, if the table SBCS CPGID is 850, this field contains '00850'.
<b>IXFHDBCP</b>	Double-byte code page field, containing a single-byte character representation of a DBCS CPGID or '00000'.  The export utility sets this field equal to the DBCS CPGID of the exported database table. For example, if the table DBCS CPGID is 301, this field contains '00301'.
<b>IXFHFIL1</b>	Spare field set to two blanks to match a reserved field in host IXF files.

TABLE RECORD				
FIELD NAME	LENGTH	TYPE	COMMENTS	
-----	-----	-----	-----	
IXFTRECL	06-BYTE	CHARACTER	record length	
IXFTRECT	01-BYTE	CHARACTER	record type = 'T'	
IXFTNAML	02-BYTE	CHARACTER	name length	
IXFTNAME	18-BYTE	CHARACTER	name of data	
IXFTQUAL	08-BYTE	CHARACTER	qualifier	
IXFTSRC	12-BYTE	CHARACTER	data source	
IXFTDATA	01-BYTE	CHARACTER	data convention = 'C'	
IXFTFORM	01-BYTE	CHARACTER	data format = 'M'	
IXFTMFRM	05-BYTE	CHARACTER	machine format='PC'	
IXFTLOC	01-BYTE	CHARACTER	data location = 'I'	
IXFTCNT	05-BYTE	CHARACTER	'C' record count	
IXFTFIL1	02-BYTE	CHARACTER	reserved	
IXFTDESC	30-BYTE	CHARACTER	data description	

## PC Version of IXF File Format

The following fields are contained in the table record:

<b>IXFTRECL</b>	The record length indicator. A 6-byte character representation of an integer value specifying the length, in bytes, of the portion of the PC/IXF record that follows the record length indicator; that is, the total record size minus 6 bytes. The T record must be sufficiently long to include all of its defined fields.
<b>IXFTRECT</b>	The IXF record type, which is set to T for this record.
<b>IXFTNAML</b>	The length, in bytes, of the table name in the IXFTNAME field.
<b>IXFTNAME</b>	The name of the table. If each file has only one table, this is an informational field only. The database manager does not use this field when importing data. When writing a PC/IXF file, the database manager writes the DOS file name (and possibly path information) to this field.
<b>IXFTQUAL</b>	Table name qualifier, which identifies the creator of a table in a relational system. This is an informational field only. If a program writing a file has no data to write to this field, the preferred fill value is blanks. Programs reading a file may print or display this field, or store it in an informational field, but no computations should depend on the content of this field.
<b>IXFTSRC</b>	Used to indicate the original source of the data. This is an informational field only. If a program writing a file has no data to write to this field, the preferred fill value is blanks. Programs reading a file may print or display this field, or store it in an informational field, but no computations should depend on the content of this field.
<b>IXFTDATA</b>	Convention used to describe the data. This field must be set to C for import and export, indicating that individual column attributes are described in the following column descriptor (C) records, and that data follows PC/IXF conventions.
<b>IXFTFORM</b>	Convention used to store numeric data. This field must be set to M, indicating that numeric data in the data (D) records is stored in the machine (internal) format specified by the IXFTMFRM field.
<b>IXFTMFRM</b>	The format of any machine data in the PC/IXF file. The database manager will only read or write files if this field is set to PCbbb, where b represents a blank, and PC specifies that data in the PC/IXF file is in IBM PC machine format.
<b>IXFTLOC</b>	The location of the data. The database manager only supports a value of I, meaning the data is internal to this file.
<b>IXFTCCNT</b>	The number of C records in this table. It is a right-justified character representation of an integer value.
<b>IXFTFIL1</b>	Spare field set to two blanks to match a reserved field in host IXF files.

## PC Version of IXF File Format

**IXFTDESC** Descriptive data about the table. This is an informational field only. If a program writing a file has no data to write to this field, the preferred fill value is blanks. Programs reading a file may print or display this field, or store it in an informational field, but no computations should depend on the content of this field. This field contains NOT NULL WITH DEFAULT if the column was not null with default, and the table name came from a workstation database.

COLUMN DESCRIPTOR RECORD			
FIELD NAME	LENGTH	TYPE	COMMENTS
-----	-----	-----	-----
IXFCRECL	06-BYTE	CHARACTER	record length
IXFCRECT	01-BYTE	CHARACTER	record type = 'C'
IXFCNAML	02-BYTE	CHARACTER	column name length
IXFCNAME	18-BYTE	CHARACTER	column name
IXFCNULL	01-BYTE	CHARACTER	column allows nulls
IXFCSLCT	01-BYTE	CHARACTER	column selected flag
IXFCKEY	01-BYTE	CHARACTER	key column flag
IXFCCLAS	01-BYTE	CHARACTER	data class
IXFCSTYPE	03-BYTE	CHARACTER	data type
IXFCSBCP	05-BYTE	CHARACTER	single byte code page
IXFCBCP	05-BYTE	CHARACTER	double byte code page
IXFCLENG	05-BYTE	CHARACTER	column data length
IXFCDRID	03-BYTE	CHARACTER	'D' record identifier
IXFCPOSN	06-BYTE	CHARACTER	column position
IXFCDESC	30-BYTE	CHARACTER	column description
IXFCNDIM	02-BYTE	CHARACTER	number of dimensions
IXFCDSIZ	varying	CHARACTER	size of each dimension

The following fields are contained in column descriptor records:

- IXFCRECL** The record length indicator. A 6-byte character representation of an integer value specifying the length, in bytes, of the portion of the PC/IXF record that follows the record length indicator; that is, the total record size minus 6 bytes. The C record must be sufficiently long to include all of its defined fields.
- IXFCRECT** The IXF record type, which is set to C for this record.
- IXFCNAML** The length, in bytes, of the column name in the IXFCNAME field.
- IXFCNAME** The name of the column.
- IXFCNULL** Specifies if nulls are permitted in this column. Valid settings are Y or N.
- IXFCSLCT** An obsolete field whose intended purpose was to allow selection of a subset of columns in the data. Programs writing PC/IXF files should always store a Y in this field. Programs reading PC/IXF files should ignore the field.

## PC Version of IXF File Format

<b>IXFCKEY</b>	<p>The key indicator. If the value of this field is Y, the column is a key column; if the value is N, the column is not a key column. The database manager does not use this field. It ignores the field when importing data, and sets it to N when generating an export file.</p>
<b>IXFCCLAS</b>	<p>The class of data types to be used in the IXFCTYPE field. The database manager only supports relational types (R).</p>
<b>IXFCTYPE</b>	<p>The data type for the column. For more information about data types, see “PC/IXF Data Types” on page 414.</p>
<b>IXFCSBCP</b>	<p>Contains a single-byte character representation of a SBCS CPGID. This field specifies the CPGID for single-byte character data, which occurs with the IXFDCOLS field of the D records for this column.</p> <p>The semantics of this field vary with the data type for the column (specified in the IXFCTYPE field).</p> <ul style="list-style-type: none"><li>• For a character string column, this field should normally contain a nonzero value equal to that of the IXFHSBCP field in the H record; however, other values are permitted. If this value is zero, the column is interpreted to contain bit string data.</li><li>• For a numeric column, this field is not meaningful. It is set to zero by the export utility, and ignored by the import utility.</li><li>• For a date or time column, this field is not meaningful. It is set to the value of the IXFHSBCP field by the export utility, and ignored by the import utility.</li><li>• For a graphic column, this field must be zero.</li></ul> <p>See also Table 13 on page 418.</p>
<b>IXFCDBC</b>	<p>Contains a single-byte character representation of a DBCS CPGID. This field specifies the CPGID for double-byte character data, which occurs with the IXFDCOLS field of the D records for this column.</p> <p>The semantics of this field vary with the data type for the column (specified in the IXFCTYPE field).</p> <ul style="list-style-type: none"><li>• For a character string column, this field should either be zero, or contain a value equal to that of the IXFHDBC field in the H record; however, other values are permitted. If the value in the IXFCSBCP field is zero, the value in this field must be zero.</li><li>• For a numeric column, this field is not meaningful. It is set to zero by the export utility, and ignored by the import utility.</li><li>• For a date or time column, this field is not meaningful. It is set to zero by the export utility, and ignored by the import utility.</li><li>• For a graphic column, this field must have a value equal to the value of the IXFHDBC field.</li></ul> <p>See also Table 13 on page 418.</p>
<b>IXFCLENG</b>	<p>Provides information about the size of the column being described. For some data types, this field is unused, and should contain blanks. For other data types, this field contains the right-justified character</p>

## PC Version of IXF File Format

representation of an integer specifying the column length. For yet other data types, this field is divided into two subfields: 3 bytes for precision, and 2 bytes for scale; both of these subfields are right-justified character representations of integers.

<b>IXFCDRID</b>	The D record identifier. This field contains the right-justified character representation of an integer value. Several D records can be used to contain each row of data in the PC/IXF file. This field specifies which D record (of the several D records contributing to a row of data) contains the data for the column. A value of one (for example, 001) indicates that the data for a column is in the first D record in a row of data. The first C record must have an IXFCDRID value of one. All subsequent C records must have an IXFCDRID value equal to the value in the preceding C record, or one higher.
<b>IXFCPOSN</b>	The value in this field is used to locate the data for the column within one of the D records representing a row of table data. It is the starting position of the data for this column within the IXFCOLS field of the D record. If the column is nullable, IXFCPOSN points to the null indicator; otherwise, it points to the data itself. If a column contains varying length data, the data itself begins with the current length indicator. The IXFCPOSN value for the first byte in the IXFCOLS field of the D record is one (not zero). If a column is in a new D record, the value of IXFCPOSN should be one; otherwise, IXFCPOSN values should increase from column to column to such a degree that the data values do not overlap.
<b>IXFCDESC</b>	<p>Descriptive information about the column. This is an informational field only. If a program writing to a file has no data to write to this field, the preferred fill value is blanks. Programs reading a file may print or display this field, or store it in an informational field, but no computations should depend on the content of this field. If</p> <ul style="list-style-type: none"><li>• the column is not null with default</li><li>• the table resides in a workstation database</li><li>• the select statement on the export is of the form <code>select * from table</code></li></ul> <p>the export utility will put NOT NULL WITH DEFAULT in this field, and when the import utility creates a new table with this file, it will create it as not null with default.</p>
<b>IXFCNDIM</b>	The number of dimensions in the column. Arrays are not supported in this version of PC/IXF. This field must therefore contain a character representation of a zero integer value.
<b>IXFCDSIZ</b>	The size or range of each dimension. The length of this field is five bytes per dimension. Since arrays are not supported (that is, the number of dimensions must be zero), this field has zero length, and does not actually exist.

## PC Version of IXF File Format

DATA RECORD			
FIELD NAME	LENGTH	TYPE	COMMENTS
-----	-----	-----	-----
IXFDRECL	06-BYTE	CHARACTER	record length
IXFDRECT	01-BYTE	CHARACTER	record type = 'D'
IXFDRID	03-BYTE	CHARACTER	'D' record identifier
IXDFIL1	04-BYTE	CHARACTER	reserved
IXFDCOLS	varying	variable	columnar data

The following fields are contained in the data records:

**IXFDRECL** The record length indicator. A 6-byte character representation of an integer value specifying the length, in bytes, of the portion of the PC/IXF record that follows the record length indicator; that is, the total record size minus 6 bytes. Each D record must be sufficiently long to include all significant data for the current occurrence of the last data column stored in the record.

**IXFDRECT** The IXF record type, which is set to D for this record, indicating that it contains data values for the table.

**IXFDRID** The record identifier, which identifies a particular D record within the sequence of several D records contributing to a row of data. For the first D record in a row of data, this field has a value of one; for the second D record in a row of data, this field has a value of two, and so on. In each row of data, all the D record identifiers called out in the C records must actually exist.

**IXDFIL1** Spare field set to four blanks to match reserved fields, and hold a place for a possible shift-out character, in host IXF files.

**IXFDCOLS** The area for columnar data. The data area of a data record (D record) is composed of one or more column entries. There is one column entry for each column descriptor record, which has the same D record identifier as the D record. In the D record, the starting position of the column entries is indicated by the IXFCPOSN value in the C records.

The format of the column entry data depends on whether or not the column is nullable:

- If the column is nullable (the IXFCNULL field is set to Y), the column entry data includes a null indicator. If the column is not null, the indicator is followed by data type-specific information, including the actual database value. The null indicator is a two-byte value set to x'0000' for not null, and x'FFFF' for null.
- If the column is not nullable, the column entry data includes only data type-specific information, including the actual database value.

## PC Version of IXF File Format

For varying-length data types, the data type-specific information includes a current length indicator. The current length indicators are 2-byte integers in a form specified by the IXFTMFRM field.

The length of the data area of a D record may not exceed 32771 bytes.

APPLICATION RECORD			
FIELD NAME	LENGTH	TYPE	COMMENTS
-----	-----	-----	-----
IXFARECL	06-BYTE	CHARACTER	record length
IXFARECT	01-BYTE	CHARACTER	record type = 'A'
IXFAPPID	12-BYTE	CHARACTER	application identifier
IXFADATA	varying	variable	application-specific data

The following fields are contained in application records:

- IXFARECL** The record length indicator. A 6-byte character representation of an integer value specifying the length, in bytes, of the portion of the PC/IXF record that follows the record length indicator; that is, the total record size minus 6 bytes. Each A record must be sufficiently long to include at least the entire IXFAPPID field.
- IXFARECT** The IXF record type, which is set to A for this record, indicating that this is an application record. These records are ignored by programs which do not have particular knowledge about the content and the format of the data implied by the application identifier.
- IXFAPPID** The application identifier, which identifies the application creating the A record. PC/IXF files created by the database manager may have A records with the first 6 characters of this field set to a constant identifying the database manager, and the last 6 characters identifying the release or version of the database manager or another application writing the A record.
- IXFADATA** This field contains application dependent supplemental data, whose form and content are known only to the program creating the A record, and to other applications which are likely to process the A record.

## PC Version of IXF File Format

### PC/IXF Data Types

Name	IXFCTYPE Value	Description
BLOB, CLOB	404, 408	<p>A variable-length character string. The maximum length of the string is contained in the IXFCLENG field of the column descriptor record, and cannot exceed 32 767 bytes. The string itself is preceded by a current length indicator, which is a 4-byte integer specifying the length of the string, in bytes. The string is in the code page indicated by IXFCSBCP.</p> <p>The following applies to BLOBs only: If IXFCSBCP is zero, the string is bit data, and should not be translated by any transformation program.</p> <p>The following applies to CLOBs only: If IXFCDBCP is nonzero, the string can also contain double-byte characters in the code page indicated by IXFCDBCP.</p>
BLOB_FILE, CLOB_FILE, DBCLOB_FILE	804, 808, 812	<p>A fixed-length field containing an SQLFILE structure with the <i>name_length</i> and the <i>name</i> fields filled in. The length of the structure is contained in the IXFCLENG field of the column descriptor record, and cannot exceed 255 bytes. The file name is in the code page indicated by IXFCSBCP. If IXFCDBCP is nonzero, the file name can also contain double-byte characters in the code page indicated by IXFCDBCP. If IXFCSBCP is zero, the file name is bit data and should not be translated by any transformation program.</p> <p>Since the length of the structure is stored in IXFCLENG, the actual length of the original LOB is lost. IXF files with columns of type BLOB_FILE, CLOB_FILE, or DBCLOB_FILE should not be used to recreate the LOB field, since the LOB will be created with a length of <i>sql_lobfile_len</i>.</p>
CHAR	452	<p>A fixed-length character string. The string length is contained in the IXFCLENG field of the column descriptor record, and cannot exceed 254 bytes. The string is in the code page indicated by IXFCSBCP. If IXFCDBCP is nonzero, the string can also contain double-byte characters in the code page indicated by IXFCDBCP. If IXFCSBCP is zero, the string is bit data and should not be translated by any transformation program.</p>



## PC Version of IXF File Format

Table 12 (Page 2 of 5). PC/IXF Data Types

Name	IXFCTYPE Value	Description
DATE	384	A point in time in accordance with the Gregorian calendar. Each date is a 10-byte character string in International Standards Organization (ISO) format: <i>yyyy-mm-dd</i> . The range of the year part is 0001 to 9999. The range of the month part is 01 to 12. The range of the day part is 01 to <i>n</i> , where <i>n</i> depends on the month, using the usual rules for days of the month and leap year. Leading zeros cannot be omitted from any part. IXFLEN is not used, and should contain blanks. Valid characters within DATE are invariant in all PC ASCII code pages; therefore, IXFCSBCP and IXFCDBCP are not significant, and should be zero.
DBCLOB	412	A variable-length string of double-byte characters. The IXFLEN field in the column descriptor record specifies the maximum number of double-byte characters in the string, and cannot exceed 16383. The string itself is preceded by a current length indicator, which is a 4-byte integer specifying the length of the string in double-byte characters (that is, the value of this integer is one half the length of the string, in bytes). The string is in the DBCS code page, as specified by IXFCDBCP in the C record. Since the string consists of double-byte character data only, IXFCSBCP should be zero. There are no surrounding shift-in or shift-out characters.
DECIMAL	484	A packed decimal number with precision P (as specified by the first three bytes of IXFLEN in the column descriptor record) and scale S (as specified by the last two bytes of IXFLEN). The length, in bytes, of a packed decimal number is $(P+2)/2$ . The precision must be an odd number between 1 and 31, inclusive. The packed decimal number is in the internal format specified by IXFTMFRM, where packed decimal for the PC is defined to be the same as packed decimal for the System/370. IXFCSBCP and IXFCDBCP are not significant, and should be zero.

## PC Version of IXF File Format

Table 12 (Page 3 of 5). PC/IXF Data Types

Name	IXFCTYPE Value	Description
FLOATING POINT	480	Either a long (8-byte) or short (4-byte) floating point number, depending on whether IXFLENG is set to eight or to four. The data is in the internal machine form, as specified by IXFTMFRM. IXFCSBCP and IXFCDBCP are not significant, and should be zero. Four-byte floating point is not supported by the database manager.
GRAPHIC	468	A fixed-length string of double-byte characters. The IXFLENG field in the column descriptor record specifies the number of double-byte characters in the string, and cannot exceed 127. The actual length of the string is twice the value of the IXFLENG field, in bytes. The string is in the DBCS code page, as specified by IXFCDBCP in the C record. Since the string consists of double-byte character data only, IXFCSBCP should be zero. There are no surrounding shift-in or shift-out characters.
INTEGER	496	A 4-byte integer in the form specified by IXFTMFRM. It represents a whole number between -2 147 483 648 and +2 147 483 647. IXFCSBCP and IXFCDBCP are not significant, and should be zero. IXFLENG is not used, and should contain blanks.
LONGVARCHAR	456	A variable-length character string. The maximum length of the string is contained in the IXFLENG field of the column descriptor record, and cannot exceed 32 767 bytes. The string itself is preceded by a current length indicator, which is a 2-byte integer specifying the length of the string, in bytes. The string is in the code page indicated by IXFCSBCP. If IXFCDBCP is nonzero, the string can also contain double-byte characters in the code page indicated by IXFCDBCP. If IXFCSBCP is zero, the string is bit data and should not be translated by any transformation program.

## PC Version of IXF File Format

Table 12 (Page 4 of 5). PC/IXF Data Types

Name	IXFCTYPE Value	Description
LONG VARGRAPHIC	472	A variable-length string of double-byte characters. The IXFCLENG field in the column descriptor record specifies the maximum number of double-byte characters for the string, and cannot exceed 16383. The string itself is preceded by a current length indicator, which is a 2-byte integer specifying the length of the string in double-byte characters (that is, the value of this integer is one half the length of the string, in bytes). The string is in the DBCS code page, as specified by IXFCDBCP in the C record. Since the string consists of double-byte character data only, IXFCSBCP should be zero. There are no surrounding shift-in or shift-out characters.
SMALLINT	500	A 2-byte integer in the form specified by IXFTMFRM. It represents a whole number between -32 768 and +32 767. IXFCSBCP and IXFCDBCP are not significant, and should be zero. IXFCLENG is not used, and should contain blanks.
TIME	388	A point in time in accordance with the 24-hour clock. Each time is an 8-byte character string in ISO format: <i>hh.mm.ss</i> . The range of the hour part is 00 to 24, and the range of the other parts is 00 to 59. If the hour is 24, the other parts are 00. The smallest time is 00.00.00, and the largest is 24.00.00. Leading zeros cannot be omitted from any part. IXFCLENG is not used, and should contain blanks. Valid characters within TIME are invariant in all PC ASCII code pages; therefore, IXFCSBCP and IXFCDBCP are not significant, and should be zero.
TIMESTAMP	392	The date and time with microsecond precision. Each time stamp is a character string of the form <i>yyyy-mm-dd-hh.mm.ss.nnnnnn</i> (year month day hour minutes seconds microseconds). IXFCLENG is not used, and should contain blanks. Valid characters within TIMESTAMP are invariant in all PC ASCII code pages; therefore, IXFCSBCP and IXFCDBCP are not significant, and should be zero.

## PC Version of IXF File Format

*Table 12 (Page 5 of 5). PC/IXF Data Types*

Name	IXFCTYPE Value	Description
VARCHAR	448	A variable-length character string. The maximum length of the string, in bytes, is contained in the IXFLENG field of the column descriptor record, and cannot exceed 254 bytes. The string itself is preceded by a current length indicator, which is a two-byte integer specifying the length of the string, in bytes. The string is in the code page indicated by IXFCSBCP. If IXFCDBCP is nonzero, the string can also contain double-byte characters in the code page indicated by IXFCDBCP. If IXFCSBCP is zero, the string is bit data and should not be translated by any transformation program.
VARGRAPHIC	464	A variable-length string of double-byte characters. The IXFLENG field in the column descriptor record specifies the maximum number of double-byte characters in the string, and cannot exceed 127. The string itself is preceded by a current length indicator, which is a 2-byte integer specifying the length of the string in double-byte characters (that is, the value of this integer is one half the length of the string, in bytes). The string is in the DBCS code page, as specified by IXFCDBCP in the C record. Since the string consists of double-byte character data only, IXFCSBCP should be zero. There are no surrounding shift-in or shift-out characters.

Not all combinations of IXFCSBCP and IXFCDBCP values for PC/IXF character or graphic columns are valid. A PC/IXF character or graphic column with an invalid (IXFCSBCP,IXFCDBCP) combination is an invalid data type.

*Table 13 (Page 1 of 2). Valid PC/IXF Data Types*

PC/IXF Data Type	Valid (IXFCSBCP,IXFCDBCP) Pairs	Invalid (IXFCSBCP,IXFCDBCP) Pairs
CHAR, VARCHAR, or LONG VARCHAR	(0,0), (x,0), or (x,y)	(0,y)
BLOB	(0,0)	(x,0), (0,y), or (x,y)
CLOB	(x,0), (x,y)	(0,0), (0,y)

## PC Version of IXF File Format

*Table 13 (Page 2 of 2). Valid PC/IXF Data Types*

PC/IXF Data Type	Valid (IXFCSBCP,IXFCDBCP) Pairs	Invalid (IXFCSBCP,IXFCDBCP) Pairs
GRAPHIC, VARGRAPHIC, LONG VARGRAPHIC, or DBCLOB	(0,y)	(0,0), (x,0), or (x,y)
<b>Note:</b> x and y are not 0.		

### PC/IXF Data Type Descriptions

*Table 14 (Page 1 of 6). Acceptable Data Type Forms for the PC/IXF File Format*

Data Type	Form in Files Created by the Export Utility	Form Acceptable to the Import Utility
BLOB	A PC/IXF BLOB column is created. The maximum length of the database column, the SBCS CPGID value, and the DBCS CPGID value are copied to the column descriptor record.	A PC/IXF CHAR, VARCHAR, LONG VARCHAR, BLOB, or BLOB_FILE column is acceptable if: <ul style="list-style-type: none"> <li>The database column is marked FOR BIT DATA</li> <li>The PC/IXF column single-byte code page value equals the SBCS CPGID of the database column, and the PC/IXF column double-byte code page value equals zero, or the DBCS CPGID of the database column. A PC/IXF GRAPHIC, VARGRAPHIC, or LONG VARGRAPHIC BLOB column is also acceptable. If the PC/IXF column is of fixed length, its length must be compatible with the maximum length of the database column. See also the "FORCEIN Option" on page 429.</li> </ul>

## PC Version of IXF File Format

Table 14 (Page 2 of 6). Acceptable Data Type Forms for the PC/IXF File Format

Data Type	Form in Files Created by the Export Utility	Form Acceptable to the Import Utility
CHAR	A PC/IXF CHAR column is created. The database column length, the SBCS CPGID value, and the DBCS CPGID value are copied to the PC/IXF column descriptor record.	<p>A PC/IXF CHAR, VARCHAR, or LONG VARCHAR column is acceptable if:</p> <ul style="list-style-type: none"> <li>• The database column is marked FOR BIT DATA</li> <li>• The PC/IXF column single-byte code page value equals the SBCS CPGID of the database column, and the PC/IXF column double-byte code page value equals zero, or the DBCS CPGID of the database column.</li> </ul> <p>A PC/IXF GRAPHIC, VARGRAPHIC, or LONG VARGRAPHIC column is also acceptable if the database column is marked FOR BIT DATA. In any case, if the PC/IXF column is of fixed length, its length must be compatible with the length of the database column. The data is padded on the right with single-byte spaces (x'20'), if necessary. See also the "FORCEIN Option" on page 429.</p>
CLOB	A PC/IXF CLOB column is created. The maximum length of the database column, the SBCS CPGID value, and the DBCS CPGID value are copied to the column descriptor record.	A PC/IXF CHAR, VARCHAR, LONG VARCHAR, CLOB, or CLOB_FILE column is acceptable if the PC/IXF column single-byte code page value equals the SBCS CPGID of the database column, and the PC/IXF column double-byte code page value equals zero, or the DBCS CPGID of the database column. If the PC/IXF column is of fixed length, its length must be compatible with the maximum length of the database column. See also the "FORCEIN Option" on page 429.
DATE	A DATE column, identical to the database column, is created.	A PC/IXF column of type DATE is the usual input. The import utility also attempts to accept columns in any of the character types, except those with incompatible lengths. The character column in the PC/IXF file must contain dates in a format consistent with the country code of the target database.

## PC Version of IXF File Format

Table 14 (Page 3 of 6). Acceptable Data Type Forms for the PC/IXF File Format

Data Type	Form in Files Created by the Export Utility	Form Acceptable to the Import Utility
DBCLOB	A PC/IXF DBCLOB column is created. The maximum length of the database column, the SBCS CPGID value, and the DBCS CPGID value are copied to the column descriptor record.	A PC/IXF GRAPHIC, VARGRAPHIC, LONG VARGRAPHIC, DBCLOB, or DBCLOB_FILE column is acceptable if the PC/IXF column double-byte code page value equals that of the database column. If the PC/IXF column is of fixed length, its length must be compatible with the maximum length of the database column. See also the "FORCEIN Option" on page 429.
DECIMAL	A DECIMAL column, identical to the database column, is created. The precision and scale of the column is stored in the column descriptor record.	A column in any numeric type (SMALLINT, INTEGER, DECIMAL, or FLOAT) is accepted. Individual values are rejected if they are not in the range of the DECIMAL column into which they are being imported.
FLOAT	A FLOAT column, identical to the database column, is created.	A column in any numeric type (SMALLINT, INTEGER, DECIMAL, or FLOAT) is accepted. All values are within range.
GRAPHIC (DBCS only)	A PC/IXF GRAPHIC column is created. The database column length, the SBCS CPGID value, and the DBCS CPGID value are copied to the column descriptor record.	A PC/IXF GRAPHIC, VARGRAPHIC, or LONG VARGRAPHIC column is acceptable if the PC/IXF column double-byte code page value equals that of the database column. If the PC/IXF column is of fixed length, its length must be compatible with the database column length. The data is padded on the right with double-byte spaces (x'8140'), if necessary. See also the "FORCEIN Option" on page 429.
INTEGER	An INTEGER column, identical to the database column, is created.	A column in any numeric type (SMALLINT, INTEGER, DECIMAL, or FLOAT) is accepted. Individual values are rejected if they are not in the range -2 147 483 648 to 2 147 483 647.

## PC Version of IXF File Format

Table 14 (Page 4 of 6). Acceptable Data Type Forms for the PC/IXF File Format

Data Type	Form in Files Created by the Export Utility	Form Acceptable to the Import Utility
LONG VARCHAR	A PC/IXF LONG VARCHAR column is created. The maximum length of the database column, the SBCS CPGID value, and the DBCS CPGID value are copied to the column descriptor record.	<p>A PC/IXF CHAR, VARCHAR, or LONG VARCHAR column is acceptable if:</p> <ul style="list-style-type: none"> <li>• The database column is marked FOR BIT DATA</li> <li>• The PC/IXF column single-byte code page value equals the SBCS CPGID of the database column, and the PC/IXF column double-byte code page value equals zero, or the DBCS CPGID of the database column.</li> </ul> <p>A PC/IXF GRAPHIC, VARGRAPHIC, or LONG VARGRAPHIC column is also acceptable if the database column is marked FOR BIT DATA. In any case, if the PC/IXF column is of fixed length, its length must be compatible with the maximum length of the database column. See also the "FORCEIN Option" on page 429.</p>
LONG VARGRAPHIC (DBCS only)	A PC/IXF LONG VARGRAPHIC column is created. The maximum length of the database column, the SBCS CPGID value, and the DBCS CPGID value are copied to the column descriptor record.	A PC/IXF GRAPHIC, VARGRAPHIC, or LONG VARGRAPHIC column is acceptable if the PC/IXF column double-byte code page value equals that of the database column. If the PC/IXF column is of fixed length, its length must be compatible with the maximum length of the database column. See also the "FORCEIN Option" on page 429.
SMALLINT	A SMALLINT column, identical to the database column, is created.	A column in any numeric type (SMALLINT, INTEGER, DECIMAL, or FLOAT) is accepted. Individual values are rejected if they are not in the range -32768 to 32767.
TIME	A TIME column, identical to the database column, is created.	A PC/IXF column of type TIME is the usual input. The import utility also attempts to accept columns in any of the character types, except those with incompatible lengths. The character column in the PC/IXF file must contain time data in a format consistent with the country code of the target database.



## PC Version of IXF File Format

Table 14 (Page 5 of 6). Acceptable Data Type Forms for the PC/IXF File Format

Data Type	Form in Files Created by the Export Utility	Form Acceptable to the Import Utility
TIMESTAMP	A TIMESTAMP column, identical to the database column, is created.	A PC/IXF column of type TIMESTAMP is the usual input. The import utility also attempts to accept columns in any of the character types, except those with incompatible lengths. The character column in the PC/IXF file must contain data in the input format for time stamps.
VARCHAR	If the maximum length of the database column is $\leq 254$ , a PC/IXF VARCHAR column is created. If the maximum length of the database column is $> 254$ , a PC/IXF LONG VARCHAR column is created. The maximum length of the database column, the SBCS CPGID value, and the DBCS CPGID value are copied to the column descriptor record.	<p>A PC/IXF CHAR, VARCHAR, or LONG VARCHAR column is acceptable if:</p> <ul style="list-style-type: none"> <li>• The database column is marked FOR BIT DATA</li> <li>• The PC/IXF column single-byte code page value equals the SBCS CPGID of the database column, and the PC/IXF column double-byte code page value equals zero, or the DBCS CPGID of the database column.</li> </ul> <p>A PC/IXF GRAPHIC, VARGRAPHIC, or LONG VARGRAPHIC column is also acceptable if the database column is marked FOR BIT DATA. In any case, if the PC/IXF column is of fixed length, its length must be compatible with the maximum length of the database column. See also the "FORCEIN Option" on page 429.</p>

## PC Version of IXF File Format

Table 14 (Page 6 of 6). Acceptable Data Type Forms for the PC/IXF File Format

Data Type	Form in Files Created by the Export Utility	Form Acceptable to the Import Utility
VARGRAPHIC (DBCS only)	<p>If the maximum length of the database column is <math>\leq 127</math>, a PC/IXF VARGRAPHIC column is created.</p> <p>If the maximum length of the database column is <math>&gt; 127</math>, a PC/IXF LONG VARGRAPHIC column is created. The maximum length of the database column, the SBCS CPGID value, and the DBCS CPGID value are copied to the column descriptor record.</p>	<p>A PC/IXF GRAPHIC, VARGRAPHIC, or LONG VARGRAPHIC column is acceptable if the PC/IXF column double-byte code page value equals that of the database column. If the PC/IXF column is of fixed length, its length must be compatible with the maximum length of the database column. See also the "FORCEIN Option" on page 429.</p>

### General Rules Governing PC/IXF File Import into Databases

The database manager import utility applies the following general rules when importing a PC/IXF file in either an SBCS or a DBCS environment:

- The import utility accepts PC/IXF format files only (IXFHID = 'IXF'). IXF files of other formats cannot be imported.
- The import utility rejects a PC/IXF file with more than 1024 columns.
- The value of IXFHSBCP in the PC/IXF H record must equal the SBCS CPGID, or there must be a conversion table between the IXFHSBCP/IXFHDBCP and the SBCS/DBCS CPGID of the target database. The value of IXFHDBCP must equal either '00000', or the DBCS CPGID of the target database. If either of these conditions is not satisfied, the import utility rejects the PC/IXF file, unless the "FORCEIN Option" on page 429 is specified.
- Invalid Data Types — New Table
 

Import of a PC/IXF file into a *new* table is specified by the CREATE or the REPLACE\_CREATE keywords in the IMPORT command. If a PC/IXF column of an invalid data type (valid data types are defined in "PC/IXF Data Types" on page 414) is selected for import into a new table, the import utility terminates. The entire PC/IXF file is rejected, no table is created, and no data is imported.
- Invalid Data Types — Existing Table

## PC Version of IXF File Format

Import of a PC/IXF file into an *existing* table is specified by the INSERT, the INSERT\_UPDATE, or the REPLACE\_CREATE keywords in the IMPORT command. If a PC/IXF column of an invalid data type is selected for import into an existing table, one of two actions is possible:

- If the target table column is nullable, all values for the invalid PC/IXF column are ignored, and the table column values are set to NULL
  - If the target table column is not nullable, the import utility terminates. The entire PC/IXF file is rejected, and no data is imported. The existing table remains unaltered.
- When importing into a new table, nullable PC/IXF columns generate nullable database columns, and not nullable PC/IXF columns generate not nullable database columns.
  - A not nullable PC/IXF column can be imported into a nullable database column.
  - A nullable PC/IXF column can be imported into a not nullable database column. If a NULL value is encountered in the PC/IXF column, the import utility rejects the values of all columns in the PC/IXF row that contains the NULL value (the entire row is rejected), and processing continues with the next PC/IXF row. That is, no data is imported from a PC/IXF row that contains a NULL value if a target table column (for the NULL) is not nullable.

- Incompatible Columns — New Table

If, during import to a *new* database table, a PC/IXF column is selected that is incompatible with the target database column, the import utility terminates. The entire PC/IXF file is rejected, no table is created, and no data is imported.

**Note:** The IMPORT “FORCEIN Option” on page 429 extends the scope of compatible columns.

- Incompatible Columns — Existing Table

If, during import to an *existing* database table, a PC/IXF column is selected that is incompatible with the target database column, one of two actions is possible:

- If the target table column is nullable, all values for the PC/IXF column are ignored, and the table column values are set to NULL
- If the target table column is not nullable, the import utility terminates. The entire PC/IXF file is rejected, and no data is imported. The existing table remains unaltered.

**Note:** The IMPORT “FORCEIN Option” on page 429 extends the scope of compatible columns.

- Invalid Values

If, during import, a PC/IXF column value is encountered that is not valid for the target database column, the import utility rejects the values of all columns in the PC/IXF row that contains the invalid value (the entire row is rejected), and processing continues with the next PC/IXF row.

## PC Version of IXF File Format

### Data Type-Specific Rules Governing PC/IXF File Import into Databases

- A valid PC/IXF numeric column can be imported into any compatible numeric database column. PC/IXF columns containing 4-byte floating point data are not imported, because this is an invalid data type.
- Database date/time columns can accept values from matching PC/IXF date/time columns (DATE, TIME, and TIMESTAMP), as well as from PC/IXF character columns (CHAR, VARCHAR, and LONG VARCHAR), subject to column length and value compatibility restrictions.
- A valid PC/IXF character column (CHAR, VARCHAR, or LONG VARCHAR) can always be imported into an *existing* database character column marked FOR BIT DATA; otherwise:
  - IXFCSBCP and the SBCS CPGID must agree
  - There must be a conversion table for the IXFCSBCP/IXFCDBCP and the SBCS/DBCS
  - One set must be all zeros (FOR BIT DATA).

If IXFCSBCP is not zero, the value of IXFCDBCP must equal either zero or the DBCS CPGID of the target database column.

If either of these conditions is not satisfied, the PC/IXF and database columns are incompatible.

When importing a valid PC/IXF character column into a *new* database table, the value of IXFCSBCP must equal either zero or the SBCS CPGID of the database, or there must be a conversion table. If IXFCSBCP is zero, IXFCDBCP must also be zero (otherwise the PC/IXF column is an invalid data type); IMPORT creates a character column marked FOR BIT DATA in the new table. If IXFCSBCP is not zero, and equals the SBCS CPGID of the database, the value of IXFCDBCP must equal either zero or the DBCS CPGID of the database; in this case, the utility creates a character column in the new table with SBCS and DBCS CPGID values equal to those of the database. If these conditions are not satisfied, the PC/IXF and database columns are incompatible.

The “FORCEIN Option” on page 429 can be used to override code page equality checks. However, a PC/IXF character column with IXFCSBCP equal to zero and IXFCDBCP not equal to zero is an invalid data type, and cannot be imported, even if FORCEIN is specified.

- A valid PC/IXF graphic column (GRAPHIC, VARGRAPHIC, or LONG VARGRAPHIC) can always be imported into an *existing* database character column marked FOR BIT DATA, but is incompatible with all other database columns. The “FORCEIN Option” on page 429 can be used to relax this restriction. However, a PC/IXF graphic column with IXFCSBCP not equal to zero, or IXFCDBCP equal to zero, is an invalid data type, and cannot be imported, even if FORCEIN is specified.

When importing a valid PC/IXF graphic column into a database graphic column, the value of IXFCDBCP must equal the DBCS CPGID of the target database column (that is, the double-byte code pages of the two columns must agree).

## PC Version of IXF File Format

- If, during import of a PC/IXF file into an existing database table, a fixed-length string column (CHAR or GRAPHIC) is selected whose length is greater than the maximum length of the target column, the columns are incompatible.
- If, during import of a PC/IXF file into an existing database table, a variable-length string column (VARCHAR, LONG VARCHAR, VARGRAPHIC, or LONG VARGRAPHIC) is selected whose length is greater than the maximum length of the target column, the columns *are* compatible. Individual values are processed according to the compatibility rules governing the database manager INSERT statement, and PC/IXF values which are too long for the target database column are invalid.
- PC/IXF values imported into a fixed-length database *character* column (that is, a CHAR column) are padded on the right with single-byte spaces (0x20), if necessary, to obtain values whose length equals that of the database column. PC/IXF values imported into a fixed-length database *graphic* column (that is, a GRAPHIC column) are padded on the right with double-byte spaces (0x8140), if necessary, to obtain values whose length equals that of the database column.
- Since PC/IXF VARCHAR columns have a maximum length of 254 bytes, a database VARCHAR column of maximum length  $n$ , with  $254 < n < 4001$ , must be exported into a PC/IXF LONG VARCHAR column of maximum length  $n$ .
- Although PC/IXF LONG VARCHAR columns have a maximum length of 32 767 bytes, and database LONG VARCHAR columns have a maximum length restriction of 32 700 bytes, PC/IXF LONG VARCHAR columns of length greater than 32 700 bytes (but less than 32 768 bytes) are still valid, and can be imported into database LONG VARCHAR columns, but data may be lost.
- Since PC/IXF VARGRAPHIC columns have a maximum length of 127 bytes, a database VARGRAPHIC column of maximum length  $n$ , with  $127 < n < 2001$ , must be exported into a PC/IXF LONG VARGRAPHIC column of maximum length  $n$ .
- Although PC/IXF LONG VARGRAPHIC columns have a maximum length of 16 383 bytes, and database LONG VARGRAPHIC columns have a maximum length restriction of 16 350, PC/IXF LONG VARGRAPHIC columns of length greater than 16 350 bytes (but less than 16 384 bytes) are still valid, and can be imported into database LONG VARGRAPHIC columns, but data may be lost.

Table 15 summarizes PC/IXF file import into new or existing database tables without the FORCEIN option.

Table 15 (Page 1 of 2). Summary of PC/IXF File Import without FORCEIN Option											
PC/IXF COLUMN DATA TYPE	DATABASE COLUMN DATA TYPE										
	NUMERIC				CHARACTER			GRAPH	DATETIME		
	SMALL INT	INT	DEC	FLT	(0,0)	(SBCS, 0) <sup>d</sup>	(SBCS, DBCS) <sup>b</sup>	b	DATE	TIME	TIME STAMP
Numeric											
-SMALLINT	N										
	E	E	E <sup>a</sup>	E							

## PC Version of IXF File Format

Table 15 (Page 2 of 2). Summary of PC/IXF File Import without FORCEIN Option

PC/IXF COLUMN DATA TYPE	DATABASE COLUMN DATA TYPE										
	NUMERIC				CHARACTER			GRAPH	DATETIME		
	SMALL INT	INT	DEC	FLT	(0,0)	(SBCS, 0) <sup>d</sup>	(SBCS, DBCS) <sup>b</sup>	b	DATE	TIME	TIME STAMP
-INTEGER		N									
	E <sup>a</sup>	E	E <sup>a</sup>	E							
-DECIMAL			N								
	E <sup>a</sup>	E <sup>a</sup>	E <sup>a</sup>	E							
-FLOAT				N							
	E <sup>a</sup>	E <sup>a</sup>	E <sup>a</sup>	E							
Character											
-(0,0)					N						
					E				E <sup>c</sup>	E <sup>c</sup>	E <sup>c</sup>
-(SBCS,0)						N	N				
					E	E	E		E <sup>c</sup>	E <sup>c</sup>	E <sup>c</sup>
-(SBCS, DBCS)							N		E <sup>c</sup>	E <sup>c</sup>	E <sup>c</sup>
					E		E				
Graphic											
								N			
					E			E			
Datetime											
-DATE									N		
									E		
-TIME										N	
										E	
-TIME STAMP											N
											E
<b>Notes:</b>											
<ol style="list-style-type: none"> <li>The table is a matrix of all valid PC/IXF and database manager data types. If a PC/IXF column can be imported into a database column, a letter is displayed in the matrix cell at the intersection of the PC/IXF data type matrix row and the database manager data type matrix column. An 'N' indicates that the utility is creating a new database table (a database column of the indicated data type is created). An 'E' indicates that the utility is importing data to an existing database table (a database column of the indicated data type is a valid target).</li> <li>Character string data types are distinguished by code page attributes. These attributes are shown as an ordered pair (SBCS,DBCS), where: <ul style="list-style-type: none"> <li>SBCS is either zero or denotes a nonzero value of the single-byte code page attribute of the character data type</li> <li>DBCS is either zero or denotes a nonzero value of the double-byte code page attribute of the character data type.</li> </ul> </li> <li>If the table indicates that a PC/IXF character column can be imported into a database character column, the values of their respective code page attribute pairs satisfy the rules governing code page equality.</li> </ol>											
<sup>a</sup> Individual values are rejected if they are out of range for the target numeric data type.											
<sup>b</sup> Data type is available only in DBCS environments.											
<sup>c</sup> Individual values are rejected if they are not valid date or time values.											
<sup>d</sup> Data type is not available in DBCS environments.											

### FORCEIN Option

The FORCEIN option permits import of a PC/IXF file despite code page differences between data in the PC/IXF file and the target database. It offers additional flexibility in the definition of compatible columns.

### FORCEIN General Semantics

The following general semantics apply when using the FORCEIN option in either an SBCS or a DBCS environment:

- The FORCEIN option should be used with caution. It is usually advisable to attempt an import without this option enabled. However, because of the generic nature of the PC/IXF data interchange architecture, some PC/IXF files may contain data types or values that cannot be imported without intervention.
- Import with FORCEIN to a *new* table may yield a different result than import to an existing table. An existing table has predefined target data types for each PC/IXF data type.
- When LOB data is exported with the LOBSINFILE option, and the files move to another client with a different codepage, then, unlike other data, the CLOBs and DBCLOBs in the separate files are not converted to the client codepage when imported or loaded into a database.

### FORCEIN Code Page Semantics

The following code page semantics apply when using the FORCEIN option in either an SBCS or a DBCS environment:

- The FORCEIN option disables all import utility code page comparisons.  

This rule applies to code page comparisons at the column level and at the file level as well, when importing to a new or an existing database table. At the column (for example, data type) level, this rule applies only to the following database manager and PC/IXF data types: character (CHAR, VARCHAR, and LONG VARCHAR), and graphic (GRAPHIC, VARGRAPHIC, and LONG VARGRAPHIC). The restriction follows from the fact that code page attributes of other data types are not relevant to the interpretation of data type values.
- The FORCEIN option does not disable inspection of code page attributes to determine data types.  

For example, the database manager allows a CHAR column to be declared with the FOR BIT DATA attribute. Such a declaration sets both the SBCS CPGID and the DBCS CPGID of the column to zero; it is the zero value of these CPGIDs that identifies the column values as bit strings (rather than character strings).
- The FORCEIN option does not imply code page translation.  

Values of data types that are sensitive to the FORCEIN option are copied "as is". No code point mappings are employed to account for a change of code page environments. Padding of the imported value with spaces may be necessary in the case of fixed length target columns.
- When data is imported to an *existing* table using the FORCEIN option:

## PC Version of IXF File Format

- The code page value of the target database table and columns always prevails.
- The code page value of the PC/IXF file and columns is ignored.

This rule applies whether or not the FORCEIN option is used. The database manager does not permit changes to a database or a column code page value once a database is created.

- When importing to a *new* table using the FORCEIN option:
  - The code page value of the target database prevails.
  - PC/IXF character columns with IXFCSBCP = IXFCDBCP = 0 generate table columns marked FOR BIT DATA.
  - All other PC/IXF character columns generate table character columns with SBCS and DBCS CPGID values equal to those of the database.
  - PC/IXF graphic columns generate table graphic columns with an SBCS CPGID of "undefined", and a DBCS CPGID equal to that of the database (DBCS environment only).

### FORCEIN Example

Consider a PC/IXF CHAR column with IXFCSBCP = '00897' and IXFCDBCP = '00301'. This column is to be imported into a database CHAR column whose SBCS CPGID = '00850' and DBCS CPGID = '00000'. Without FORCEIN, the utility terminates, and no data is imported, or the PC/IXF column values are ignored, and the database column contains NULLs (if the database column is nullable). With FORCEIN, the utility proceeds, ignoring code page incompatibilities. If there are no other data type incompatibilities (such as length, for example), the values of the PC/IXF column are imported "as is", and become available for interpretation under the database column code page environment.

The following table shows:

- The code page attributes of a column created in a *new* database table when a PC/IXF file data type with specified code page attributes is imported
- That the import utility rejects PC/IXF data types if they invalid or incompatible.

<i>Table 16 (Page 1 of 2). Summary of Import Utility Code Page Semantics (New Table). This table assumes there is no conversion table between a and x. If there were, items 3 and 4 would work successfully without the FORCEIN option.</i>		
CODE PAGE ATTRIBUTES of PC/IXF DATA TYPE	CODE PAGE ATTRIBUTES OF DATABASE TABLE COLUMN	
	Without FORCEIN	With FORCEIN
SBCS		
(0,0)	(0,0)	(0,0)
(a,0)	(a,0)	(a,0)
(x,0)	reject	(a,0)
(x,y)	reject	(a,0)



## PC Version of IXF File Format

*Table 16 (Page 2 of 2). Summary of Import Utility Code Page Semantics (New Table). This table assumes there is no conversion table between a and x. If there were, items 3 and 4 would work successfully without the FORCEIN option.*

CODE PAGE ATTRIBUTES of PC/IXF DATA TYPE	CODE PAGE ATTRIBUTES OF DATABASE TABLE COLUMN	
	Without FORCEIN	With FORCEIN
(a,y)	reject	(a,0)
(0,y)	reject	(0,0)
DBCS		
(0,0)	(0,0)	(0,0)
(a,0)	(a,b)	(a,b)
(x,0)	reject	(a,b)
(a,b)	(a,b)	(a,b)
(x,y)	reject	(a,b)
(a,y)	reject	(a,b)
(x,b)	reject	(a,b)
(0,b)	(-,b)	(-,b)
(0,y)	reject	(-,b)

**Notes:**

1. Code page attributes of a PC/IXF data type are shown as an ordered pair, where x represents a nonzero single-byte code page value, and y represents a nonzero double-byte code page value. A '-' represents an undefined code page value.
2. The use of different letters in various code page attribute pairs is deliberate. Different letters imply different values. For example, if a PC/IXF data type is shown as (x,y), and the database column as (a,y), x does not equal a, but the PC/IXF file and the database have the same double-byte code page value y.
3. Only character and graphic data types are affected by the FORCEIN code page semantics.
4. It is assumed that the database containing the new table has code page attributes of (a,0); therefore, all character columns in the new table must have code page attributes of either (0,0) or (a,0).  
  
In a DBCS environment, it is assumed that the database containing the new table has code page attributes of (a,b); therefore, all graphic columns in the new table must have code page attributes of (-,b), and all character columns must have code page attributes of (a,b). The SBCS CPGID is shown as '-', because it is undefined for graphic data types.
5. The data type of the result is determined by the rules described in "FORCEIN Data Type Semantics" on page 433.
6. The reject result is a reflection of the rules for invalid or incompatible data types (see "General Rules Governing PC/IXF File Import into Databases" on page 424).

## PC Version of IXF File Format

The following table shows:

- That the import utility accepts PC/IXF data types with various code page attributes into an *existing* table column (the *target* column) having the specified code page attributes
- That the import utility does not permit a PC/IXF data type with certain code page attributes to be imported into an *existing* table column having the code page attributes shown. The utility rejects PC/IXF data types if they are invalid or incompatible.

Table 17 (Page 1 of 2). Summary of Import Utility Code Page Semantics (Existing Table). This table assumes there is no conversion table between a and x.

CODE PAGE ATTRIBUTES OF PC/IXF DATA TYPE	CODE PAGE ATTRIBUTES OF TARGET DATABASE COLUMN	RESULTS OF IMPORT	
		Without FORCEIN	With FORCEIN
SBCS			
(0,0)	(0,0)	accept	accept
(a,0)	(0,0)	accept	accept
(x,0)	(0,0)	accept	accept
(x,y)	(0,0)	accept	accept
(a,y)	(0,0)	accept	accept
(0,y)	(0,0)	accept	accept
(0,0)	(a,0)	null or reject	accept
(a,0)	(a,0)	accept	accept
(x,0)	(a,0)	null or reject	accept
(x,y)	(a,0)	null or reject	accept
(a,y)	(a,0)	null or reject	accept
(0,y)	(a,0)	null or reject	null or reject
DBCS			
(0,0)	(0,0)	accept	accept
(a,0)	(0,0)	accept	accept
(x,0)	(0,0)	accept	accept
(a,b)	(0,0)	accept	accept
(x,y)	(0,0)	accept	accept
(a,y)	(0,0)	accept	accept
(x,b)	(0,0)	accept	accept
(0,b)	(0,0)	accept	accept
(0,y)	(0,0)	accept	accept

## PC Version of IXF File Format

Table 17 (Page 2 of 2). Summary of Import Utility Code Page Semantics (Existing Table). This table assumes there is no conversion table between a and x.

CODE PAGE ATTRIBUTES OF PC/IXF DATA TYPE	CODE PAGE ATTRIBUTES OF TARGET DATABASE COLUMN	RESULTS OF IMPORT	
		Without FORCEIN	With FORCEIN
(0,0)	(a,b)	null or reject	accept
(a,0)	(a,b)	accept	accept
(x,0)	(a,b)	null or reject	accept
(a,b)	(a,b)	accept	accept
(x,y)	(a,b)	null or reject	accept
(a,y)	(a,b)	null or reject	accept
(x,b)	(a,b)	null or reject	accept
(0,b)	(a,b)	null or reject	null or reject
(0,y)	(a,b)	null or reject	null or reject
(0,0)	(-,b)	null or reject	accept
(a,0)	(-,b)	null or reject	null or reject
(x,0)	(-,b)	null or reject	null or reject
(a,b)	(-,b)	null or reject	null or reject
(x,y)	(-,b)	null or reject	null or reject
(a,y)	(-,b)	null or reject	null or reject
(x,b)	(-,b)	null or reject	null or reject
(0,b)	(-,b)	accept	accept
(0,y)	(-,b)	null or reject	accept

**Notes:**

1. See the notes for Table 16 on page 430.
2. The null or reject result is a reflection of the rules for invalid or incompatible data types (see "General Rules Governing PC/IXF File Import into Databases" on page 424).

### FORCEIN Data Type Semantics

The FORCEIN option permits import of certain PC/IXF columns into target database columns of unequal and otherwise incompatible data types. The following data type semantics apply when using the FORCEIN option in either an SBCS or a DBCS environment (except where noted):

- In SBCS environments, the FORCEIN option permits import of:
  - A PC/IXF BIT data type (IXFCSBCP = 0 = IXFCDBCP for a PC/IXF character column) into a database character column (nonzero SBCS CPGID, and DBCS CPGID = 0); existing tables only

## PC Version of IXF File Format

- A PC/IXF MIXED data type (nonzero IXFCSBCP and IXFCDBCP) into a database character column; both new and existing tables
  - A PC/IXF GRAPHIC data type into a database FOR BIT DATA column (SBCS CPGID = 0 = DBCS CPGID); new tables only (this is always permitted for existing tables).
- The FORCEIN option does not extend the scope of valid PC/IXF data types.  
PC/IXF columns with data types not defined as valid in “PC/IXF Data Types” on page 414 are invalid for import with or without the FORCEIN option.
  - In DBCS environments, the FORCEIN option permits import of:
    - A PC/IXF BIT data type into a database character column
    - A PC/IXF BIT data type into a database graphic column; however, if the PC/IXF BIT column is of fixed length, that length must be even. A fixed length PC/IXF BIT column of odd length is not compatible with a database graphic column. A varying-length PC/IXF BIT column *is* compatible whether its length is odd or even, although an odd-length value from a varying-length column is an invalid value for import into a database graphic column
    - A PC/IXF MIXED data type into a database character column.

Table 18 summarizes PC/IXF file import into new or existing database tables with the FORCEIN option.

*Table 18 (Page 1 of 2). Summary of PC/IXF File Import with FORCEIN Option*

PC/IXF COLUMN DATA TYPE	DATABASE COLUMN DATA TYPE										
	NUMERIC				CHARACTER			GRAPH	DATETIME		
	SMALL INT	INT	DEC	FLT	(0,0)	(SBCS, 0) <sup>e</sup>	(SBCS, DBCS) <sup>b</sup>	b	DATE	TIME	TIME STAMP
Numeric											
-SMALLINT	N										
	E	E	E <sup>a</sup>	E							
-INTEGER		N									
	E <sup>a</sup>	E	E <sup>a</sup>	E							
-DECIMAL			N								
	E <sup>a</sup>	E <sup>a</sup>	E <sup>a</sup>	E							
-FLOAT				N							
	E <sup>a</sup>	E <sup>a</sup>	E <sup>a</sup>	E							
Character											
-(0,0)					N						
					E	E w/F	E w/F	E w/F	E <sup>c</sup>	E <sup>c</sup>	E <sup>c</sup>
-(SBCS,0)						N	N				
					E	E	E		E <sup>c</sup>	E <sup>c</sup>	E <sup>c</sup>
-(SBCS, DBCS)						N w/F <sup>d</sup>	N		E <sup>c</sup>	E <sup>c</sup>	E <sup>c</sup>
					E	E w/F	E				

## PC Version of IXF File Format

Table 18 (Page 2 of 2). Summary of PC/IXF File Import with FORCEIN Option

PC/IXF COLUMN DATA TYPE	DATABASE COLUMN DATA TYPE										
	NUMERIC				CHARACTER			GRAPH	DATETIME		
	SMALL INT	INT	DEC	FLT	(0,0)	(SBCS, 0) <sup>e</sup>	(SBCS, DBCS) <sup>b</sup>	b	DATE	TIME	TIME STAMP
Graphic											
					N w/F <sup>d</sup>			N			
					E			E			
Datetime											
-DATE								N			
								E			
-TIME									N		
									E		
-TIME STAMP											N
											E
<p><b>Note:</b> If a PC/IXF column can be imported into a database column only with the FORCEIN option, the string 'w/F' is displayed together with an 'N' or an 'E'. An 'N' indicates that the utility is creating a new database table; an 'E' indicates that the utility is importing data to an existing database table. The FORCEIN option affects compatibility of character and graphic data types only.</p> <p><sup>a</sup> Individual values are rejected if they are out of range for the target numeric data type.</p> <p><sup>b</sup> Data type is available only in DBCS environments.</p> <p><sup>c</sup> Individual values are rejected if they are not valid date or time values.</p> <p><sup>d</sup> Applies only if the source PC/IXF data type is not supported by the target database.</p> <p><sup>e</sup> Data type is not available in DBCS environments.</p>											

### Differences between Version 1 PC/IXF and Version 0 System/370 IXF

The following describes differences between Version 1 PC/IXF, used by the database manager, and Version 0 System/370 IXF, used by several host database products:

- PC/IXF files are ASCII, rather than EBCDIC oriented. PC/IXF files have significantly expanded code page identification, including new code page identifiers in the H record, and the use of actual code page values in the column descriptor records. There is also a mechanism for marking columns of character data as FOR BIT DATA. FOR BIT DATA columns are of special significance, because transforms which convert a PC/IXF file format to or from any other IXF or database file format cannot perform any code page translation on the values contained in FOR BIT DATA columns.
- Only the machine data form is permitted; that is, the IXFTFORM field must always contain the value M. Furthermore, the machine data must be in PC forms; that is, the IXFTMFRM field must contain the value PC. This means that integers, floating point numbers, and decimal numbers in data portions of PC/IXF data records must be in PC forms.

## Non-delimited ASCII (ASC) File Format

- Application (A) records are permitted anywhere after the H record in a PC/IXF file. They are not counted when the value of the IXFHHCNT field is computed.
- Every PC/IXF record begins with a record length indicator. This is a 6-byte character representation of an integer value containing the length, in bytes, of the PC/IXF record not including the record length indicator itself; that is, the total record length minus 6 bytes. The purpose of the record length field is to enable PC programs to identify record boundaries.
- To facilitate the compact storage of variable-length data, and to avoid complex processing when a field is split into multiple records, PC/IXF does not support Version 0 IXF X records, but does support D record identifiers. Whenever a variable-length field or a nullable field is the last field in a data D record, it is not necessary to write the entire maximum length of the field to the PC/IXF file.

---

## Non-delimited ASCII (ASC) File Format

A Non-delimited ASCII (ASC) file is a sequential ASCII file with row delimiters. It can be used for data exchange with any ASCII product that has a columnar format for data, including word processors. Each ASC file is a stream of ASCII characters consisting of data values ordered by row and column. Rows in the data stream are separated by row delimiters. Each column within a row is defined by a beginning-ending location pair (specified by IMPORT parameters). Each pair represents locations within a row specified as byte positions. The first position within a row is byte position 1. The first element of each location pair is the byte on which the column begins, and the second element of each location pair is the byte on which the column ends. The columns may overlap. Every row in an ASC file has the same column definition.

An ASC file is defined by:

```
ASC file ::= Row 1 data || Row delimiter ||  
           Row 2 data || Row delimiter ||  
           .  
           .  
           .  
           Row n data  
  
Row i data ::= ASCII characters || Row delimiter  
  
Row Delimiter ::= ASCII line feed sequencea
```

<sup>a</sup> The record delimiter is assumed to be a new line character, ASCII x0A. Data generated on OS/2 or the Windows operating system can use the carriage return/line feed 2-byte standard of 0x0D0A. Data in EBCDIC code pages should use the EBCDIC LF character (0x25) as the record delimiter (EBCDIC data can be loaded using the CODEPAGE option on the LOAD command). The record delimiter is never interpreted to be part of a field of data.

## Non-delimited ASCII (ASC) File Format

### Sample ASC File

Following is an example of an ASC file. Each line ends with a line feed sequence (on OS/2 or the Windows operating system, each line ends with a carriage return/line feed sequence).

```
Smith, Bob      4973      15.46
Jones, Suzanne 12345     16.34
Williams, Sam   452123    193.78
```

#### Notes:

1. ASC files are assumed not to contain column names.
2. Character strings are *not* enclosed by delimiters. The data type of a column in the ASC file is determined by the data type of the target column in the database table.
3. A null is imported into a nullable database column if:
  - A field of blanks is targeted for a numeric, DATE, TIME, or TIMESTAMP database column
  - A field with no beginning and ending location pairs is specified
  - A location pair with beginning and ending locations equal to zero is specified
  - A row of data is too short to contain a valid value for the target column.
4. If the target column is not nullable, an attempt to import a field of blanks into a numeric, DATE, TIME, or TIMESTAMP column causes the row to be rejected.
5. If the input data is not compatible with the target column, and that column is nullable, a null is imported or the row is rejected, depending on where the error is detected. If the column is not nullable, the row is rejected. Messages are written to the message file, specifying incompatibilities that are found.

### ASC Data Type Descriptions

Data Type	Form Acceptable to the Import Utility
BLOB/CLOB	A string of characters. The character string is truncated on the right, if necessary, to match the maximum length of the target column. If the ASC <i>truncate blanks</i> option is in effect, trailing blanks are stripped from the original or the truncated string.
BLOB_FILE, CLOB_FILE, DBCLOB_FILE (DBCS only)	A delimited or non-delimited name of the file that holds the data.
CHAR	A string of characters. The character string is truncated or padded with spaces on the right, if necessary, to match the width of the target column.
DATE	A character string representing a date value in a format consistent with the country code of the target database.  The beginning and ending locations should specify a field width that is within the range for the external representation of a date.

## Non-delimited ASCII (ASC) File Format

<i>Table 19 (Page 2 of 3). Acceptable Data Type Forms for the ASC File Format</i>	
<b>Data Type</b>	<b>Form Acceptable to the Import Utility</b>
DBCLOB (DBCS only)	A string of an even number of bytes. A string of an odd number of bytes is invalid and is not accepted. A valid string is truncated on the right, if necessary, to match the maximum length of the target column.
DECIMAL	<p>A constant in any numeric type (SMALLINT, INTEGER, DECIMAL, or FLOAT) is accepted. Individual values are rejected if they are not in the range of the database column into which they are being imported. If the input value has more digits after the decimal point than the scale of the database column, the excess digits are truncated. A comma, period, or colon is considered to be a decimal point. Thousands separators are not allowed.</p> <p>The beginning and ending locations should specify a field whose width does not exceed 50 bytes. Integers, decimal numbers, and the mantissas of floating point numbers can have no more than 31 digits. Exponents of floating point numbers can have no more than 3 digits.</p>
FLOAT(long)	<p>A constant in any numeric type (SMALLINT, INTEGER, DECIMAL, or FLOAT) is accepted. All values are valid. A comma, period, or colon is considered to be a decimal point. An uppercase or lowercase E is accepted as the beginning of the exponent of a FLOAT constant.</p> <p>The beginning and ending locations should specify a field whose width does not exceed 50 bytes. Integers, decimal numbers, and the mantissas of floating point numbers can have no more than 31 digits. Exponents of floating point numbers can have no more than 3 digits.</p>
GRAPHIC (DBCS only)	A string of an even number of bytes. A string of an odd number of bytes is invalid and is not accepted. A valid string is truncated or padded with double-byte spaces (0x8140) on the right, if necessary, to match the maximum length of the target column.
INTEGER	<p>A constant in any numeric type (SMALLINT, INTEGER, DECIMAL, or FLOAT) is accepted. Individual values are rejected if they are not in the range -2 147 483 648 to 2 147 483 647. Decimal numbers are truncated to integer values. A comma, period, or colon is considered to be a decimal point. Thousands separators are not allowed.</p> <p>The beginning and ending locations should specify a field whose width does not exceed 50 bytes. Integers, decimal numbers, and the mantissas of floating point numbers can have no more than 31 digits. Exponents of floating point numbers can have no more than 3 digits.</p>
LONG VARCHAR	A string of characters. The character string is truncated on the right, if necessary, to match the maximum length of the target column. If the ASC <i>truncate blanks</i> option is in effect, trailing blanks are stripped from the original or the truncated string.



## Non-delimited ASCII (ASC) File Format

<i>Table 19 (Page 3 of 3). Acceptable Data Type Forms for the ASC File Format</i>	
<b>Data Type</b>	<b>Form Acceptable to the Import Utility</b>
LONG VARGRAPHIC (DBCS only)	A string of an even number of bytes. A string of an odd number of bytes is invalid and is not accepted. A valid string is truncated on the right, if necessary, to match the maximum length of the target column.
SMALLINT	<p>A constant in any numeric type (SMALLINT, INTEGER, DECIMAL, or FLOAT) is accepted. Individual values are rejected if they are not in the range -32 768 to 32 767. Decimal numbers are truncated to integer values. A comma, period, or colon is considered to be a decimal point. Thousands separators are not allowed.</p> <p>The beginning and ending locations should specify a field whose width does not exceed 50 bytes. Integers, decimal numbers, and the mantissas of floating point numbers can have no more than 31 digits. Exponents of floating point numbers can have no more than 3 digits.</p>
TIME	<p>A character string representing a time value in a format consistent with the country code of the target database.</p> <p>The beginning and ending locations should specify a field width that is within the range for the external representation of a time.</p>
TIMESTAMP	<p>A character string representing a time stamp value acceptable for storage in a database.</p> <p>The beginning and ending locations should specify a field width that is within the range for the external representation of a time stamp.</p>
VARCHAR	A string of characters. The character string is truncated on the right, if necessary, to match the maximum length of the target column. If the ASC <i>truncate blanks</i> option is in effect, trailing blanks are stripped from the original or the truncated string.
VARGRAPHIC (DBCS only)	A string of an even number of bytes. A string of an odd number of bytes is invalid and is not accepted. A valid string is truncated on the right, if necessary, to match the maximum length of the target column.

# Non-delimited ASCII (ASC) File Format

---

## Appendix D. How the DB2 Library Is Structured

The DB2 Universal Database library consists of SmartGuides, online help, and books. This section describes the information that is provided, and how to access it.

To help you access product information online, DB2 provides the Information Center on OS/2, Windows 95, and the Windows NT operating systems. You can view task information, DB2 books, troubleshooting information, sample programs, and DB2 information on the Web. "About the Information Center" on page 448 has more details.

---

### SmartGuides

SmartGuides help you complete some administration tasks by taking you through each task one step at a time. SmartGuides are available on OS/2, Windows 95, and the Windows NT operating systems. The following table lists the SmartGuides.

<b>SmartGuide</b>	<b>Helps you to...</b>	<b>How to Access...</b>
<i>Add Database</i>	Catalog a database on a client workstation.	From the Client Configuration Assistant, click on <b>Add</b> .
<i>Create Database</i>	Create a database, and to perform some basic configuration tasks.	From the Control Center, click with the right mouse button on the <b>Databases</b> icon and select <b>Create-&gt;New</b> .
<i>Performance Configuration</i>	Tune the performance of a database by updating configuration parameters to match your business requirements.	From the Control Center, click with the right mouse button on the database you want to tune and select <b>Configure performance</b> .
<i>Backup Database</i>	Determine, create, and schedule a backup plan.	From the Control Center, click with the right mouse button on the database you want to backup and select <b>Backup-&gt;Database using SmartGuide</b> .
<i>Restore Database</i>	Recover a database after a failure. It helps you understand which backup to use, and which logs to replay.	From the Control Center, click with the right mouse button on the database you want to restore and select <b>Restore-&gt;Database using SmartGuide</b> .
<i>Create Table</i>	Select basic data types, and create a primary key for the table.	From the Control Center, click with the right mouse button on the <b>Tables</b> icon and select <b>Create-&gt;Table using SmartGuide</b> .
<i>Create Table Space</i>	Create a new table space.	From the Control Center, click with the right mouse button on the <b>Table spaces</b> icon and select <b>Create-&gt;Table space using SmartGuide</b> .

---

## Online Help

Online help is available with all DB2 components. The following table describes the various types of help.

Type of Help	Contents	How to Access...
<i>Command Help</i>	Explains the syntax of commands in the command line processor.	From the command line processor in interactive mode, enter:  <i>? command</i>  where <i>command</i> is a keyword or the entire command.  For example, <i>? catalog</i> displays help for all the CATALOG commands, whereas <i>? catalog database</i> displays help for the CATALOG DATABASE command.
<i>Control Center Help</i>	Explains the tasks you can perform in a window or notebook. The help includes prerequisite information you need to know, and describes how to use the window or notebook controls.	From a window or notebook, click on the <b>Help</b> push button or press the F1 key.
<i>Message Help</i>	Describes the cause of a message number, and any action you should take.	From the command line processor in interactive mode, enter:  <i>? message number</i>  where <i>message number</i> is a valid message number.  For example, <i>? SQL30081</i> displays help about the SQL30081 message.  To view message help one screen at a time, enter:  <i>? XXXnnnnn   more</i>  where <i>XXX</i> is the message prefix, such as SQL, and <i>nnnnn</i> is the message number, such as 30081.  To save message help in a file, enter:  <i>? XXXnnnnn &gt; filename.ext</i>  where <i>filename.ext</i> is the file where you want to save the message help.  <b>Note:</b> On UNIX-based systems, enter:  <i>\? XXXnnnnn   more</i> or  <i>\? XXXnnnnn &gt; filename.ext</i>

Type of Help	Contents	How to Access...
<i>SQL Help</i>	Explains the syntax of SQL statements.	<p>From the command line processor in interactive mode, enter:</p> <p><b>help</b> <i>statement</i></p> <p>where <i>statement</i> is an SQL statement.</p> <p>For example, <b>help</b> <i>SELECT</i> displays help about the SELECT statement.</p>
<i>SQLSTATE Help</i>	Explains SQL states and class codes.	<p>From the command line processor in interactive mode, enter:</p> <p><b>? <i>sqlstate</i></b> or <b>? <i>class-code</i></b></p> <p>where <i>sqlstate</i> is a valid five digit SQL state and <i>class-code</i> is a valid two digit class code.</p> <p>For example, <b>? 08003</b> displays help for the 08003 SQL state, whereas <b>? 08</b> displays help for the 08 class code.</p>

---

## DB2 Books

The table in this section lists the DB2 books. They are divided into two groups:

- Cross-platform books: These books are for DB2 on any of the supported platforms.
- Platform-specific books: These books are for DB2 on a specific platform. For example, there is a separate *Quick Beginnings* book for DB2 on OS/2, Windows NT, and UNIX-based operating systems.

Most books are available in HTML and PostScript format, and in hardcopy that you can order from IBM. The exceptions are noted in the table.

You can obtain DB2 books and access information in a variety of different ways:

- View** To view an HTML book, you can do the following:
- If you are running DB2 administration tools on OS/2, Windows 95, or the Windows NT operating systems, you can use the Information Center. “About the Information Center” on page 448 has more details.
  - Use the open file function of the Web browser supplied by DB2 (or one of your own) to open the following page:  

```
sqllib/doc/html/index.htm
```

The page contains descriptions of and links to the DB2 books. The path is located on the drive where DB2 is installed.

You can also open the page by double-clicking on the **DB2 Online Books** icon. Depending on the system you are using, the icon is in the main product folder or the Windows Start menu.
- Search** To search for information in the HTML books, you can do the following:
- Click on **Search the DB2 Books** at the bottom of any page in the HTML books. Use the search form to find a specific topic.
  - Click on **Index** at the bottom of any page in an HTML book. Use the Index to find a specific topic in the book.
  - Display the Table of Contents or Index of the HTML book, and then use the find function of the Web browser to find a specific topic in the book.
  - Use the bookmark function of the Web browser to quickly return to a specific topic.
  - Use the search function of the Information Center to find specific topics. “About the Information Center” on page 448 has more details.
- Print** To print a book on a PostScript printer, look for the file name shown in the table.
- Order** To order a hardcopy book from IBM, use the form number.

<b>Book Name</b>	<b>Book Description</b>	<b>Form Number</b> <b>File Name</b>
<b>Cross-Platform Books</b>		
<i>Administration Getting Started</i>	Introduces basic DB2 database administration concepts and tasks, and walks you through the primary administrative tasks.	S10J-8154 db2k0x50
<i>Administration Guide</i>	Contains information required to design, implement, and maintain a database to be accessed either locally or in a client/server environment.	S10J-8157 db2d0x50
<i>API Reference</i>	Describes the DB2 application programming interfaces (APIs) and data structures you can use to manage your databases. Explains how to call APIs from your applications.	S10J-8167 db2b0x50
<i>CLI Guide and Reference</i>	Explains how to develop applications that access DB2 databases using the DB2 Call Level Interface, a callable SQL interface that is compatible with the Microsoft ODBC specification.	S10J-8159 db2l0x50
<i>Command Reference</i>	Explains how to use the command line processor, and describes the DB2 commands you can use to manage your database.	S10J-8166 db2n0x50
<i>DB2 Connect Enterprise Edition Quick Beginnings</i>	Provides planning, installing, configuring, and using information for DB2 Connect Enterprise Edition. Also contains installation and setup information for all supported clients.	S10J-7888 db2cyx50
<i>DB2 Connect Personal Edition Quick Beginnings</i>	Provides planning, installing, configuring, and using information for DB2 Connect Personal Edition.	S10J-8162 db2c1x50
<i>DB2 Connect User's Guide</i>	Provides concepts, programming and general using information about the DB2 Connect products.	S10J-8163 db2c0x50
<i>DB2 Connectivity Supplement</i>	Provides setup and reference information for customers who want to use DB2 for AS/400, DB2 for OS/390, DB2 for MVS, or DB2 for VM as DRDA Application Requesters with DB2 Universal Database servers, and customers who want to use DRDA Application Servers with DB2 Connect (formerly DDCS) application requesters.  <b>Note:</b> Available in HTML and PostScript formats only.	No form number db2h1x50
<i>Embedded SQL Programming Guide</i>	Explains how to develop applications that access DB2 databases using embedded SQL, and includes discussions about programming techniques and performance considerations.	S10J-8158 db2a0x50
<i>Glossary</i>	Provides a comprehensive list of all DB2 terms and definitions.  <b>Note:</b> Available in HTML format only.	No form number db2t0x50

<b>Book Name</b>	<b>Book Description</b>	<b>Form Number File Name</b>
<i>Installing and Configuring DB2 Clients</i>	Provides installation and setup information for all DB2 Client Application Enablers and DB2 Software Developer's Kits.  <b>Note:</b> Available in HTML and PostScript formats only.	No form number db2iyx50
<i>Master Index</i>	Contains a cross reference to the major topics covered in the DB2 library.  <b>Note:</b> Available in PostScript format and hardcopy only.	S10J-8170 db2w0x50
<i>Message Reference</i>	Lists messages and codes issued by DB2, and describes the actions you should take.	S10J-8168 db2m0x50
<i>Replication Guide and Reference</i>	Provides planning, configuring, administering, and using information for the IBM Replication tools supplied with DB2.	S95H-0999 db2e0x50
<i>Road Map to DB2 Programming</i>	Introduces the different ways your applications can access DB2, describes key DB2 features you can use in your applications, and points to detailed sources of information for DB2 programming.	S10J-8155 db2u0x50
<i>SQL Getting Started</i>	Introduces SQL concepts, and provides examples for many constructs and tasks.	S10J-8156 db2y0x50
<i>SQL Reference</i>	Describes SQL syntax, semantics, and the rules of the language. Also includes information about release-to-release incompatibilities, product limits, and catalog views.	S10J-8165 db2s0x50
<i>System Monitor Guide and Reference</i>	Describes how to collect different kinds of information about your database and the database manager. Explains how you can use the information to understand database activity, improve performance, and determine the cause of problems.	S10J-8164 db2f0x50
<i>Troubleshooting Guide</i>	Helps you determine the source of errors, recover from problems, and use diagnostic tools in consultation with DB2 Customer Service.	S10J-8169 db2p0x50
<i>What's New</i>	Describes the new features, functions, and enhancements in DB2 Universal Database.  <b>Note:</b> Available in HTML and PostScript formats only.	No form number db2q0x50
<b>Platform-Specific Books</b>		
<i>Building Applications for UNIX Environments</i>	Provides environment setup information and step-by-step instructions to compile, link, and run DB2 applications on a UNIX system.	S10J-8161 db2axx50
<i>Building Applications for Windows and OS/2 Environments</i>	Provides environment setup information and step-by-step instructions to compile, link, and run DB2 applications on a Windows or OS/2 system.	S10J-8160 db2a1x50



<b>Book Name</b>	<b>Book Description</b>	<b>Form Number</b> <b>File Name</b>
<i>DB2 Extended Enterprise Edition Quick Beginnings</i>	Provides planning, installing, configuring, and using information for DB2 Universal Database Extended Enterprise Edition for AIX.	S72H-9620 db2v3x50
<i>DB2 Personal Edition Quick Beginnings</i>	Provides planning, installing, configuring, and using information for DB2 Universal Database Personal Edition on OS/2, Windows 95, and the Windows NT operating systems.	S10J-8150 db2i1x50
<i>DB2 SDK for Macintosh Building Your Applications</i>	Provides environment setup information and step-by-step instructions to compile, link, and run DB2 applications on a Macintosh system.  <b>Note:</b> Available in PostScript format and hardcopy for DB2 Version 2.1.2 only.	S50H-0528 sqla7x02
<i>DB2 SDK for SCO OpenServer Building Your Applications</i>	Provides environment setup information and step-by-step instructions to compile, link, and run DB2 applications on a SCO OpenServer system.  <b>Note:</b> Available for DB2 Version 2.1.2 only.	S89H-3242 sqla9x02
<i>DB2 SDK for Silicon Graphics IRIX Building Your Applications</i>	Provides environment setup information and step-by-step instructions to compile, link, and run DB2 applications on a Silicon Graphics system.  <b>Note:</b> Available in PostScript format and hardcopy for DB2 Version 2.1.2 only.	S89H-4032 sqlaax02
<i>DB2 SDK for SINIX Building Your Applications</i>	Provides environment setup information and step-by-step instructions to compile, link, and run DB2 applications on a SINIX system.  <b>Note:</b> Available in PostScript format and hardcopy for DB2 Version 2.1.2 only.	S50H-0530 sqla8x00
<i>Quick Beginnings for OS/2</i>	Provides planning, installing, configuring, and using information for DB2 Universal Database on OS/2. Also contains installing and setup information for all supported clients.	S10J-8147 db2i2x50
<i>Quick Beginnings for UNIX</i>	Provides planning, installing, configuring, and using information for DB2 Universal Database on UNIX-based platforms. Also contains installing and setup information for all supported clients.	S10J-8148 db2ixx50
<i>Quick Beginnings for Windows NT</i>	Provides planning, installing, configuring, and using information for DB2 Universal Database on the Windows NT operating system. Also contains installing and setup information for all supported clients.	S10J-8149 db2i6x50

**Notes:**

1. The character in the sixth position of the file name indicates the language of a book. For example, the file name db2d0e50 indicates that the *Administration Guide* is in English. The following letters are used in the file names to indicate the language of a book:

<b>Language</b>	<b>Identifier</b>	<b>Language</b>	<b>Identifier</b>
Brazilian Portuguese	B	Hungarian	H
Bulgarian	U	Italian	I
Czech	X	Norwegian	N
Danish	D	Polish	P
English	E	Russian	R
Finnish	Y	Slovenian	L
French	F	Spanish	Z
German	G	Swedish	S

2. For late breaking information that could not be included in the DB2 books, see the README file. Each DB2 product includes a README file which you can find in the directory where the product is installed.

---

## About the Information Center

The Information Center provides quick access to DB2 product information. The Information Center is available on OS/2, Windows 95, and the Windows NT operating systems. You must install the DB2 administration tools to see the Information Center.

Depending on your system, you can access the Information Center from the:

- Main product folder
- Toolbar in the Control Center
- Windows Start menu.

The Information Center provides the following kinds of information. Click on the appropriate tab to look at the information:

<b>Tasks</b>	Lists tasks you can perform using DB2.
<b>Reference</b>	Lists DB2 reference information, such as keywords, commands, and APIs.
<b>Books</b>	Lists DB2 books.
<b>Troubleshooting</b>	Lists categories of error messages and their recovery actions.
<b>Sample Programs</b>	Lists sample programs that come with the DB2 Software Developer's Kit. If the Software Developer's Kit is not installed, this tab is not displayed.
<b>Web</b>	Lists DB2 information on the World Wide Web. To access this information, you must have a connection to the Web from your system.

When you select an item in one of the lists, the Information Center launches a viewer to display the information. The viewer might be the system help viewer, an editor, or a Web browser, depending on the kind of information you select.

The Information Center provides search capabilities so you can look for specific topics, and filter capabilities to limit the scope of your searches.



---

## Appendix E. Notices

Any reference to an IBM licensed program in this publication is not intended to state or imply that only IBM's licensed program may be used. Any functionally equivalent product, program or service that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program, or service. Evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, is the user's responsibility.

IBM may have patents or pending patent applications covering subject matter 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 the

IBM Director of Licensing,  
IBM Corporation,  
500 Columbus Avenue,  
Thornwood, NY, 10594  
USA.

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 Canada Limited  
Department 071  
1150 Eglinton Ave. East  
North York, Ontario  
M3C 1H7  
CANADA

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

This publication 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.

---

## Trademarks

The following terms are trademarks or registered trademarks of the IBM Corporation in the United States and/or other countries:

ACF/VTAM	MVS/ESA
ADSTAR	MVS/XA
AISPO	NetView
AIX	OS/400
AIXwindows	OS/390
AnyNet	OS/2
APPN	PowerPC
AS/400	QMF
CICS	RACF
C Set++	RISC System/6000
C/370	SAA
DATABASE 2	SP
DatagLANce	SQL/DS
DataHub	SQL/400
DataJoiner	S/370
DataPropagator	System/370
DataRefresher	System/390
DB2	SystemView
Distributed Relational Database Architecture	VisualAge
DRDA	VM/ESA
Extended Services	VSE/ESA
FFST	VTAM
First Failure Support Technology	WIN-OS/2
IBM	
IMS	
Lan Distance	

---

## Trademarks of Other Companies

The following terms are trademarks or registered trademarks of the companies listed:

C-bus is a trademark of Corollary, Inc.

HP-UX is a trademark of Hewlett-Packard.

Java and HotJava are trademarks of Sun Microsystems, Inc.

Microsoft, Windows, Windows NT, and the Windows 95 logo are trademarks or registered trademarks of Microsoft Corporation.

PC Direct is a trademark of Ziff Communications Company and is used by IBM Corporation under license.

Solaris is a trademark of Sun Microsystems, Inc.

UNIX is a registered trademark in the United States and other countries licensed exclusively through X/Open Company Limited.

Other company, product, or service names, which may be denoted by a double asterisk (\*\*), may be trademarks or service marks of others.

---

## Index

### Special Characters

!, shell command 69  
\, line continuation character 78

### A

abnormal termination  
  restart 335  
access path  
  creating new 352  
  optimizing 350  
action  
  precompile/bind option 101, 287  
ACTIVATE DATABASE 87  
ADD NODE 89  
admin configuration  
  file 170  
  network parameter values 376  
  resetting to default 327  
  sample 170  
adsm\_mgmtclass  
  database configuration parameter 177  
adsm\_nodename  
  database configuration parameter 177  
adsm\_owner  
  database configuration parameter 177  
adsm\_password  
  database configuration parameter 177  
agent\_stack\_sz  
  database manager configuration parameter 186  
agentpri  
  database manager configuration parameter 186  
alias  
  naming conventions 397  
anyorder 273  
app\_ctl\_heap\_sz  
  database configuration parameter 177  
APPC node  
  uncataloging 372  
applheapsz  
  database configuration parameter 177  
application record, PC/IXF 413  
ASC data type descriptions 437  
ASC file  
  format 436

ASC file (*continued*)  
  sample 437  
ASC, as an import file type 213  
aslheapsz  
  database manager configuration parameter 186  
ATTACH 91  
authentication  
  database manager configuration parameter 186  
authentication ID  
  naming conventions 397  
authorities  
  granting when creating a database 148  
authority level  
  direct, defined 173  
  for creating databases, granting 148  
  indirect, defined 173  
  report 172  
Autoloader 5  
autorestart  
  database configuration parameter 177  
avg\_appls  
  database configuration parameter 178

### B

backbufsz  
  database manager configuration parameter 186  
BACKUP DATABASE 93  
backup\_pending  
  database configuration parameter 178  
Benchmark Tool 7  
binarynumerics 276  
BIND 98  
  to create new access path 352  
Bind File Description Tool 12  
bindfile  
  precompile option 288  
binding  
  defaults 109  
  errors during 147  
  implicitly created schema 109, 301  
blocking  
  precompile/bind option 101, 289  
buffpage  
  database configuration parameter 178

## C

- case sensitivity 81
  - in naming conventions 397
- CATALOG APPC NODE 111
- CATALOG APPCLU NODE 114
- CATALOG APPN NODE 116
- CATALOG DATABASE 118
- CATALOG DCS DATABASE 121
- CATALOG GLOBAL DATABASE 124
- CATALOG IPX/SPX NODE 126
- CATALOG LOCAL NODE 129
- CATALOG NAMED PIPE NODE 131
- CATALOG NETBIOS NODE 133
- CATALOG ODBC DATA SOURCE 135
- CATALOG TCP/IP NODE 136
- catalogcache\_sz
  - database configuration parameter 178
- cataloging
  - database 118
  - host database 121
- CCSIDG
  - precompile/bind option 102, 289
- CCSIDM
  - precompile/bind option 102, 289
- CCSIDS
  - precompile/bind option 102, 289
- CHANGE DATABASE COMMENT 139
- CHANGE ISOLATION LEVEL 141
- character string delimiter 401
- characters, special
  - permitted in CLP commands 81
- chardel 164, 218, 277
- charsub
  - precompile/bind option 102, 289
- chngpgs\_thresh
  - database configuration parameter 178
- CLOSE statement
  - executing through the CLP 387
- cnulreqd
  - bind option 103
- codepage 275
  - database configuration parameter 178
- codeset
  - database configuration parameter 178
- coldel 164, 219, 277
- collection
  - precompile/bind option 103, 290
- column
  - naming conventions 397
- column descriptor record, PC/IXF 409
- column values, invalid 425
- columns, incompatible 425
- command line processor
  - accessing databases through 69
  - accessing help 70
  - batch mode 69
  - command mode 69
  - description 69
  - interactive input mode 69
  - invoking 69
  - quitting 69, 308
  - shell command 69
  - terminating 69, 368
  - using 78
- command syntax
  - interpreting 393
- compound 217
- configuration, admin
  - resetting to default 327
  - sample 170
- configuration, database
  - resetting to default 329
  - sample 175
  - updating 379
- configuration, database manager
  - sample 184
- conn\_elapse
  - database manager configuration parameter 186
- connect
  - precompile option 290
- CONNECT statement
  - database connection 78
- consistency
  - required for backup 96
- continuation character, line
  - in command line processor 78
- Control Center 13
- conventions, naming
  - for aliases 397
  - for authentication IDs 397
  - for columns 397
  - for database manager objects 397
  - for databases 397
  - for tables 397
  - for views 397
  - in SNA 397
- copyprotect
  - database configuration parameter 178



- country
  - database configuration parameter 178
- cpuspeed
  - database manager configuration parameter 186
- CREATE DATABASE 143
- Create Instance 38
- Create Sample Database 49
- cursor stability (CS)
  - changing 141

## D

- Data Declustering Tool 53
- data fragmentation
  - eliminating, by table reorganization 317
- data integrity
  - maintaining, with isolation levels 141
- data record, PC/IXF 411
- data skew, redistributing data in nodegroup 312
- data type descriptions
  - ASC 437
  - DEL 402
  - PC/IXF 419
- data types
  - PC/IXF 414
- database
  - cataloging 118
  - changing comments in directory 139
  - checking authorizations 172
  - connection, overview of 78
  - deleting, ensuring recovery with log files 157
  - exporting table to a file 161
  - home directory entry, definition of 233
  - implicit connection 78
  - importing file to table 212
  - indirect directory entry, definition of 233
  - information 200
  - migrating, command for 282
  - monitor, resetting 333
  - naming conventions 397
  - recovering 344
  - remote directory entry, definition of 233
  - removing 157
  - removing entries (uncataloging) 369
  - removing host DCS entries 371
  - reorganizing 320
  - restarting 335
  - restoring (rebuilding) 337
  - roll-forward recovery of 344
  - statistics 350

- database access
  - starting database manager 78
- database backup
  - history file 303
- database configuration
  - network parameter values 380
  - resetting to default 329
  - sample 175
  - updating 379
- Database Connection Services (DCS) directory
  - uncataloging entries 371
- database directories
  - changing comments 139
  - definition of 233
  - sample content of 232
- database manager
  - accessing from command prompt 1
  - instances of 196
  - monitor switches, checking 194, 197
  - starting 361
  - statistics 199
  - stopping 365
  - system commands 1
- database manager configuration
  - file 193
  - network parameter values 381
  - sample 184
- database monitor
  - description 383
- Database Pre-migration Tool 15
- database system monitor
  - GET DATABASE MANAGER MONITOR SWITCHES 194
  - GET MONITOR SWITCHES 197
  - GET SNAPSHOT 199
  - RESET MONITOR 333
  - UPDATE MONITOR SWITCHES 383
- database\_consistent
  - database configuration parameter 178
- database\_level
  - database configuration parameter 178
- datesiso 164
- datetime
  - precompile/bind option 103, 290
- db2
  - CMD description 69
  - command syntax 70
- DB2 Administration Server 3
- DB2 Governor 29

DB2 Governor Log Query 37  
 DB2 Interactive CLI 17  
 DB2 Profile Registry Command 50  
 DB2 SQL Explain Tool 27  
 DB2 Statistics Extraction Tool 44  
 db2admin 3  
 db2autold 5  
 db2batch 7  
 db2bfd 12  
 db2bcc 13  
 db2cidmg 14  
 db2ckmig 15  
 db2cli 17  
 db2cmd 18  
 db2drdat 19  
 db2empfa 21  
 db2eva 22  
 db2evmon 24  
 db2exfmt 25  
 db2expln 27  
 db2icrt 38  
 db2idrop 39  
 db2iilist 40  
 db2imigr 41  
 db2ipxad 42  
 db2iupdt 43  
 db2look 44  
 db2migdr 47  
 DB2OPTIONS  
     environment variable 71  
 db2rbind 48  
 db2sampl 49  
 db2set 50  
 db2split 53  
 db2sql92 54  
 db2start 57, 361  
 db2stop 58, 365  
 db2tbst 59  
 db2trc 60  
 db2uiddl 63  
 db2untag 64  
 db2vexp 66  
 dbheap  
     database configuration parameter 178  
 DEACTIVATE DATABASE 149  
 dec  
     precompile/bind option 103, 290  
 decdel  
     precompile/bind option 104, 291  
 DECLARE CURSOR statement  
     executing through the CLP 387  
 decplusblank 164  
 decpt 164, 219, 277  
 default  
     admin configuration, resetting to 327  
     database configuration, resetting to 329  
 deferred\_prepare  
     precompile option 291  
 degree  
     precompile/bind option 104, 291  
 DEL data type descriptions 402  
 DEL file  
     format 399  
     sample 400  
 delimited ASCII (DEL) file format 399  
 delimiter  
     character string 401  
 Deregister 151  
 DESCRIBE 152  
 DETACH 156  
 device, tape 94  
 dft\_account\_str  
     database manager configuration parameter 187  
 dft\_client\_adpt  
     database manager configuration parameter 187  
 dft\_client\_comm  
     database manager configuration parameter 187  
 dft\_degree  
     database configuration parameter 178  
 dft\_extent\_sz  
     database configuration parameter 179  
 dft\_loadrec\_ses  
     database configuration parameter 179  
 dft\_mon\_bufpool  
     database manager configuration parameter 187  
 dft\_mon\_lock  
     database manager configuration parameter 187  
 dft\_mon\_sort  
     database manager configuration parameter 187  
 dft\_mon\_stmt  
     database manager configuration parameter 187  
 dft\_mon\_table  
     database manager configuration parameter 187  
 dft\_mon\_uow  
     database manager configuration parameter 187  
 dft\_prefetch\_sz  
     database configuration parameter 179  
 dft\_queryopt  
     database configuration parameter 179

dftdbpath  
     database manager configuration parameter 187  
 diaglevel  
     database manager configuration parameter 187  
 diagpath  
     database manager configuration parameter 187  
 differences between PC/IXF and System/370 IXF 435  
 dir\_cache  
     database manager configuration parameter 187  
 dir\_obj\_name  
     database configuration parameter 179  
     database manager configuration parameter 187  
 dir\_path\_name  
     database manager configuration parameter 187  
 dir\_type  
     database manager configuration parameter 188  
 directories  
     Database Connection Services (DCS), uncataloging  
         entries 371  
     database, changing comments 139  
     deleting entries 372  
     node, removing entries from 372  
     system database, removing 369  
     uncataloging 369  
 disconnect  
     precompile option 292  
 disconnecting  
     command line processor front-end and back-end  
         processes 368  
 discover  
     database manager configuration parameter 188  
 discover\_comm  
     database manager configuration parameter 188  
 discover\_db  
     database configuration parameter 179  
 discover\_inst  
     database manager configuration parameter 188  
 Distributed Database Connection Services (DDCS)  
     supported connections to other systems 122  
 dlchktime  
     database configuration parameter 179  
 dos\_rqriblk  
     database manager configuration parameter 188  
 DRDA Trace 19  
 drda\_heap\_sz  
     database manager configuration parameter 188  
 DROP DATABASE 157  
 DROP NODE VERIFY 159  
 dumpfile 275

dumping a trace to file 61  
 Dynamic Visual Explain 66  
 dynamicrules  
     precompile/bind option 104, 292

## E

ECHO 160  
 Enable Multi-page File Allocation 21  
 environment variables  
     auto-commit option (-c) 72  
     DB2OPTIONS 71  
     display DB2 interactive prompt option (-p) 75  
     display output option (-o) 74  
     display SQLCODE/SQLSTATE option (-e) 73  
     log commands in history file option (-l) 74  
     read from input file option (-f) 73  
     save all output to file option (-z) 76  
     save to report file option (-r) 75  
     show SQLCA data option (-a) 72  
     show warning messages option (-w) 76  
     statement termination character option (-t) 76  
     stop execution on command error option (-s) 75  
     verbose output option (-v) 76  
 error messages  
     database configuration file 182  
     dropping remote database 157  
     during backup 96  
     during binding 110  
     invalid checksum, database configuration file 329,  
         380  
     invalid checksum, database manager configuration  
         file 328, 331, 376, 382  
 estore\_seg\_sz  
     database configuration parameter 179  
 Event Analyzer 22  
 Event Monitor Productivity Tool 24  
 example  
     forcein 430  
 explain  
     bind option 104, 292  
 Explain Table Format Tool 25  
 explsnap  
     precompile/bind option 105, 293  
 EXPORT 161  
 export utility file formats 399  
 exporting 161  
     choosing file formats for 163  
     database table to a file 161  
     to PC/IXF format 163

## F

fastparse 273  
fcm\_num\_anchors  
    database manager configuration parameter 188  
fcm\_num\_connect  
    database manager configuration parameter 188  
fcm\_num\_rqb  
    database manager configuration parameter 188  
fcm\_num buffers  
    database manager configuration parameter 188  
FETCH statement  
    executing through the CLP 387  
file format  
    delimited ASCII (DEL) 399  
    non-delimited ASCII (ASC) 436  
    PC version of IXF (PC/IXF) 404  
file formats  
    for exporting table to file 161  
    for importing file to table 213  
fileserv  
    database manager configuration parameter 188  
FORCE APPLICATION 167  
forcein 219, 277  
    code page semantics 429  
    data type semantics 433  
    example 430  
    general semantics 429  
    option 429  
    summary of PC/IXF file import with 434  
funcpath  
    precompile/bind option 105, 293

## G

generic  
    bind option 106  
GET ADMIN CONFIGURATION 169  
GET AUTHORIZATIONS 172  
GET CONNECTION STATE 174  
GET DATABASE CONFIGURATION 175  
GET DATABASE MANAGER CONFIGURATION 184  
GET DATABASE MANAGER MONITOR  
    SWITCHES 194  
GET INSTANCE 196  
Get IPX/SPX Internetwork Address 42  
GET MONITOR SWITCHES 197  
GET SNAPSHOT 199  
    effect on UPDATE MONITOR SWITCHES 383  
Get Tablespace State 59

grant  
    bind option 106  
grant\_group  
    bind option 106  
grant\_user  
    bind option 106

## H

header record, PC/IXF 406  
HELP 211  
host systems  
    cataloging databases located on 121  
    connections supported by DDCCS 122  
    removing DCS catalog entries 371  
host variables  
    not supported in command line processor 82

## I

implicit connection  
    database access 78  
implieddecimal 218, 275  
IMPORT 212  
    of PC/IXF files, with forcein 434  
import of PC/IXF files  
    data type-specific rules 426  
    general rules 424  
import utility file formats 399  
importing  
    choosing file formats for 216  
    database access through DDCCS 217  
    DEL files 216  
    file to database table 212  
    PC/IXF file to table 215  
    PC/IXF, multiple-part files 217  
    WSF files 216  
incompatible columns 425  
index  
    reorganization 324, 325  
    statistics 350  
indexif 219  
indexrec  
    database configuration parameter 179  
    database manager configuration parameter 189  
indexschema 219  
indexsort  
    database configuration parameter 179  
indicator  
    record length 405

indoubt transaction field, description 244  
 INITIALIZE TAPE 221  
 insert  
     precompile/bind option 106, 293  
 Integration Exchange Format (IXF) 404  
 intra\_parallel  
     database manager configuration parameter 189  
 invalid PC/IXF column values 425  
 invalid PC/IXF data type 418  
 INVOKE STORED PROCEDURE 222  
 ipx\_socket  
     database manager configuration parameter 189  
 IPX/SPX node  
     uncataloging 372  
 isolation  
     precompile/bind option 106, 294

## K

keepdari  
     database manager configuration parameter 189  
 keywords  
     syntax for 393

## L

langlevel  
     precompile option 294  
 level  
     precompile option 295  
 line continuation character  
     in command line processor 78  
 LIST ACTIVE DATABASES 224  
 LIST APPLICATIONS 225  
 LIST BACKUP/HISTORY 227  
 LIST COMMAND OPTIONS 229  
 LIST DATABASE DIRECTORY 231  
 LIST DCS APPLICATIONS 235  
 LIST DCS DIRECTORY 237  
 LIST DRDA INDOUBT TRANSACTIONS 239  
 LIST INDOUBT TRANSACTIONS 241  
 List Instances 40  
 LIST NODE DIRECTORY 245  
 LIST NODEGROUPS 248  
 LIST NODES 250  
 LIST ODBC DATA SOURCES 251  
 LIST PACKAGES/TABLES 253  
 LIST TABLESPACE CONTAINERS 256  
 LIST TABLESPACES 258

LOAD 262  
     remote filenames 266  
 LOAD QUERY 280  
 load utility file formats 399  
 lobsinfile 164, 217, 273  
 Local Database Directory Migration 47  
 local node  
     uncataloging 372  
 locklist  
     database configuration parameter 179  
 locks  
     resetting maximum to default 329  
 locktimeout  
     database configuration parameter 180  
 log file  
     listing during roll forward 347  
 log\_retain\_status  
     database configuration parameter 180  
 logbufsz  
     database configuration parameter 180  
 logfilsiz  
     database configuration parameter 180  
 loghead  
     database configuration parameter 180  
 logpath  
     database configuration parameter 180  
 logprimary  
     database configuration parameter 180  
 logretain  
     database configuration parameter 180  
 logsecond  
     database configuration parameter 180

## M

max\_connretries  
     database manager configuration parameter 189  
 max\_coordagents  
     database manager configuration parameter 189  
 max\_idleagents 190  
 max\_rt\_degree  
     database manager configuration parameter 189  
 max\_time\_diff  
     database manager configuration parameter 189  
 maxagents  
     database manager configuration parameter 190  
 maxappl  
     database configuration parameter 180  
 maxcagents  
     database manager configuration parameter 190

- maxdari
  - database manager configuration parameter 190
- maxfilop
  - database configuration parameter 180
- maxlocks
  - database configuration parameter 180
- maxtotfilop
  - database manager configuration parameter 190
- messages
  - accessing help text 70
  - precompile/bind option 107, 295
- metacharacters 81
- MIGRATE DATABASE 282
- Migrate Instance 41
- min\_priv\_mem
  - database manager configuration parameter 190
- mincommit
  - database configuration parameter 180
- mon\_heap\_sz
  - database manager configuration parameter 190
- monitoring databases 194, 197
- multipage\_alloc
  - database configuration parameter 180

## N

- naming a binary file for output 61
- naming conventions
  - for aliases 397
  - for authentication IDs 397
  - for columns 397
  - for database manager objects 397
  - for databases 397
  - for tables 397
  - for views 397
  - in SNA 397
- NetBIOS node
  - uncataloging 372
- newlogpath
  - database configuration parameter 181
- nextactive
  - database configuration parameter 181
- nname
  - database manager configuration parameter 190
- no commit (NC)
  - changing 141
- node
  - directories, removing entries from 372
- node, SOCKS 137

- nodefaults 273
- nodetype
  - database manager configuration parameter 190
- noeofchar 218, 275
- noheader 274
- nolinemacro
  - precompile option 295
- non-delimited ASCII (ASC) file format 436
- norowwarnings 274
- NULL string
  - use in setting blanks 78
- nullindchar 276
- num\_estore\_segs
  - database configuration parameter 181
- num\_freqvalues
  - database configuration parameter 181
- num\_initagents
  - database manager configuration parameter 190
- num\_iocleaners
  - database configuration parameter 181
- num\_ioservers
  - database configuration parameter 181
- num\_poolagents
  - database manager configuration parameter 190
- num\_quantiles
  - database configuration parameter 181
- numdb
  - database manager configuration parameter 191
- numsegs
  - database configuration parameter 181

## O

- objectname
  - database manager configuration parameter 191
- Open DB2 Command Window 18
- OPEN statement
  - executing through the CLP 388
- optimization 317
- option
  - forcein 429
- optlevel
  - precompile option 296
- output
  - precompile option 296
- owner
  - precompile/bind option 107, 296

## P

- package
  - precompile option 296
  - recreating 309
- packeddecimal 276
- pagefreespace 274
- parameters
  - syntax for 393
- PC version of IXF (PC/IXF) file format 404
- PC/IXF
  - contrasted with System/370 IXF 435
  - data type descriptions 419
  - data types 414
  - invalid column values 425
  - invalid data type 418, 424
  - record types 404, 406
  - valid data type 418
- PC/IXF file
  - format 404
- PC/IXF file import
  - data type-specific rules 426
  - general rules 424
  - with forcein 434
- PC/IXF record type
  - application 413
  - column descriptor 409
  - data 411
  - header 406
  - table 407
- pckcachesz
  - database configuration parameter 181
- performance, improving 325
  - by reorganizing tables 318
- PRECOMPILE PROGRAM 284
- Prepare Unique Index Conversion to V5 Semantics 63
- priv\_mem\_thresh
  - database manager configuration parameter 191
- privileges
  - direct, defined 173
  - granting when creating a database 148
  - indirect, defined 173
- PRUNE HISTORY 303

## Q

- qualifier
  - precompile/bind option 107, 296
- QUERY CLIENT 304
- query\_heap\_sz

- query\_heap\_sz (*continued*)
  - database manager configuration parameter 191
- queryopt
  - precompile/bind option 107, 296
- QUIESCE TABLESPACES FOR TABLE 305
- QUIT 308

## R

- read stability (RS)
  - changing 141
- REBIND 309
- Rebind all Packages 48
- rec\_his\_retentn
  - database configuration parameter 181
- reclen 218, 276
- record length indicator 405
- record type, PC/IXF
  - application 413
  - column descriptor 409
  - data 411
  - header 406
  - table 407
- record types
  - PC/IXF 404, 406
- recovering a database 337
- recovery
  - with roll forward 344
  - without roll forward 340, 341
- redirecting output 82
- REDISTRIBUTE NODEGROUP 312
- REGISTER 315
- release
  - database configuration parameter 181
  - database manager configuration parameter 191
  - precompile/bind option 107, 297
- Release Container Tag 64
- Remote Database Migration 14
- remote filenames
  - LOAD 266
- remote server
  - invoking stored procedure on 222
- Remove Instance 39
- REORGANIZE TABLE 317
- REORGCHK 320
- repeatable read (RR)
  - changing 141
- RESET ADMIN CONFIGURATION 327
- RESET DATABASE CONFIGURATION 329

RESET DATABASE MANAGER  
   CONFIGURATION 331  
 RESET MONITOR 333  
 RESTART DATABASE 335  
 restbufsz  
   database manager configuration parameter 191  
 RESTORE DATABASE 337  
 restore\_pending  
   database configuration parameter 181  
 restoring earlier versions of DB2 databases 337  
 resync\_interval  
   database manager configuration parameter 191  
 REWIND TAPE 343  
 roll forward  
   enabling, when backing up 96  
 ROLLFORWARD DATABASE 344  
 rollfwd\_pending  
   database configuration parameter 181  
 route\_obj\_name  
   database manager configuration parameter 191  
 rqrrioblk  
   database manager configuration parameter 191  
 rules governing PC/IXF file import 424, 426  
 RUNSTATS 350

**S**  
 sample ASC file 437  
 sample DEL file 400  
 schema  
   created when creating a database 147  
 SELECT statement  
   executing through the CLP 388  
   in EXPORT command 162  
   resolving ambiguous symbols in WHERE clause 81  
 semantics  
   forcein, code page 429  
   forcein, data type 433  
   forcein, general 429  
 seqdetect  
   database configuration parameter 182  
 SET CLIENT 353  
 SET RUNTIME DEGREE 356  
 SET TABLESPACE CONTAINERS 358  
 SET TAPE POSITION 360  
 sheapthresh  
   database manager configuration parameter 191  
 SIGALRM signal 364  
   starting the database manager 364  
 SIGINT signal, starting database manager 364  
 SIGINT signal, starting the database manager 364  
 SOCKS node 137  
 softmax  
   database configuration parameter 182  
 sortheap  
   database configuration parameter 182  
 spm\_log\_file\_sz  
   database manager configuration parameter 192  
 spm\_max\_resync  
   database manager configuration parameter 192  
 spm\_name  
   database manager configuration parameter 191  
 SQL NULL value  
   command line processor representation 82  
 SQL statements  
   accessing help 70  
   executing through the CLP 387  
 SQL92 Compliant SQL Statement Processor 54  
 sqlca  
   precompile option 297  
 SQLDA structure  
   calling server procedures that use, restrictions  
     on 222  
 sqlerror  
   precompile/bind option 108, 297  
 sqlflag  
   precompile option 297  
 sqlrules  
   precompile option 298  
 sqlstmtsz  
   database manager configuration parameter 192  
 sqlwarn  
   precompile/bind option 108, 298  
 ss\_logon  
   database manager configuration parameter 192  
 Start Control Center 13  
 START DATABASE MANAGER 361  
 Start DB2 57  
 start\_stop\_time  
   database manager configuration parameter 192  
 starting a trace 61  
 stat\_heap\_sz  
   database configuration parameter 182  
 statistics  
   database 350  
   database manager 199  
   reorganizing indexes 324  
 REORGCHK 320



- stmthear
  - database configuration parameter 182
- STOP DATABASE MANAGER 365
- Stop DB2 58
- storage
  - physical 317
- stored procedure
  - invoking 222
- strdel
  - precompile/bind option 108, 299
- striptblanks 218, 277
- striptnulls 218, 277
- structure
  - delimited ASCII (DEL) file 399
  - non-delimited ASCII (ASC) file 436
- svcname
  - database manager configuration parameter 192
- syncpoint
  - precompile option 299
- syntax
  - for command line processor SQL statements 387
  - for host variables not supported in command line processor 82
  - precompile option 299
- syntax diagrams 393
- sysadm\_group
  - database manager configuration parameter 192
- sysctrl\_group
  - database manager configuration parameter 192
- sysmaint\_group
  - database manager configuration parameter 192
- system commands 1
- system database directory
  - uncataloging 369
- System/370 IXF 435
  - contrasted with PC/IXF 435

## T

- table
  - exporting to a file 161
  - importing file to 212
  - naming conventions 397
  - reorganization, determining if required 320
  - statistics 350
- table record, PC/IXF 407
- table reorganization
  - command for 317
- tape device 94

- target
  - precompile option 299
- TCP/IP node
  - uncataloging 372
- TERMINATE 368
  - cautions on use of 368
- termination 368
  - abnormal 335
  - normal 366
- territory
  - database configuration parameter 182
- text
  - precompile/bind option 108, 300
- tm\_database
  - database manager configuration parameter 192
- totalfreespace 274
- tp\_mon\_name
  - database manager configuration parameter 192
- tpname
  - database manager configuration parameter 192
- Trace 60
- tracefile 19
- trust\_allcints
  - database manager configuration parameter 193
- trust\_clnauth
  - database manager configuration parameter 193

## U

- udf\_mem\_sz
  - database manager configuration parameter 193
- UNCATALOG DATABASE 369
- UNCATALOG DCS DATABASE 371
- UNCATALOG NODE 372
- UNCATALOG ODBC DATA SOURCE 374
- uncataloging
  - database entries 369
  - host DCS database entries 371
  - system database directory 369
- uncommitted read (UR)
  - changing 141
- UPDATE ADMIN CONFIGURATION 375
- UPDATE COMMAND OPTIONS 377
- UPDATE DATABASE CONFIGURATION 379
- UPDATE DATABASE MANAGER CONFIGURATION 381
- Update Instances 43
- UPDATE MONITOR SWITCHES 383
- UPDATE RECOVERY HISTORY FILE 385

- usedefaults 274
- user
  - authorization 172
- user\_exit\_status
  - database configuration parameter 182
- userexit
  - database configuration parameter 182
- util\_heap\_sz
  - database configuration parameter 182
- utility file formats 399

## V

- valid PC/IXF data type 418
- validate
  - precompile/bind option 108, 300
- variables
  - syntax for 393
- version
  - precompile option 301
- view
  - naming conventions 397

## W

- wchartype
  - precompile option 301
- WHERE clause
  - resolving ambiguous symbols in SELECT statement 81
- workstation, remote
  - cataloging databases 118
  - removing catalog entries for databases from 369
  - uncataloging from local workstation 372

---

## Contacting IBM

This section lists ways you can get more information from IBM.

If you have a technical problem, please take the time to review and carry out the actions suggested by the *Troubleshooting Guide* before contacting DB2 Customer Support. Depending on the nature of your problem or concern, this guide will suggest information you can gather to help us to serve you better.

For information or to order any of the DB2 Universal Database products contact an IBM representative at a local branch office or contact any authorized IBM software remarketer.

### Telephone

If you live in the U.S.A., call one of the following numbers:

- 1-800-237-5511 to learn about available service options.
- 1-800-IBM-CALL (1-800-426-2255) or 1-800-3IBM-OS2 (1-800-342-6672) to order products or get general information.
- 1-800-879-2755 to order publications.

For information on how to contact IBM outside of the United States, see Appendix A of the IBM Software Support Handbook. You can access this document by selecting the "Roadmap to IBM Support" item at: <http://www.ibm.com/support/>.

Note that in some countries, IBM-authorized dealers should contact their dealer support structure instead of the IBM Support Center.

### World Wide Web

<http://www.software.ibm.com/data/>  
<http://www.software.ibm.com/data/db2/library/>

The DB2 World Wide Web pages provide current DB2 information about news, product descriptions, education schedules, and more. The DB2 Product and Service Technical Library provides access to frequently asked questions, fixes, books, and up-to-date DB2 technical information. (Note that this information may be in English only.)

### Anonymous FTP Sites

<ftp.software.ibm.com>

Log on as anonymous. In the directory `/ps/products/db2`, you can find demos, fixes, information, and tools concerning DB2 and many related products.

### Internet Newsgroups

`comp.databases.ibm-db2`, `bit.listserv.db2-l`

These newsgroups are available for users to discuss their experiences with DB2 products.

### CompuServe

**GO IBMDB2** to access the IBM DB2 Family forums

All DB2 products are supported through these forums.

To find out about the IBM Professional Certification Program for DB2 Universal Database, go to <a href="http://www.software.ibm.com/data/db2/db2tech/db2cert.html">http://www.software.ibm.com/data/db2/db2tech/db2cert.html</a>
--



Part Number: 10J8166



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

S10J-8166-00



10J8166

