

DB2 Universal Database for OS/390 and z/OS



Release Planning Guide

Version 7

DB2 Universal Database for OS/390 and z/OS



Release Planning Guide

Version 7

Note

Before using this information and the product it supports, be sure to read the general information under “Notices” on page 141.

Fourth Edition, Softcopy Only (June 2003)

This edition applies to Version 7 of IBM DATABASE 2 Universal Database Server for OS/390 and z/OS (DB2 for OS/390 and z/OS), 5675-DB2, and to any subsequent releases until otherwise indicated in new editions. Make sure you are using the correct edition for the level of the product.

This softcopy version is based on the printed edition of the book and includes the changes indicated in the printed version by vertical bars. Additional changes made to this softcopy version of the book since the hardcopy book was published are indicated by the hash (#) symbol in the left-hand margin. Editorial changes that have no technical significance are not noted.

This and other books in the DB2 for OS/390 and z/OS library are periodically updated with technical changes. These updates are made available to licensees of the product on CD-ROM and on the Web (currently at www.ibm.com/software/data/db2/os390/library.html). Check these resources to ensure that you are using the most current information.

© Copyright International Business Machines Corporation 2001. All rights reserved.

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

About this book	vii
Who should read this book	vii
Product terminology and citations	vii
How to send your comments.	viii
Chapter 1. Summary of changes to DB2 for OS/390 and z/OS Version 7.	1
Enhancements for managing data	1
Enhancements for reliability, scalability, and availability.	1
Easier development and integration of e-business applications.	3
Improved connectivity	4
Features of DB2 for OS/390 and z/OS.	4
Migration considerations	5
Chapter 2. Managing data in DB2 for OS/390 and z/OS	7
Comprehensive statistics history	7
Utility lists with pattern matching and dynamic allocation	7
# LOAD utility enhancements	8
More DBADM authority	9
CREATE ALIAS authority for DBADM	9
CREATE VIEW authority for DBADM	9
Enhanced management of constraints	9
# Real-time statistics	10
# Enhanced aggregation of statistics.	11
Chapter 3. DB2 for your e-business—reliable, scalable, and available	13
Utility support for DB2	13
A new UNLOAD utility	14
A new COPYTOCOPY utility	15
Parallel LOAD with multiple inputs	15
# DB2 resynchronizes BSDS data sets during restart	16
Faster online REORG	17
More concurrency with online LOAD RESUME	17
More efficient processing for SQL queries	17
More transformations of subqueries into join	17
Fewer sorts for ORDER BY clauses	18
More parallelism for IN-list index access	19
Changing subsystem parameters without stopping DB2	19
Improved availability of some user objects	20
Allow retry of log-read request	20
Data sharing enhancements	20
Group attachment enhancements	20
Restart light	20
Persistent structure size changes	21
Additional data sharing enhancements	22
# Better monitoring of the distributed environment.	22
Additional enhancement for e-business	22
Chapter 4. Easier development and integration of e-business applications 25	
DB2 for OS/390 and z/OS and the multitier environment	25
Three-tier architecture	25
Application servers on the Web	27
Database access with DB2 Connect	27
Transaction manager support for multisite updates.	28

	DB2 XML Extender for OS/390 and z/OS	29
	Storing, transmitting, and searching XML documents	29
	Graphical support for DB2 XML Extender for OS/390 and z/OS	30
	More flexibility with SQL	30
	Support for UNION and UNION ALL operators	30
	Scrollable cursors	31
#	Wrapping of identity column values	34
#	Self-referencing subselect on UPDATE or DELETE	35
	The FETCH FIRST <i>n</i> ROWS ONLY clause	35
	Fast implicit close	36
	Support for USER and USING keywords on the CONNECT statement	37
	FOR UPDATE enhancement	37
	ORDER BY <i>expression</i> of the SELECT statement	37
	IN and OUT parameters of an SQL procedure	37
	SQL scalar functions	37
	Alternative datetime functions that use ISO rules	38
#	Compatibility for C <i>long</i> data type	38
	ODBC enhancements	38
#	SQLJ and JDBC enhancements	39
	LOAD utility enhancements	40
	Precompiler services	40
#	Using the C SQL statement coprocessor	41
	Using the COBOL SQL statement coprocessor	41
#	Using the PL/I SQL statement coprocessor	42
	Unicode support for international data and e-business	42
	How Unicode works	43
	Related enhancements	44
	 Chapter 5. Improvements in connectivity	45
	Support for COMMIT and ROLLBACK in stored procedures	45
	Support for Windows Kerberos security	46
	Support for encrypted password or user ID and password	46
	Support for encrypted change password	47
	Global transaction support for distributed applications.	47
	Reporting server-elapsed time at the workstation	47
	Additional enhancements for connectivity	48
	 Chapter 6. Features of DB2 for OS/390 and z/OS	49
	Reporting and governing your enterprise using QMF	49
	QMF Version 7	49
	QMF High Performance Option	50
	QMF for Windows.	50
	Managing your enterprise with the DB2 Management Clients Package	52
	Managing DB2 for OS/390 and z/OS from your workstation with Control Center	52
	Building DB2 stored procedures from your workstation	55
	Installing DB2 from a workstation	56
	Using workstation views of DB2 Explain output	57
	Estimating DB2 performance with DB2 Estimator	59
	Web-ready applications	60
	Text search capability for the Internet with Net Search Extender	60
	Net.Data for secure Web applications	61
	Integrating and analyzing your business information using the DB2 Warehouse Manager	62
	Data Management Tools for your database	64

Chapter 7. Planning for migration and fallback	69
Migration considerations (Version 5 to Version 7)	69
Scrollable cursors are supported	69
Unicode support	69
Utility enhancements	69
More than 32 000 databases are supported	69
Increased maximum number of data sets open	69
Log buffer size increased	70
Change in recording soft errors in SYS1.LOGREC data set	70
Revoking SYSADM authorization	70
Changed messages from the LOAD utility	70
Encoding schemes of string parameters for routines	70
Increased size for generated code	70
Consider increasing IDBACK and CTHREAD	70
Customized DB2I defaults can be migrated	71
Migrating a data sharing group	71
Creating tables with DBCS and mixed columns	71
Consider enlarging the BSDS	71
Stored procedures	71
Size calculations for the work file database	72
Shared read-only data function is removed	73
Data set password protection is removed	73
Private protocol function not enhanced	73
Type 2 indexes are required	73
Preparing for fallback to Version 5	74
Frozen objects	74
Other fallback considerations	76
Migration considerations (Version 6 to Version 7)	78
Unicode support	78
New default encoding scheme	78
Scrollable cursors are supported	78
Revoking SYSADM authorization	78
Creating tables with DBCS and mixed columns	78
Changed messages from the LOAD utility	79
Change in recording soft errors in SYS1.LOGREC data set	79
Consider increasing IDBACK and CTHREAD	79
Increased size for generated code	79
Encoding schemes of string parameters for routines	79
Customized DB2I defaults can be migrated	79
Migrating a data sharing group	79
Work file database size calculations	80
SYSIBM.SYSPSM and SYSIBM.SYSPSMOPTS no longer used	80
Preparing for fallback to Version 6	81
Frozen objects	81
Other fallback considerations	82
Release incompatibilities	83
Release incompatibilities (migration from Version 5)	83
Release incompatibilities (migration from Version 6)	89
Release coexistence	93
IRLM	93
Data sharing	93
Distributed environment	94
Installation changes	94
Version 7 panels	94
Subsystem parameters added to installation panels	96
Changes to sample jobs	96

Appendix A. Changes to commands	97
New commands	97
Changed commands	97
Appendix B. Changes to utilities	99
New utilities	99
Changed utilities	100
Other utility changes	105
Appendix C. Changes to SQL	107
New SQL statements	107
Changed SQL statements	107
New functions	117
Changed functions	119
Other SQL language changes	119
Appendix D. Catalog changes	121
New catalog tables	121
Changed catalog tables	122
New and revised indexes	123
Appendix E. EXPLAIN table changes	125
Format of the Version 7 PLAN_TABLE	125
Descriptions of new and changed columns in PLAN_TABLE	126
Changed columns in DSN_STATEMNT_TABLE	126
Appendix F. New and changed IFCIDs	127
New IFCIDs	127
Changed IFCIDs	127
Appendix G. Prerequisites of Version 7 of DB2 for OS/390 and z/OS	129
DB2 for OS/390 and z/OS Version 7 prerequisites	129
Hardware requirements	129
Program requirements and optional programs	131
Function-Dependent Requirements for Features of DB2 Universal Database	
Server for OS/390 and z/OS	136
DB2 Installer requirements	136
DB2 for OS/390 and z/OS Visual Explain requirements	136
DB2 Estimator requirements	137
Net.Data requirements	137
DB2 Warehouse Center Requirements	137
QMF requirements	138
Appendix H. How to use the DB2 library	139
Notices	141
Programming interface information	143
Trademarks	143
Glossary	145
Bibliography	153
Index	161

About this book

DB2 Release Planning Guide is intended to help you plan for Version 7 of the licensed program DB2 Universal Database™ for OS/390® and z/OS.

Important

In this version of DB2® for OS/390 and z/OS™, some utility functions are available as optional products. You must separately order and purchase a license to such utilities, and discussion of those utility functions in this publication is not intended to otherwise imply that you have a license to them.

Who should read this book

This book is intended for all users of DB2, including application programmers, database administrators, and system programmers. It assumes that the user is familiar with DB2 Version 6.

Product terminology and citations

In this book, DB2 Universal Database Server for OS/390 and z/OS is referred to as "DB2 for OS/390 and z/OS." In cases where the context makes the meaning clear, DB2 for OS/390 and z/OS is referred to as "DB2." When this book refers to other books in this library, a short title is used. (For example, "See *DB2 SQL Reference*" is a citation to *IBM® DATABASE 2™ Universal Database Server for OS/390 and z/OS SQL Reference*.)

When referring to a DB2 product other than DB2 for OS/390 and z/OS, this book uses the product's full name to avoid ambiguity.

The following terms are used as indicated:

DB2 Represents either the DB2 licensed program or a particular DB2 subsystem.

C, C++, and the C language
Represent the C or C++ programming language.

CICS® Represents CICS/ESA® and CICS Transaction Server for OS/390.

IMS™ Represents IMS or IMS/ESA®.

MVS Represents the MVS element of OS/390.

OS/390
Represents the OS/390 or z/OS operating system.

RACF®
Represents the functions that are provided by the RACF component of the SecureWay® Security Server for OS/390 or by the RACF component of the OS/390 Security Server.

How to send your comments

Your feedback helps IBM to provide quality information. Please send any comments that you have about this book or other DB2 for OS/390 and z/OS documentation.

You can use any of the following methods to provide comments:

- Send your comments by e-mail to db2pubs@vnet.ibm.com and include the name of the product, the version number of the product, and the number of the book. If you are commenting on specific text, please list the location of the text (for example, a chapter and section title, page number, or a help topic title).
- Send your comments from the Web. Visit the Web site at:

<http://www.ibm.com/software/db2os390>

The Web site has a feedback page that you can use to send comments.

- Complete the readers' comment form at the back of the book and return it by mail, by fax (800-426-7773 for the United States and Canada), or by giving it to an IBM representative.

Chapter 1. Summary of changes to DB2 for OS/390 and z/OS Version 7

DB2 for OS/390 and z/OS Version 7 delivers an enhanced relational database server solution for OS/390. This release focuses on greater ease and flexibility in managing your data, better reliability, scalability, and availability, and better integration with the DB2 family.

In Version 7, some utility functions are available as optional products; you must separately order and purchase a license to such utilities. Discussion of utility functions in this publication is not intended to otherwise imply that you have a license to them. See Appendix B, “Changes to utilities”, on page 99 for more information about utilities products.

Enhancements for managing data

Version 7 delivers the following enhancements for managing data:

- DB2 now collects a comprehensive statistics history that:
 - Lets you track changes to the physical design of DB2 objects
 - Lets DB2 predict future space requirements for table spaces and indexes more accurately and run utilities to improve performance
- Database administrators can now manage DB2 objects more easily and no longer must maintain their utility jobs (even when new objects are added) by using enhancements that let them:
 - Dynamically create object lists from a pattern-matching expression
 - Dynamically allocate the data sets that are required to process those objects
- More flexible DBADM authority lets database administrators create views for other users.
- Enhancements to management of constraints let you specify a constraint at the time you create primary or unique keys. A new restriction on the DROP INDEX statement requires that you drop the primary key, unique key, or referential constraint before you drop the index that enforces a constraint.

Enhancements for reliability, scalability, and availability

Version 7 delivers the following enhancements for the reliability, scalability, and availability of your e-business:

- The DB2 Utilities Suite provides utilities for all of your data management tasks that are associated with the DB2 catalog.
- The new UNLOAD utility lets you unload data from a table space or an image copy data set. In most cases, the UNLOAD utility is faster than the DSNTIAUL sample program, especially when you activate partition parallelism for a large partitioned table space. UNLOAD is also easier to use than REORG UNLOAD EXTERNAL.
- The new COPYTOCOPY utility lets you make additional image copies from a primary image copy and registers those copies in the DB2 catalog. COPYTOCOPY leaves the target object in read/write access mode (UTRW), which allows Structured Query Language (SQL) statements and some utilities to run concurrently with the same target objects.

- Parallel LOAD with multiple inputs lets you easily load large amounts of data into partitioned table spaces for use in data warehouse applications or business intelligence applications. Parallel LOAD with multiple inputs runs in a single step, rather than in different jobs.
- A faster online REORG is achieved through the following enhancements:
 - Online REORG no longer renames data sets, which greatly reduces the time that data is unavailable during the SWITCH phase.
 - Additional parallel processing improves the elapsed time of the BUILD2 phase of REORG SHRLEVEL(CHANGE) or SHRLEVEL(REFERENCE).
- More concurrency with online LOAD RESUME is achieved by letting you give users read and write access to the data during LOAD processing so that you can load data concurrently with user transactions.
- More efficient processing for SQL queries:
 - More transformations of subqueries into a join for some UPDATE and DELETE statements
 - Fewer sort operations for queries that have an ORDER BY clause and WHERE clauses with predicates of the form *COL=constant*
 - More parallelism for IN-list index access, which can improve performance for queries involving IN-list index access
- The ability to change system parameters without stopping DB2 supports online transaction processing and e-business without interruption.
- Improved availability of user objects that are associated with failed or canceled transactions:
 - You can cancel a thread without performing rollback processing.
 - Some restrictions imposed by the restart function have been removed.
 - A NOBACKOUT option has been added to the CANCEL THREAD command.
- Improved availability of the DB2 subsystem when a log-read failure occurs: DB2 now provides a timely warning about failed log-read requests and the ability to retry the log read so that you can take corrective action and avoid a DB2 outage.
- Improved availability in the data sharing environment:
 - Group attachment enhancements let DB2 applications generically attach to a DB2 data sharing group.
 - A new LIGHT option of the START DB2 command lets you restart a DB2 data sharing member with a minimal storage footprint, and then terminate normally after DB2 frees the retained locks that it can.
 - You can let changes in structure size persist when you rebuild or reallocate a structure.
 - A new function in z/OS Version 1 Release 2 supports SCA and lock structure duplexing. As a result, a more robust failure recovery is possible in some data sharing environments.
- Additional data sharing enhancements include:
 - Notification of incomplete units of recovery
 - Use of a new OS/390 and z/OS function to improve failure recovery of group buffer pools
- An additional enhancement for e-business provides improved performance with preformatting for INSERT operations.

Easier development and integration of e-business applications

Version 7 provides the following enhancements, which let you more easily develop and integrate applications that access data from various DB2 operating systems and distributed environments:

- DB2 XML Extender for OS/390 and z/OS, a new member of the DB2 Extender family, lets you store, retrieve, and search XML documents in a DB2 database.
- Improved support for UNION and UNION ALL operators in a view definition, a nested table expression, or a subquery predicate, improves DB2 family compatibility and is consistent with SQL99 standards.
- More flexibility with SQL gives you greater compatibility with DB2 on other operating systems:
 - Scrollable cursors let you move forward, backward, or randomly through a result table or a result set. You can use scrollable cursors in any DB2 applications that do not use DB2 private protocol access.
 - A search condition in the WHERE clause can include a subquery in which the base object of both the subquery and the searched UPDATE or DELETE statement are the same.
 - A new SQL clause, FETCH FIRST *n* ROWS, improves performance of applications in a distributed environment.
 - Fast implicit close in which the DB2 server, during a distributed query, automatically closes the cursor when the application attempts to fetch beyond the last row.
 - Support for options USER and USING in a new authorization clause for CONNECT statements lets you easily port applications that are developed on the workstation to DB2 for OS/390 and z/OS. These options also let applications that run under WebSphere to reuse DB2 connections for different users and to enable DB2 for OS/390 and z/OS to check passwords.
 - For positioned updates, you can specify the FOR UPDATE clause of the cursor SELECT statement without a list of columns. As a result, all updatable columns of the table or view that is identified in the first FROM clause of the fullselect are included.
 - A new option of the SELECT statement, ORDER BY *expression*, lets you specify operators as the sort key for the result table of the SELECT statement.
 - New datetime ISO functions return the day of the week with Monday as day 1 and every week with seven days.
- Enhancements to Open Database Connectivity (ODBC) provide partial ODBC 3.0 support, including many new application programming interfaces (APIs), which increase application portability and alignment with industry standards.
- Enhancements to the LOAD utility let you load the output of an SQL SELECT statement directly into a table.
- A new component called Precompiler Services lets compiler writers modify their compilers to invoke Precompiler Services and produce an *SQL statement coprocessor*. An SQL statement coprocessor performs the same functions as the DB2 precompiler, but it performs those functions at compile time. If your compiler has an SQL statement coprocessor, you can eliminate the precompile step in your batch program preparation jobs for C, COBOL, and PL/I programs.
- Support for Unicode-encoded data lets you easily store multilingual data within the same table or on the same DB2 subsystem. The Unicode encoding scheme represents the code points of many different geographies and languages.

#

Improved connectivity

Version 7 offers improved connectivity:

- Support for COMMIT and ROLLBACK in stored procedures lets you commit or roll back an entire unit of work, including uncommitted changes that are made from the calling application before the stored procedure call is made.
- Support for Windows Kerberos security lets you more easily manage workstation clients who seek access to data and services from heterogeneous environments.
- Global transaction support for distributed applications lets independent DB2 agents participate in a global transaction that is coordinated by an XA-compliant transaction manager on a workstation or a gateway server (Microsoft Transaction Server or Encina, for example).
- Support for a DB2 Connect Version 7 enhancement lets remote workstation clients quickly determine the amount of time that DB2 takes to process a request (the server elapsed time).
- Additional enhancements include:
 - Support for connection pooling and transaction pooling for IBM DB2 Connect
 - Support for DB2 Call Level Interface (DB2 CLI) bookmarks on DB2 UDB for UNIX, Windows, OS/2

Features of DB2 for OS/390 and z/OS

Version 7 of DB2 UDB Server for OS/390 and z/OS offers several features that help you integrate, analyze, summarize, and share data across your enterprise:

- #
- #
- #
- DB2 Warehouse Manager feature. The DB2 Warehouse Manager feature brings together the tools to build, manage, govern, and access DB2 for OS/390 and z/OS-based data warehouses. The DB2 Warehouse Manager feature uses proven technologies with new enhancements that are not available in previous releases, including:
 - DB2 Warehouse Center, which includes:
 - DB2 Universal Database Version 7 Release 1 Enterprise Edition
 - Warehouse agents for UNIX, Windows, and OS/390
 - Information Catalog
 - QMF Version 7
 - QMF High Performance Option
 - QMF for Windows
- DB2 Management Clients Package. The elements of the DB2 Management Clients Package are:
 - DB2 Control Center
 - DB2 Stored Procedure Builder
 - DB2 Installer
 - DB2 Visual Explain
 - DB2 Estimator
- Net Search Extender for in-memory text search for e-business applications
- Net.Data for secure Web applications

See Chapter 6, “Features of DB2 for OS/390 and z/OS”, on page 49 for more information about these features.

Migration considerations

Migration with full fallback protection is available when you have either DB2 for
OS/390 Version 5 or Version 6 installed. You should ensure that you are fully
operational on DB2 for OS/390 Version 5, or later, before migrating to DB2 for
OS/390 and z/OS Version 7.

To learn about all of the migration considerations from Version 5 to Version 7, read the *DB2 Release Planning Guide* for Version 6 and Version 7; to learn about content information, also read appendixes A through F in both books.

Chapter 2. Managing data in DB2 for OS/390 and z/OS

Several new capabilities in DB2 for OS/390 and z/OS help you track changes, submit utility changes, manage authorizations, share data across your enterprise, and schedule maintenance. The following sections contain information about those new functions:

- “Comprehensive statistics history”
- “Utility lists with pattern matching and dynamic allocation”
- “More DBADM authority” on page 9
- “Enhanced management of constraints” on page 9
- “Real-time statistics” on page 10
- “Enhanced aggregation of statistics” on page 11

#

Comprehensive statistics history

The DB2 catalog now has a new table space (SYSHIST) and new tables to provide historical statistics. You can use the statistics to study trends, determine when to run IBM utilities for maintenance, determine when to run REORG, and manage space. The following tables have been added:

- SYSIBM.SYSCOLDIST_HIST
- SYSIBM.SYSCOLUMNS_HIST
- SYSIBM.SYSINDEXES_HIST
- SYSIBM.SYSINDEXPART_HIST
- SYSIBM.SYSINDEXSTATS_HIST
- SYSIBM.SYSLOBSTATS_HIST
- SYSIBM.SYSTABLEPART_HIST
- SYSIBM.SYSTABLES_HIST
- SYSIBM.SYSTABSTATS_HIST

SYSIBM.SYSTABLESPACE_HIST, for example, provides statistics for activity in SYSIBM.SYSTABLESPACE. SYSIBM.SYSTABLEPART_HIST provides statistics for activity in SYSIBM.SYSTABLEPART, and so on. When rows are added or changed in a table, information about the rows is written to the corresponding history table.

A new HISTORY keyword is available for the RUNSTATS utility and for the STATISTICS function with LOAD, REORG TABLESPACE, REORG INDEX, and REBUILD INDEX utilities. When the HISTORY keyword is specified, all inserts and updates that are made by DB2 to the catalog tables are recorded in the new catalog history tables. The new HISTORY keyword includes the following options:

- SPACE for only space statistics
- ACCESSPATH for only access-path statistics
- ALL for all inserts and updates
- NONE for no changes recorded in the history tables

In addition, a new utility, MODIFY STATISTICS, lets you delete unwanted statistics from the history tables. You can use the utility to delete statistical records, from the catalog history tables, that were written before a specific date or records of a specific age.

Utility lists with pattern matching and dynamic allocation

With Version 7, database administrators can manage DB2 objects more easily and no longer must maintain their utility jobs (even when new objects are added) by using enhancements that let them:

- Dynamically generate object lists explicitly or from a pattern-matching expression

- Dynamically allocate the data sets that are required to process those objects
- Dynamically alter the utility processing characteristics such as error handling and return codes

Using the LISTDEF utility control statement, you can standardize object lists and the utility control statements that reference them. Standardization reduces the need to customize and change utility job streams over time. See Figure 1 for an example of how you can define a list with pattern-matching expressions that you can use with two utilities, QUIESCE and COPY.

```

...
LISTDEF  PAYROLL INCLUDE TABLESPACE PAYROLL.*
          INCLUDE INDEXSPACE PAYROLL.*IX
          EXCLUDE TABLESPACE PAYROLL.TEMP*
          EXCLUDE INDEXSPACE PAYROLL.TMPIX*
QUIESCE LIST PAYROLL WRITE YES
COPY LIST PAYROLL ...

```

Figure 1. Example of a LISTDEF utility control statement that defines a list of objects.

The TEMPLATE utility control statement simplifies job control language (JCL) and supports data set allocation during the processing of LISTDEF lists in the absence of data definition (DD) cards. In its simplest form, the TEMPLATE utility control statement defines the data set naming convention. Optionally the control statement can also contain allocation parameters that define data set size, location, and attributes.

Traditional database administration includes querying the catalog to identify objects to be processed. Once the database administrator identified objects, utility jobs then had to be altered in order to specify the correct utility control statements, DD names, and JCL DD cards. DB2 for OS/390 and z/OS Version 7 lets you group database objects into reusable lists and to dynamically allocate the utility data sets that are required to process that list.

These Version 7 enhancements are most useful for installations in which data base objects have rigid naming conventions that can be used to group objects for administration purposes, and in which data set naming conventions are adopted. By significantly reducing the time needed to perform these core utility jobs, the Version 7 enhancements can reduce the cost of administering databases.

LOAD utility enhancements

You can use any DRDA-compliant server as a data input source for the LOAD utility
 # to populate your tables in DB2 for OS/390 and z/OS. You can input data from
 # non-DB2 data sources by using DataJoiner for access to other data sources. This
 # enhancement delivers a DB2 family cross loader function. Use the EXEC SQL utility
 # control statement to declare a cursor. Then use the INCURSOR option of LOAD
 # utility to use the cursor to load the data.

You can load the output from any SELECT statement directly into a table on DB2
 # for OS/390 and z/OS. Use the EXEC SQL utility control statement to execute
 # dynamic SQL statements. Because the SELECT statement can access any DRDA
 # server, the data source can be any member of the DB2 family, DataJoiner, or any
 # other vendor that supports DRDA server capabilities. This extension to the LOAD
 # utility provides synergy between the power of the LOAD utility and the connectivity
 # function and reliability of DRDA.

More DBADM authority

DBADM authority is expanded to give more flexibility to database administrators. A new subsystem parameter, DBACRVW, is associated with these changes on installation panel DSNTIPP. For more information, see Chapter 7, “Planning for migration and fallback”, on page 69.

The expanded authority is described in the following sections:

- “CREATE ALIAS authority for DBADM”
- “CREATE VIEW authority for DBADM”

CREATE ALIAS authority for DBADM

Previously, users with DBADM authority could not create aliases for others and needed to hold SYSCTRL authority for this task. Now, DBADM has the authority to create aliases for other authorization IDs on any table in the database if the DBADM CREATE VIEW field on installation panel DNSTIPP is YES.

CREATE VIEW authority for DBADM

Previously, users with DBADM authority could not create views for others and needed to hold SYSADM authority for this task. Now, DBADM has the authority to create views for other authorization IDs.

The authorization ID of the view owner can be different from the authorization ID that created the view if all of the following conditions are true:

- The user who created the view had DBADM authority on at least one database that contains a table on which the view is based.
- The value of the DBADM CREATE VIEW field on installation panel DSNTIPP is YES.
- The view includes at least one table and is not based only on other views.

Existing requirements for creating views remain unchanged. For example, if the view includes user-defined functions, the view owner must have EXECUTE authority on the user-defined functions.

Enhanced management of constraints

An enhancement for helping to manage constraints restricts the dropping of an index that is enforcing a primary key, unique key, or referential constraint and provides a means to specify a constraint name for the primary key or unique key constraint.

Enhancements have been introduced for the way primary key, unique key, foreign key, and check constraints are created, maintained, and deleted.

In prior releases, constraints were not all treated in a consistent manner. For example:

- Only foreign key and check constraints could be named.
- The number of unique key constraints that could be created was limited to an arbitrary number.
- Unique key constraints could not be added or dropped.
- Constraint clauses had different syntax.
- Dropping an index that enforces a unique key implicitly dropped the unique key definition.

- Primary key and unique key constraint details were not stored explicitly in the catalog tables.
- Indexes enforcing constraints could be dropped, causing table definitions to be incomplete and inaccessible.
- Foreign keys could reference a parent key that was not defined as a primary key or unique key, as long as a unique index existed on the parent key.

In Version 7, the following enhancements have been made to treat constraints more consistently and to make constraint management easier:

- All constraints can be named.
- The number of unique key constraints that can be created is not limited.
- Unique key constraints can be added or dropped with the ADD or DROP UNIQUE clause of the ALTER TABLE statement.
- Consistent syntax is used for specifying the constraints.
- The catalog explicitly stores primary key and unique key information in new catalog tables SYSIBM.SYSTABCONST and SYSIBM.SYSKEYCOLUSE.
- All cases for which the table definition can be incomplete are flagged in the TABLESTATUS column of SYSIBM.SYSTABLES.
- Indexes that enforce constraints can no longer be dropped because the constraint must be dropped first.
- Foreign keys must now reference a parent key that has been defined as a primary key or unique key.

Real-time statistics

With Version 7, DB2 provides the structure that enables automation of utility job
 # scheduling. Specifically, DB2 provides the necessary statistics that end users or
 # automated task schedulers can use to determine the objects for which REORG,
 # RUNSTATS or COPY need to be run.

DB2 collects the statistics in real time and periodically writes them to DB2 tables.
 # Applications can query the tables to obtain the statistics. Some of the statistics that
 # DB2 collects are:

- # • The number of rows, LOBs or index entries that have been modified since the
 # last REORG, RUNSTATS or COPY
- # • Physical space information, such as the number of preformatted pages, the
 # amount of allocated space, and the number of extents used
- # • The number of distinct pages that have been updated since the last COPY
- # • The time of the first update since the last COPY

To help you see how real-time statistics can be used, DB2 provides sample
 # real-time statistics stored procedure DSNACCOR. DSNACCOR queries the
 # real-time statistics tables and recommends when you need run REORG,
 # RUNSTATS, or COPY.

For detailed information on real-time statistics, see Appendix G (Volume 2) of *DB2*
 # *Administration Guide*. For detailed information on the real-time statistics stored
 # procedure, see Appendix H (Volume 2) of *DB2 Administration Guide*.

Enhanced aggregation of statistics

Prior to DB2 Version 7, aggregation of statistics could be performed only if all
partitions contained data. With DB2 Version 7, aggregation of statistics can be
performed even if some partitions contain no data. This is done by the addition of
the STATISTICS ROLLUP statement to the **Operator function panel: DSNTIPO**
and the FORCEROLLUP keyword to the RUNSTATS utility.

Specify STATISTICS ROLLUP as YES on the **Operator function panel: DSNTIPO**
to aggregate the partition level statistics. The default is NO.

In the RUNSTATS utility, specifying YES or NO for the FORCEROLLUP keyword
also selects whether aggregate statistics are preformed.

Chapter 3. DB2 for your e-business—reliable, scalable, and available

Many businesses already use DB2 for their e-business. They depend on the reliability, scalability, and availability of DB2 to serve their worldwide customers around the clock, seven days a week. Version 7 offers many improvements for electronic commerce. If you have not yet deployed your business on the Web, now is the ideal time to start. A number of enhancements in Version 7 of DB2 for OS/390 help you gain competitive advantage:

- # “Utility support for DB2”
- “A new UNLOAD utility” on page 14
- “A new COPYTOCOPY utility” on page 15
- “Parallel LOAD with multiple inputs” on page 15
- # “DB2 resynchronizes BSDS data sets during restart” on page 16
- “Faster online REORG” on page 17
- “More concurrency with online LOAD RESUME” on page 17
- “More efficient processing for SQL queries” on page 17
- “Changing subsystem parameters without stopping DB2” on page 19
- “Improved availability of some user objects” on page 20
- “Allow retry of log-read request” on page 20
- # “Data sharing enhancements” on page 20
- “Better monitoring of the distributed environment” on page 22
- “Additional enhancement for e-business” on page 22

Utility support for DB2

DB2 for OS/390 and z/OS Version 7 delivers the all DB2 utilities to support your catalog (DSNDB06) and directory (DSNDB01) data. DB2 catalog and directory data is critical for all user databases and applications. To sustain the high-availability of DB2 for OS/390 and z/OS, the DB2 utilities are delivered with each shipment of DB2 for OS/390 and z/OS Version 7. Use these powerful management utilities for all of your DB2 catalog and directory data management tasks:

- CATMAINT
- DIAGNOSE
- LISTDEF
- OPTIONS
- QUIESCE
- REPAIR
- REPORT
- TEMPLATE
- All standalone utilities

DB2 Version 7 offers other utilities for use on user data in various packages. Those utilities are available in three separate products:

- 5655-E63: DB2 Operational Utilities (FMID JDB771K)
 - COPY
 - EXEC SQL
 - LOAD
 - REBUILD INDEX
 - RECOVER
 - REORG INDEX
 - REORG TABLESPACE
 - RUNSTATS
 - STOSPACE

- UNLOAD
- 5655-E62: DB2 Diagnostic and Recovery Utilities (FMID JDB771M)
 - CHECK DATA
 - CHECK INDEX
 - CHECK LOB
 - COPY
 - COPYTOCOPY
 - MERGECOPY
 - MODIFY RECOVERY
 - MODIFY STATISTICS
 - REBUILD INDEX
 - RECOVER
- 5697-E98: DB2 Utilities Suite (FMIDs JDB771K and JDB771M)
 - CHECK DATA
 - CHECK INDEX
 - CHECK LOB
 - COPY
 - COPYTOCOPY
 - EXEC SQL
 - LOAD
 - MERGECOPY
 - MODIFY RECOVERY
 - MODIFY STATISTICS
 - REBUILD INDEX
 - RECOVER
 - REORG INDEX
 - REORG TABLESPACE
 - RUNSTATS
 - STOSPACE
 - UNLOAD

Important

In this version of DB2 for OS/390 and z/OS, some utility functions are available as optional products. You must separately order and purchase a license to such utilities, and discussion of those utility functions in this publication is not intended to otherwise imply that you have a license to them.

A new UNLOAD utility

You can unload data from a table space or an image copy data set by using the new UNLOAD utility. In most cases, the UNLOAD utility is faster than the DSNTIAUL sample program and REORG UNLOAD EXTERNAL, especially when you activate partition parallelism. UNLOAD is easier to use and has more options than REORG UNLOAD EXTERNAL. Figure 2 on page 15 shows how you can use UNLOAD to perform the following tasks:

- Unload data from an image copy data set.
- Unload data from multiple partitions in parallel.
- Select data by using a syntax similar to the SQL SELECT statement.
- Select sample rows by table.
- Use field selection, ordering, and formatting options.
- Specify SHRLEVEL CHANGE or REFERENCE.

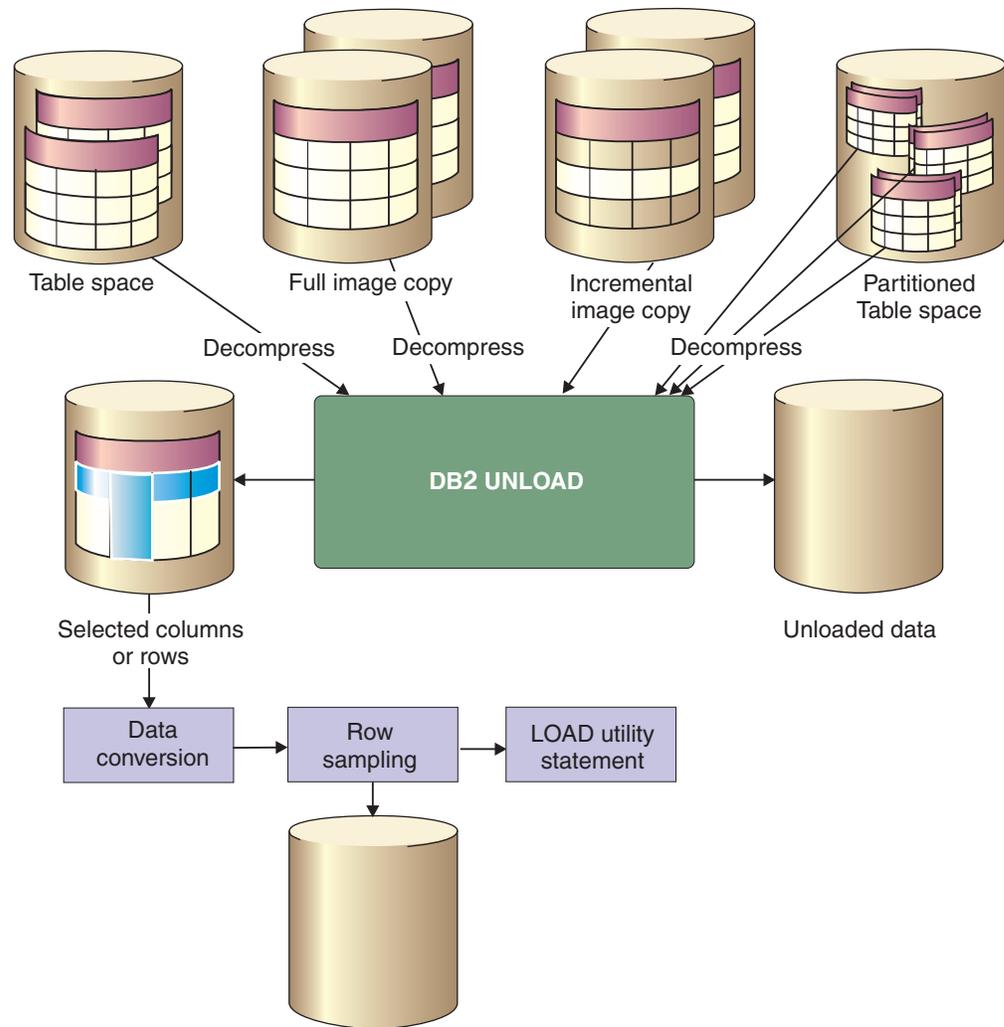


Figure 2. Unloading data using the new DB2 UNLOAD utility

A new COPYTOCOPY utility

While the COPY, LOAD, and REORG utilities can make two local and two remote site backup copies of data, you might want to make a single copy and then clone that copy at a more convenient time. The COPYTOCOPY utility asynchronously makes up to three additional backup copies from an existing copy and registers the copies in the DB2 catalog for recovery purposes. You can use object wildcarding and dynamic allocation capabilities with the COPYTOCOPY utility.

Parallel LOAD with multiple inputs

Using Version 7, you can easily load large amounts of data into partitioned table spaces for use in data warehouse or business intelligence applications. Parallel load with multiple inputs runs in a single step, rather than in different jobs.

The LOAD utility loads each partition from a separate data set so that one job can load multiple partitions in parallel. Parallel loading reduces the elapsed time for

loading the data as compared to loading the same data with a single job in earlier releases. If a table space has non-partitioned indexes(NPIs), using parallel load with multiple inputs is faster than using separate jobs for each partition. Using load parallelism is much easier than creating multiple load jobs for individual parts.

Figure 3 shows a parallel load of four partitions with the SORTKEYS keyword enabling a parallel index build of three indexes. Each load task takes input from a sequential data set and loads the data into a corresponding partition. The utility then extracts index keys and passes them in parallel to the sort task that is responsible for sorting the keys for that index. If the amount of data is too large to perform the sort in memory, the sort product writes the keys to the sort work data sets. The sort tasks pass the sorted keys to their corresponding build task, each of which builds one index. If the utility encounters errors during the load, DB2 writes error and error mapping information to the error and map data sets.

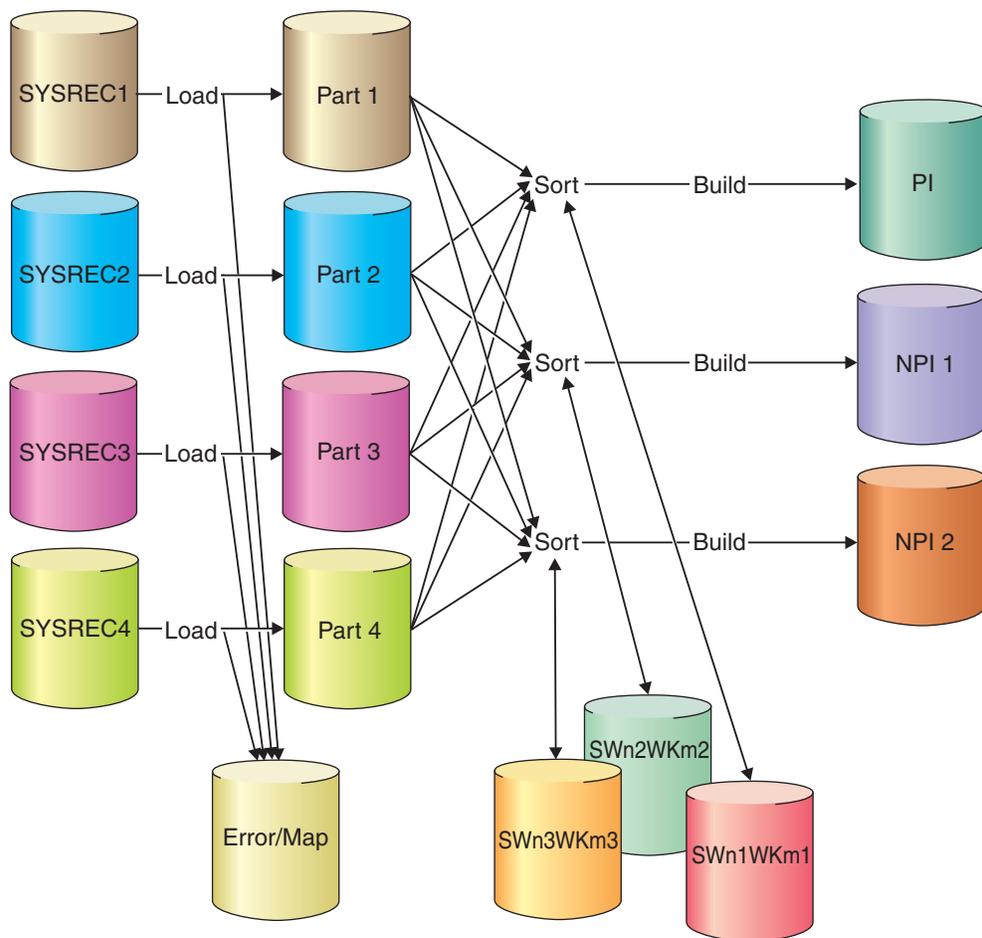


Figure 3. Parallel LOAD with multiple inputs and parallel index build

DB2 resynchronizes BSDS data sets during restart

Automatic resynchronization of the BSDS data sets during restart improves
 # recovery time. No longer does DB2 terminate when you restart DB2 in single BSDS
 # mode. Instead, when you restart DB2 in single BSDS mode, DB2 attempts to use
 # the BSDS with the latest timestamp to restore dual BSDS mode. Only after DB2
 # fails to automatically restore dual BSDS mode must you manually resynchronize the
 # BSDS data sets to restart DB2.

Faster online REORG

Online REORG enhancements improve the availability of data and keeps your business applications available longer in two ways:

- Online REORG no longer renames data sets, greatly reducing the time that data is unavailable during the SWITCH phase. You specify a new keyword, FASTSWITCH, which keeps the data set name unchanged and updates the catalog to reference the newly reorganized data set. The time savings can be quite significant when DB2 is reorganizing hundreds of table spaces and index objects.
- Additional parallel processing improves the elapsed time of the BUILD2 phase of REORG SHRLEVEL(CHANGE) or SHRLEVEL(REFERENCE).

REORG periodically needs to drain objects being reorganized. Three additional
keywords improve your ability to specify how long REORG waits for these drains.
The DRAIN_WAIT option is the aggregate time that online REORG waits to perform
a drain on a table space and associated indexes. By specifying a delay time that is
shorter than the system timeout value, you can reduce time-outs. Use the RETRY
and RETRY_DELAY options to further improve the success rate of the online
REORG. For more details on using these REORG options, see *DB2 Utility Guide*
and Reference.

More concurrency with online LOAD RESUME

Use online LOAD RESUME to improve your data availability. With prior releases of DB2, access to data during LOAD processing was prohibited. With Version 7, you choose whether users have read and write access to data during LOAD processing. Online LOAD RESUME with the option SHRLEVEL CHANGE loads data concurrently with user transactions and with minimal impact.

With prior releases of DB2, loading begins at the current end of data in the table space when records are loaded into a non-empty table space. When the freepage limit is reached, one free page is created. Online LOAD RESUME YES SHRLEVEL CHANGE places new data in the available free page as close to clustering order as possible.

More efficient processing for SQL queries

DB2 continues to improve its ability to execute queries quickly. Processing enhancements include:

- “More transformations of subqueries into join”
- “Fewer sorts for ORDER BY clauses” on page 18
- “More parallelism for IN-list index access” on page 19

More transformations of subqueries into join

Previously, for SELECT statements, DB2 could sometimes transform a subquery into a join between the result table of the subquery and the result table of an outer query. Now, DB2 can sometimes transform a subquery into a join for UPDATE and DELETE statements to significantly improve the performance of those statements, too. For example, DB2 can use a join for the following statement:

```
UPDATE T1 A SET C1 = 1
WHERE A.C2 IN (SELECT B.C2
FROM T2 B
WHERE A.C3 = B.C3);
```

DB2 considers a transformation only when the predicate that contains the subquery is IN, =ANY, =SOME (SELECT statement only), or EXISTS. (EXISTS support for SELECT is new in Version 7.) DB2 does the transformation only if certain conditions are true, for example:

- The transformation does not introduce redundancy.
- The subquery is correlated to its immediate outer query.
- The FROM clause of the subquery contains only one table, and the outer query (for SELECT), UPDATE, or DELETE references only one table.
- If the outer predicate is a quantified predicate with an operator of =ANY or an IN predicate, the following conditions are true.
 - The left side of the outer predicate is a single column.
 - The right side of the outer predicate is a subquery that references a single column.
 - The two columns have the same data type and length.
- The subquery does not contain the GROUP BY or DISTINCT clauses.
- The subquery does not contain column functions.
- The SELECT clause of the subquery does not contain a user-defined function with an external action or a user-defined function that modifies data.
- The subquery predicate is a Boolean term predicate.
- The predicates in the subquery that provide correlation are stage 1 predicates.
- The subquery does not contain nested subqueries.
- The subquery does not contain a self-referencing UPDATE or DELETE.
- For a SELECT statement, the query does not contain the FOR UPDATE clause.
- For an UPDATE or DELETE statement, the statement is a searched UPDATE or DELETE.
- For a SELECT statement, parallelism is not enabled.

For a complete list of the requirements, see *DB2 Application Programming and SQL Guide* or *DB2 Administration Guide*.

Fewer sorts for ORDER BY clauses

DB2 can now perform fewer sort operations for queries that have an ORDER BY clause. Previously, when a query had a WHERE clause with a predicate in the form of COL=*constant*, DB2 performed a sort when the column was included in the ORDER BY clause or an index key. Now, when a column has such a predicate, DB2 can consider it as a constant in the result table. As a constant, the predicate does not affect ordering; DB2 can remove it from the ORDER BY list or index key, which might avoid a sort.

For example, the following two queries provide the same results. Removing C2 from the ORDER BY list in the second query does not change the order of the results:

```
SELECT C1, C2, C3
FROM T1
WHERE C2 = 5
ORDER BY C1, C2, C3;
```

```
SELECT C1, C2, C3
FROM T1
WHERE C2 = 5
ORDER BY C1, C3;
```

The following example shows how DB2 can use an index that was not previously recognized as a useful mechanism to avoid a sort:

```

SELECT C1, C2, C3, C4
  FROM T1
 WHERE C2 = 5
        AND C4 = 7
        AND C5 = 2
 ORDER BY C1, C2, C3, C4

```

Index I1 on (C2, C1, C5, C4, C3)

In the example, DB2 removes C2 and C4 from the ORDER BY list, which leaves ordering columns C1 and C3. Similarly, C2, C4, and C5 are logically removed from index I1, which leaves ordering columns C1 and C3. With the columns removed, it is easy to see that index I1 supports the ordering that is required by the ORDER BY clause.

More parallelism for IN-list index access

Parallelism is now fully supported for queries that involve IN-list index access. Previously, these queries needed to run sequentially, although parallelism could be used when the IN-list access was for the inner table of a parallel group. Now, in environments in which parallelism is enabled, you can see a reduction in elapsed time for queries that involve IN-list index access for the outer table of a parallel group.

For example, the following query, which involves access to a single table, can now execute in parallel:

```

SELECT C1
  FROM T1
 WHERE C1 IN (1,4,2,7,5)
 ORDER BY C1;

```

Index I1 on (C1)

Changing subsystem parameters without stopping DB2

Previously, updating subsystem parameters required stopping and starting DB2 to refresh the subsystem parameter load module (which has a default of DSNZPARM). Now, availability is improved. You can change the values of many subsystem parameters while DB2 is running by issuing the SET SYSPARM command.

DB2 keeps track of the load modules so that you can back out changes. You can load the values that DB2 loaded at startup or values from a different load module. The following three options help you manage your changes:

- Issue SET SYSPARM LOAD(*module-name*) to load a new subsystem parameter load module.
- Issue SET SYSPARM RELOAD to reload the last-named subsystem parameter load module.
- Issue SET SYSPARM STARTUP to reset the subsystem parameter load module with the one that DB2 loaded at startup.

To update the subsystem parameters on a subsystem, follow these steps:

1. Run through the installation process in Update mode.
2. Produce a new subsystem parameter load module.
3. Issue the SET SYSPARM command and load the new load module.

Improved availability of some user objects

Version 7 gives you more flexibility to control the availability of your objects that are associated with failed or canceled transactions. You can cancel a thread and specify that no rollback processing is to be performed. DB2 removes some restrictions that are imposed by the RESTART function and adds a NOBACKOUT option of the CANCEL THREAD command.

Allow retry of log-read request

Sometimes a temporary problem in accessing log data sets can terminate DB2. These problems can result from attempts to allocate or open archive log data sets during the rollback of a long-running unit of recovery, a temporary problem with HSM recalls or tape subsystems, archives that are inadvertently uncataloged, or archive tape mounts that the operator cancels. DB2 attempts to retrieve the requested log records from all copies of the archive log, but terminates when all attempts to access the copies fail for a *must-complete* operation, such as a rollback.

Now, reliability is improved. You can attempt to recover from this type of failure and retry the failing log-read request instead of allowing DB2 to terminate. Correct whatever problem is causing the log read error and then respond with a Y to message DSNJ154I.

Data sharing enhancements

DB2 for OS/390 continues to improve the availability, flexibility, and performance of its Parallel Sysplex support in Version 7.

Group attachment enhancements

Before Version 7, the DB2 group attachment name could be used by DB2 applications to generically attach to a DB2 data sharing group. The group attachment capability is essential when a job or a user logon can be dynamically routed to any system in the Sysplex.

Version 7 adds more functionality to group attachment by allowing:

- Use of the *startecb* parameter when using the group attachment name to connect to DB2.
- Support for DL/I batch environments to allow for dynamic routing of IMS batch jobs across the Sysplex.
- A NOGROUP connection option, for applications that use CAF and RRSAF to connect to DB2, which indicates that the *ssnn* specified in the connect call should always be treated as a subsystem name, even though this name may also match the group attachment name.

Restart light

The new LIGHT(YES) parameter of the START DB2 command lets you restart a DB2 member in *light mode*. Restart-light mode means that a DB2 data sharing member restarts with a minimal storage footprint and then terminates normally after DB2 frees retained locks. DB2 releases all retained locks with the following exceptions:

- Locks that are held by indoubt units of recovery.
- Locks that are held by postponed-abort units of recovery.

- IX mode pageset P-locks. These locks do not block access from other DB2 members; however, they do block drainers, such as utilities.

Restart-light mode is intended for a cross-system restart in the event of an MVS system failure. The reduced storage requirement makes it possible to temporarily restart a DB2 data sharing member on a system that might not have enough resources to start and stop DB2 in normal mode. Releasing the locks with a minimum of disruption promotes faster recovery and data availability. For example, applications that are running on other DB2 members have quicker access to the data for which the failed member held incompatible locks.

You can also use restart-light mode in conjunction with the MVS Automatic Restart Manager (ARM). To have a DB2 data sharing member automatically restarted in light mode when system failure occurs, you must have an appropriately coded ARM policy. The ARM policy for the DB2 group must specify LIGHT(YES) within the RESTART_METHOD(SYSTEM) keyword for the DB2 element name. For example:

```
RESTART_METHOD(SYSTEM,STC,'cmdprfx STA DB2,LIGHT(YES)')
```

If ARM is enabled for IRLM, the ARM policy for the IRLM element name should specify PC=YES within the RESTART_METHOD(SYSTEM) keyword for the IRLM element name. PC=YES allows IRLM to obtain the full benefits of restart-light mode by having IRLM put its locks in extended private storage instead of ECSA. For example:

```
RESTART_METHOD(SYSTEM,STC,'cmdprfx S irlmproc,PC=YES')
```

ARM does not restart the DB2 member again after a light restart is performed; the member terminates normally for the light restart.

Persistent structure size changes

When a new coupling facility structure is allocated for a DB2 data sharing group, its size is usually taken from the value of the INITSIZE parameter in the CFRM policy. After the structure is allocated, you can dynamically change its size with the SETXCF START,ALTER,SIZE=*newsiz*,STRNAME=*strname* command. Dynamically changing the size of a structure does not change the INITSIZE value of the policy.

In prior releases of DB2, dynamic changes to structure sizes with the SETXCF START,ALTER command for the group buffer pool (the cache structure) were lost when DB2 shut down or the structure was rebuilt. DB2 is enhanced to allow changes made to structure sizes through the SETXCF START,ALTER command to persist across a recycle of DB2 and across a rebuild of the structure. With this enhancement, DB2 now retains and, in most cases uses, the changed size for subsequent allocations, instead of using the INITSIZE in the CFRM policy. Subsequent allocations in which the changed value is retained and used include:

- A group buffer pool or SCA that is deallocated and then reallocated
- The allocation of the secondary structure of a duplexed group buffer pool if duplexing is started *after* the size of a primary structure has been dynamically changed
- Any structure that is rebuilt with the SETXCF START,REBUILD command

The changed size is used for those subsequent allocations until one of the following events occurs:

- A CFRM policy is started, and the policy has a different INITSIZE than what was in effect when the size of the structure was dynamically changed with the SETXCF START,ALTER command.

- Another SETXCF START,ALTER command is issued to dynamically change the size of the structure.

Exception: If a lock structure is deallocated and all the members are down, the INITSIZE value is used. This is a consideration for disaster recovery or situations where data sharing groups are cloned. During normal operation of a data sharing group, this situation is unlikely to occur.

Additional data sharing enhancements

Other data sharing enhancements include:

- Notification of incomplete units of recovery. DB2 issues new message DSNR046I during normal shutdown of a DB2 member when locks are retained because of incomplete units of recovery. This message, which requires no operator response, allows you to choose to immediately restart the DB2 member to resolve the incomplete units of recovery and remove the retained locks. Removing the locks enables other DB2 members to access the affected data.
- New OS/390 function improves group buffer pool failure recovery. With the appropriate levels of OS/390 and coupling facility control code (CFCC), DB2 can ignore unnecessary events that are requested by OS/390 during coupling facility (CF) failures and CF link failures.
- New z/OS function provides more robust failure recovery. With z/OS Version 1 Release 2 or later, you can enable the operating system to duplex the SCA and lock structures. When you enable duplexing, z/OS maintains a synchronized copy of the structures during normal operation. In the case of a primary structure failure, z/OS automatically switches to the secondary structure. See *DB2 Data Sharing: Planning and Administration* for information about the implications of enabling duplexing for the SCA and lock structures.

#

Better monitoring of the distributed environment

You can use the DISPLAY DDF command to gain information about the status of
DDF. For example, you can find out if DDF is stopped, started, or suspended.

You can use the DETAIL keyword to obtain information such as the maximum
number of DBATs allowed, the maximum number of connections allowed, or the
number of queued threads.

Additional enhancement for e-business

The following additional enhancements provide improved performance for your e-business applications:

- Preformatting for insert operations
DB2 improves the performance of insert operations by asynchronously preformatting allocated but unformatted data pages. When a new page is used for an insert, that page is close to the end of the formatted pages, and allocated but unformatted space is available in the data set, DB2 preformats the next range of pages. With preformatting, an insert waits less often for a page to be formatted. When the preformatted space is used and DB2 needs to extend the table space, normal data set extending and preformatting occurs.
- Parallel access to multiple partitions in a table space
DB2 improves performance when multiple jobs access different partitions in a table space concurrently. In previous releases of DB2, when several applications

#

accessed different partitions of the same table space at the same time, DB2
opened the corresponding data sets serially. With this improvement, DB2 opens
the data sets in parallel.

• Identification of the origin of SQL statements in the dynamic SQL statement
cache

If your application programs use RRSAF to connect to DB2, DB2 can provide
information about which application program issued the PREPARE statement that
put an SQL statement in the dynamic statement cache. This information makes it
easier to identify the application program that needs editing to fix an SQL
statement that is performing poorly.

Chapter 4. Easier development and integration of e-business applications

Recent releases of DB2 for OS/390 deliver function that helps you access large data stores, develop and maintain client-server applications, and call stored procedures from popular desktop tools. Version 7 lets you more easily develop and integrate applications that access data in many environments. The following sections contain more information about those application enhancements:

“DB2 for OS/390 and z/OS and the multitier environment”

“DB2 XML Extender for OS/390 and z/OS” on page 29

“More flexibility with SQL” on page 30

“ODBC enhancements” on page 38

“SQLJ and JDBC enhancements” on page 39

“LOAD utility enhancements” on page 40

“Precompiler services” on page 40

“Unicode support for international data and e-business” on page 42

DB2 for OS/390 and z/OS and the multitier environment

This section provides an overview of DB2 for OS/390 and z/OS in a multitier environment. It describes the synergy between DB2 and Web applications and application servers that include WebSphere, Net.Data, and DB2 Connect.

For more information about DB2 and the Web and DB2 in a multitier environment, see *An Introduction to DB2 for OS/390*.

Three-tier architecture

Many businesses choose to use DB2 as their database server for the Web. The three-tier architecture is ideal for many e-businesses. In a three-tier architecture:

- The client is on the first tier (as it is in a two-tier architecture). The client handles user interactions; it does not contain any business logic. A second, or middle, tier supplements the client.
- One or more application servers reside on the middle tier. The application server handles the portion of the business logic that does not require the functionality that the database server provides.
- On the third tier is the database server, DB2 for OS/390 and z/OS, that provides data for the application. All Web-based database applications rely on a database server. For performance reasons, the database server typically uses stored procedures to handle some of the business logic.

The three primary components of Web-based database applications are the Web browser (or client), the Web application server, and the database server. The client handles the presentation logic and, in some cases, validates user-provided input. Web applications sometimes integrate Java applets or JavaScript into the client-side logic to improve the presentation layer. Hardware and software components of the second and third tiers share responsibility for the availability, scalability, and performance characteristics of the Web site.

In many cases, the client and server for a Web application are on different operating systems. The client, for example, can be on a workstation-based operating system, such as Windows 2000. The server for the application can also be on a workstation-based server or on an enterprise server, such as OS/390.

Figure 4 illustrates the use of a three-tier architecture. In this example, the Web server on the second tier is a Windows 2000 machine. WebSphere or Net.Data can use DB2 Connect to access DB2 for OS/390 and z/OS. You will read more about WebSphere, Net.Data, and DB2 Connect later in this section.

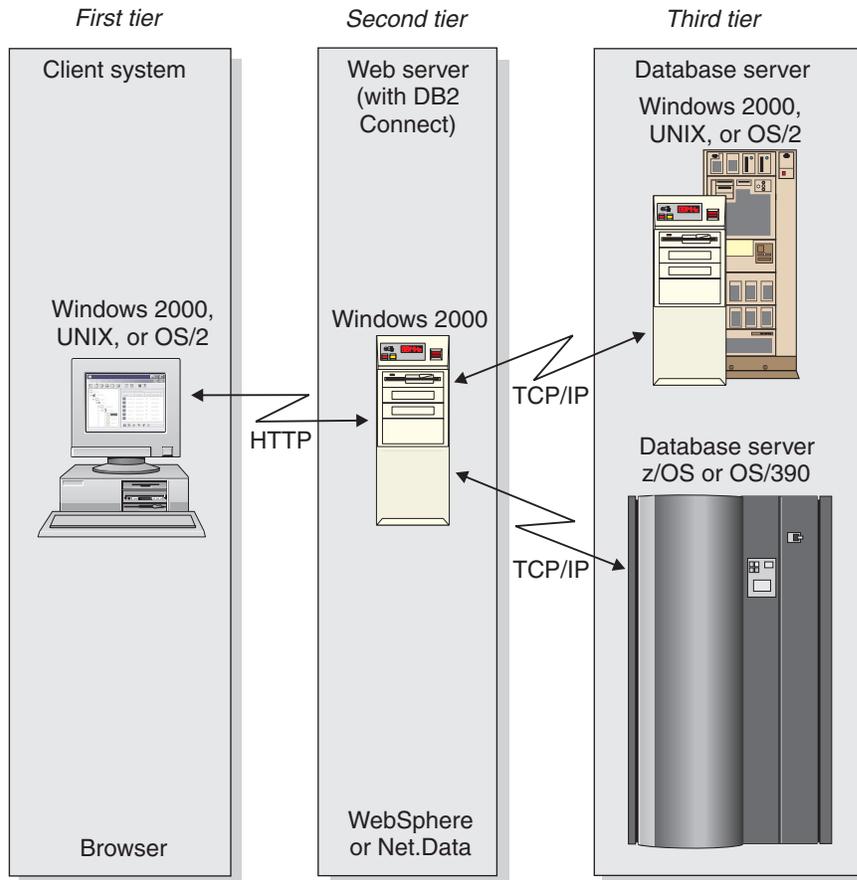


Figure 4. Three-tier connectivity with a workstation-based client and Web server, and with different database servers

DB2 for OS/390 and z/OS offers a robust solution for Web applications. Specifically, using DB2 for OS/390 and z/OS as a database server for a Web application provides the following advantages:

- **Tremendous scalability.** The volume of transactions on any Web application varies. Transaction loads increase, or spike, at different times of the day, on different days of the month, or at different times of the year. Transaction loads also tend to increase over time, as more users move to the Web. In a Parallel Sysplex environment, DB2 for OS/390 and z/OS can handle the full range of transaction loads with little or no impact on performance. Any individual user is generally unaware of how busy the system is at a given point in time.
- **High degree of availability.** When DB2 for OS/390 and z/OS runs in a Parallel Sysplex environment, the availability of data and applications is very high. If one DB2 subsystem is unavailable because of maintenance, for example, other DB2 subsystems in the Sysplex take over the workload. Users are unaware that part of the system is unavailable because they have access to the data and applications that they need.

- **Ability to manage a mixed workload.** DB2 for OS/390 and z/OS effectively and efficiently manages priorities of a mixed workload as a result of its tight integration with OS/390 Workload Manager (WLM).
- **Protection of data integrity.** Users of DB2 for OS/390 and z/OS can benefit from the product's well-known strength in the areas of security and reliability.

Application servers on the Web

IBM provides two application servers, WebSphere and Net.Data, that help companies "Web-enable" their data and business logic. WebSphere and Net.Data products run on popular operating systems, including AIX, Linux, OS/390, OS/400, Windows 2000, Windows NT, and Solaris Operating Environment.

WebSphere

The WebSphere family of products offers users:

- A Java server run-time environment that is based on industry standards
- Web-site development tools
- Commerce software
- Management software

By using these tools, companies can build, deploy, and manage portable e-business applications. Information about WebSphere is available at:

www.ibm.com/software/webservers

Net.Data

The Net.Data family of products provides a robust environment for both application development and execution. Companies can use Net.Data products to create high-performance Web applications by using a variety of programming languages, such as Java, REXX, Perl, and C++.

See "Net.Data for secure Web applications" on page 61 for more information about Net.Data.

Database access with DB2 Connect

DB2 Connect gives applications fast and easy access to existing databases on IBM enterprise servers. It has a highly scalable communication infrastructure for connecting Web, Windows, UNIX (including Linux), OS/2 and mobile applications to S/390 and AS/400 data. For detailed information about DB2 Connect, see *DB2 Connect User's Guide*.

DB2 Connect provides extensive application programming tools for developing client-server and Web applications using industry standard APIs such as ODBC, JDBC and SQLJ.

DB2 Connect Version 7 includes the following features that enhance its database access capabilities:

- Improved scalability and performance with connection concentration
 Connection concentration, also referred to as transaction pooling, benefits applications that acquire a connection and retain the connection for long periods of time. Connection concentrator technology concentrates the workload from all applications in fewer S/390 host or AS/400 database server connections. The connection concentrator provides:
 - Automatic thread reuse following commit and rollback points
 - Delayed thread reuse when open WITH HOLD cursor is detected
 - Intact SQL special registers across transaction boundaries

- Improved load-balance and fail-over support for DB2 for OS/390 and z/OS servers in Parallel Sysplex environments
- Distributed request capability that allows working with all DB2 family server in the same SQL statement and access to Oracle databases with the option of DB2 Relational Connect
- New function in support of DB2 for OS/390 and z/OS Version 7 that includes:
 - Scrollable cursors, described in “Scrollable cursors” on page 31.
 - FETCH FIRST *n* ROWS ONLY, described in “The FETCH FIRST *n* ROWS ONLY clause” on page 35.
 - Single sign-on capability using Kerberos and user ID and password encryption, described in “Support for Windows Kerberos security” on page 46.
- Improved transactional capabilities with support for tightly-coupled X/Open XA transactions such as those required by BEA Tuxedo and support for Microsoft SNA Server in two-phase commit applications

Transaction manager support for multisite updates

In a *multisite update*, data is updated in more than one database within a single transaction. A distributed transaction typically involves multiple recoverable resources, such as DB2 tables, MQSeries messages, and IMS databases.

A transaction manager coordinates commit operations across a distributed transaction. DB2 supports a wide range of transaction manager products. Application environments that use DB2 Connect to access DB2 remotely can use the following transaction manager products:

- Transaction manager products that support standard XA protocols
- IBM WebSphere
- BEA Tuxedo
- BEA WebLogic
- Microsoft Transaction Server (MTS)
- IBM TxSeries (CICS and Encina)
- Java applications that support Java Transaction API (JTA) and Enterprise JavaBeans (EJBs)
- Message and Queuing Series (MQSeries)

The XA interface is a bidirectional interface between a transaction manager and resource managers that provides coordinated updates across a distributed transaction. In this environment, DB2 Connect, DDF, and local attachment facilities (RRSAF, CICS, IMS) now recognize database connections that are part of a global transaction. As “Global transaction support for distributed applications” on page 47 explains, locks are shared across the branches of a distributed global transaction.

Example: An application server initiates a global transaction and uses EJBs to invoke applications at other application servers. These other application servers then invoke additional applications at other servers as part of the global transaction.

When this type of application runs, the middleware, such as DB2 Connect, identifies the various branches of the global transaction so that DB2 (as a resource manager) understands the relationships between the database connections. DB2 automatically allows the branches of a global transaction within a given DB2 subsystem to share locks, so that one branch does not lock out other branches of the same global transaction.

DB2 XML Extender for OS/390 and z/OS

DB2 XML Extender for OS/390 and z/OS is a new member of the DB2 Extender family. You can use DB2 XML Extender for OS/390 and z/OS to store, retrieve, and search XML documents in a DB2 database. DB2 XML Extender for OS/390 and z/OS supplies the data types, functions, and stored procedures to let you manage XML documents.

You can use DB2 XML Extender for OS/390 and z/OS to:

- Store XML documents as whole documents in columns
- Compose XML documents from a collection of data that is stored in multiple columns and tables
- Decompose an XML document back into data that is stored in columns and tables based on the previously defined mapping

An application server can send the XML documents over the Internet to other sites.

XML is the standard for data interchange for the next generation of electronic business-to-business and business-integration solutions. Use interchange formats that are based on XML to leverage your critical business information in DB2 databases in business-to-business solutions. When you store, retrieve, and search XML documents in a DB2 database, you benefit from the unmatched performance, reliability, and scalability of DB2 for OS/390 and z/OS. With the DB2 XML Extender for OS/390 and z/OS, you can integrate Internet applications that are based on XML documents with your existing DB2 database.

Storing, transmitting, and searching XML documents

DB2 XML Extender for OS/390 and z/OS supports storing XML documents in DB2 databases in two different ways—collections and columns:

Collections: You can store XML documents as untagged and traditional data items in multiple columns and tables in a DB2 database. You can also create XML documents from your existing data, as Figure 5 shows.

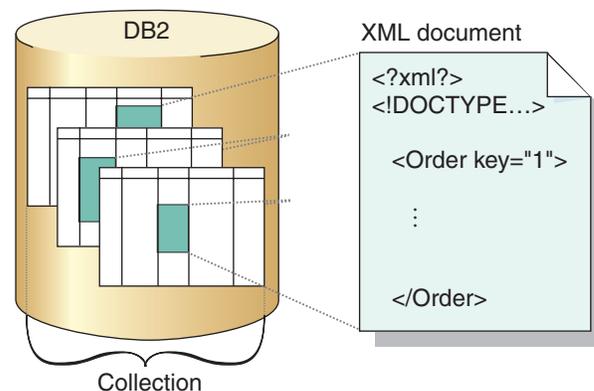


Figure 5. An XML document that is stored as a collection of traditional data items in multiple columns and tables in DB2

You define a mapping of the document elements to the data in the database. You can then use stored procedures from DB2 XML Extender for OS/390 and z/OS to compose XML documents from data that is stored in DB2 tables, based on the previously defined mapping. You can send the resulting XML document over the Internet by an application server to a corresponding application server

at another site. You can also receive XML documents and decompose them into data that is stored in the DB2 tables based on the data mappings.

Columns: You can store XML documents in a DB2 database as a whole document in a single column using an XML distinct type, as Figure 6 shows.

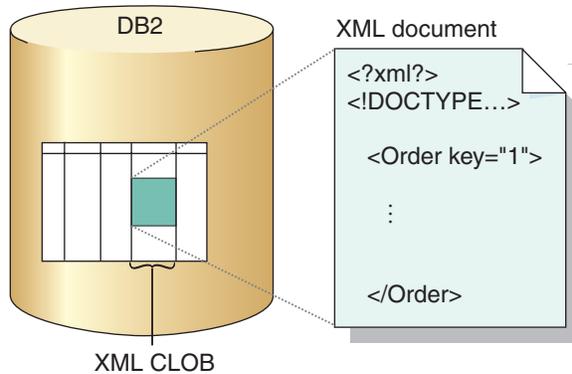


Figure 6. An XML document that is stored in a single column of a DB2 table with a new XML data type.

You use SQL to do fast and powerful searches on XML distinct types just as you would search other distinct types. You can automatically extract elements or attributes into traditional SQL data types. Then you can use the sophisticated indexing and SQL query capabilities of DB2 to search the data. In addition, the IBM DB2 Text Extender for OS/390 and z/OS supports structured documents such as XML. You can apply the powerful text search functions in DB2 to a section or list of sections within a set of XML documents. This can significantly improve the effectiveness of the search.

Graphical support for DB2 XML Extender for OS/390 and z/OS

The DB2 XML Extender for OS/390 and z/OS provides a wizard to easily perform administration tasks. DB2 XML Extender for OS/390 and z/OS helps you enable your database for XML documents, store document type definitions (DTDs), and map XML elements and attributes to DB2 tables and columns.

More flexibility with SQL

You have more choices when using SQL and greater compatibility with DB2 on other operating systems.

Support for UNION and UNION ALL operators

The scope in which UNION and UNION ALL operators can be specified has been expanded. The CREATE VIEW statement, the INSERT statement, the UPDATE statement, the DECLARE GLOBAL TEMPORARY TABLE, nested table expressions in a FROM clause, and the subquery predicate are changed to allow a fullselect where a subselect was used in previous releases.

Now, you create a view by using UNION or UNION ALL to view the data as if it were in one table. If you use UNION ALL to combine the tables, the result consists of all the rows in the tables. If you use UNION, the result is the set of all the rows in the tables without duplicate rows. Whenever possible, the following optimizations are applied to the queries referencing such views, table expressions, or subqueries:

- The joins in the queries are distributed to the subselects of the UNION ALL.

- The predicates in the queries are distributed to the subselects of the UNION ALL.
- Aggregation in the queries is distributed to the subselects of UNION ALL.
- Subselects that are not needed to answer that the queries are eliminated.

The following example illustrates creation of a view that is the UNION ALL of three fullselects, one for each month of the first quarter of 2000. The common names for the views are SNO, CHARGES, and DATE.

```
CREATE VIEW DSN8710.FIRSTQTR (SNO, CHARGES, DATE) AS
SELECT SNO, CHARGES, DATE
FROM MONTH1
WHERE DATE BETWEEN '01/01/2000' and '01/31/2000'
UNION ALL
SELECT SNO, CHARGES, DATE
FROM MONTH2
WHERE DATE BETWEEN '02/01/2000' and '02/29/2000'
UNION ALL
SELECT SNO, CHARGES, DATE
FROM MONTH3
WHERE DATE BETWEEN '03/01/2000' and '03/31/2000';
```

You can use the INSERT statement in the same way you use fullselects. The UPDATE statement is also changed to support row-fullselect and scalar-fullselect where row-select and scalar-subselect were previously supported in the SET assignment clause. In the DECLARE GLOBAL TEMPORARY TABLE statement, AS (subselect) DEFINITION ONLY is changed to AS (fullselect) DEFINITION ONLY. You now can use fullselect with a basic predicate, quantified predicate, EXISTS predicate, and IN predicate.

Scrollable cursors

When you retrieve a result table from a DB2 table or a result set from a stored procedure, you use a cursor to point to specific rows. In previous releases of DB2, the cursor could only start at the beginning of the result table or result set and move sequentially forward. In DB2 Version 7, you can use *scrollable cursors* to move forward, backward, or randomly through a result table or result set.

Scrollable cursors have an advantage, even if you always want to use them scroll forward sequentially. A scrollable cursor can be updatable and have an ORDER BY clause. Therefore, if you want the result table of a cursor to be in a specified order, and you also want to use the cursor for update operations, you need to declare the cursor as scrollable.

Using a scrollable cursor

To make a cursor scrollable, you declare it as scrollable. To use a scrollable cursor, you execute FETCH statements that indicate where you want to position the cursor.

Declaring a scrollable cursor: To indicate that a cursor is scrollable, you declare it with the SCROLL keyword. The following examples show declarations for two basic forms of a scrollable cursor:

```
EXEC SQL DECLARE C1 INSENSITIVE SCROLL CURSOR FOR
SELECT DEPTNO, DEPTNAME, MGRNO
FROM DSN8710.DEPT
ORDER BY DEPTNO
END-EXEC.
```

Figure 7. Declaration for an insensitive scrollable cursor

```
EXEC SQL DECLARE C2 SENSITIVE STATIC SCROLL CURSOR FOR
  SELECT DEPTNO, DEPTNAME, MGRNO
  FROM DSN8710.DEPT
  ORDER BY DEPTNO
END-EXEC.
```

Figure 8. Declaration for a sensitive scrollable cursor

Both of these examples show a characteristic of scrollable cursors: the *sensitivity*.

The first example shows a declaration for an *insensitive* scrollable cursor. Declaring a cursor with the `INSENSITIVE` keyword has the following effects:

- The size, the order of the rows, and the values for each row of the result table do not change after you open the cursor.
- The result table is read-only. Therefore, you cannot declare the cursor with the `FOR UPDATE` clause, and you cannot use the cursor for update or delete operations.

The second example shows a declaration for an *sensitive* scrollable cursor. Declaring a cursor as `SENSITIVE` has the following effects:

- When you execute positioned `UPDATE` and `DELETE` statements with the cursor, those updates are visible in the result table.
- When the current value of a row no longer satisfies the `SELECT` statement for the cursor, that row is no longer visible in the result table.
- When a row of the result table is deleted from the underlying table, the row is no longer visible in the result table.
- Changes that are made to the underlying table by other cursors or other application processes can be visible in the result table, depending on whether the `FETCH` statements that you use with the cursor are `FETCH INSENSITIVE` or `FETCH SENSITIVE` statements.

In DB2 Version 7, when you declare a cursor as `SENSITIVE`, you must also declare it as `STATIC`. Declaring the cursor as `STATIC` has the following effects:

- The size of the result table does not grow after you open the cursor.
Rows that are inserted into the underlying table are not added to the result table.
- The order of the rows does not change after you open the cursor.

If the cursor declaration contains an `ORDER BY` clause, and columns that are in the `ORDER BY` clause are updated after you open the cursor, the order of rows in the result table does not change.

Retrieving rows with a scrollable cursor: When you open any cursor, the cursor is positioned before the first row of the result table. You move a scrollable cursor around in the result table by specifying a *fetch orientation* keyword in a `FETCH` statement. A fetch orientation keyword indicates the absolute or relative position of the cursor when the `FETCH` statement is executed. Table 1 lists the fetch orientation keywords that you can specify and their meanings.

Table 1. Positions for a scrollable cursor

Keyword in <code>FETCH</code> statement	Cursor position when the <code>FETCH</code> is executed
BEFORE	Before the first row
FIRST	At the first row
LAST or ABSOLUTE -1	At the last row

Table 1. Positions for a scrollable cursor (continued)

Keyword in FETCH statement	Cursor position when the FETCH is executed
AFTER	After the last row
ABSOLUTE	To an absolute row number. +1 is the first row.
RELATIVE	Forward or backward a relative number of rows
CURRENT	At the current row
PRIOR or RELATIVE -1	To the previous row
NEXT	To the next row (default)

For example, to use the cursor that is declared in Figure 7 on page 31 to fetch the fifth row of the result table, use a FETCH statement like this:

```
EXEC SQL FETCH ABSOLUTE +5 C1 INTO :HVDEPTNO, :DEPTNAME, :MGRNO;
```

When you declare a cursor as SENSITIVE, changes that other processes or cursors make to the underlying table *can be* visible to the result table of your cursor. Whether those changes *are* visible depends on whether you specify the SENSITIVE or INSENSITIVE keyword when you execute FETCH statements with the cursor. When you specify FETCH INSENSITIVE, changes that other processes or cursors make to the underlying table are not visible in the result table. When you specify FETCH SENSITIVE, changes that other processes or cursors make to the underlying table are visible in the result table. Table 2 summarizes the sensitivity values and their effects on the result table of a scrollable cursor.

Table 2. How sensitivity affects the result table for a scrollable cursor

DECLARE sensitivity	FETCH sensitivity	
	INSENSITIVE	SENSITIVE
INSENSITIVE	No changes to the underlying table are visible in the result table. Positioned UPDATE and DELETE statements using the cursor are not allowed.	Not valid.
SENSITIVE	Only changes that are made by the cursor are visible in the result table.	All changes are visible in the result table.

Performance considerations for using scrollable cursors

The following recommendations help you get the best performance from your scrollable cursors:

- Determine when scrollable cursors work best for you.
Scrollable cursors are a valuable tool for writing applications such as screen-based applications, in which the result table is small and you often move back and forth through the data. However, scrollable cursors require more DB2 processing than non-scrollable cursors. If your applications require large result tables or you only need to move sequentially forward through the data, use non-scrollable cursors.
- Declare scrollable cursors as SENSITIVE only if you need to see the latest data. If you do not need to see updates that are made by other cursors or application processes, using a cursor that you declare as INSENSITIVE requires less processing by DB2.

- To ensure maximum concurrency when you use a scrollable cursor for positioned update and delete operations, specify ISOLATION(CS) and CURRENTDATA(NO) when you bind packages and plans that contain updatable scrollable cursors.

- Use the FETCH FIRST *n* ROWS ONLY clause with scrollable cursors when it is appropriate.

In a distributed environment, when you need to retrieve a limited number of rows, FETCH FIRST *n* ROWS ONLY can improve your performance for distributed queries that use DRDA access by eliminating unneeded network traffic. See “The FETCH FIRST *n* ROWS ONLY clause” on page 35 for more information.

In a local environment, if you need to scroll through a limited subset of rows in a table, you can use FETCH FIRST *n* ROWS ONLY to make the result table smaller.

- In a distributed environment, if you do not need to use your scrollable cursors to modify data, do your cursor processing in a stored procedure.

Using stored procedures can decrease the amount of network traffic that your application requires.

- Create TEMP table spaces that are large enough to process your scrollable cursors.
- Remember to commit changes often.

Because you often leave scrollable cursors open longer than non-scrollable cursors, it is important to commit changes often enough. Declare your scrollable cursors WITH HOLD to prevent the cursors from closing after a commit operation.

Wrapping of identity column values

DB2 for OS/390 now supports wrapping of identity column values. For previous
releases, DB2 issues an error when the identity column values are exhausted, and
it is not possible to insert new rows into the table. To enable wrapping, the AS
IDENTITY clause in the CREATE TABLE and ALTER TABLE statements has the
following new parameters:

- CYCLE specifies whether the identity column should continue to generate values after reaching either the maximum or minimum value of the sequence. NO CYCLE is the default.
- MINVALUE specifies the minimum value that is generated for the column. This value can be any positive or negative value that can be assigned to the column. The default value is the START WITH value, or 1 if START WITH is not specified, for an ascending sequence; or the minimum value for the column data type for a descending sequence.
- MAXVALUE specifies the maximum value that is generated for the column. This value can be any positive or negative value that can be assigned to the column. The default value is the START WITH value, or -1 if START WITH is not specified, for a descending sequence; or the maximum value for the column data type for an ascending sequence.

Suppose you create the following table with MINVALUE = -9 and START WITH = 1:

```
# CREATE TABLE T1
# (C1 DECIMAL(1,0) GENERATED ALWAYS AS IDENTITY
# (CYCLE, MINVALUE -9),
# C2 CHAR(10));
```

As identity column values are generated, column C1 has the following values for
the cycles:

```

#           -9 -8 -7 -6 -5 -4 -3 -2 -1 0 1 2 3 4 5 6 7 8 9
#           -----> Cycle 1
#           -----> Cycle 2
#           -----> Cycle 3
#           ...

```

When wrapping is in effect, duplicate values for a column are allowed even when the column is GENERATED ALWAYS, unless a unique index is defined on the column. If there is a unique index on the column and all possible values have been used, wrapping is possible only if rows are uploaded or deleted before wrapping occurs.

Self-referencing subselect on UPDATE or DELETE

For a searched UPDATE or DELETE statement, you can now allow the table that is being modified to be referenced in the WHERE clause of the statement. The following example shows giving a 10-percent increase to employees whose salary is below the average salary for the department:

```

#           UPDATE EMP X SET SALARY = SALARY * 1.10
#           WHERE SALARY < (SELECT AVG(SALARY) FROM EMP Y
#           WHERE X.WORKDEPT = Y.WORKDEPT);

```

In addition, for a searched UPDATE statement, if the table being modified is used in a subquery in the SET clause, you can include a correlated reference in the select list of the subquery. The following example shows raising the salary of employees in a department whose salary is below average to the average salary:

```

#           EXEC SQL UPDATE EMP T1
#           SET SALARY = (SELECT AVG(T2.SALARY) FROM EMP T2)
#           WHERE WORKDEPT = 'E11' AND
#           SALARY < (SELECT AVG(T3.SALARY) FROM EMP T3);

```

For a positioned UPDATE statement, you can also allow a table that is being modified to be referenced in a subquery in the SET clause of the statement. The following example shows giving the employees in a department a bonus equal to 10% of their salary:

```

#           EXEC SQL
#           DECLARE C1 CURSOR FOR
#           SELECT BONUS
#           FROM DSN8710.EMP
#           WHERE WORKDEPT = 'E12'
#           FOR UPDATE OF BONUS;
#
#           EXEC SQL
#           UPDATE DSN8710.EMP
#           SET BONUS = (SELECT .10 * SALARY FROM DSN8710.EMP Y
#           WHERE EMPNO = Y.EMPNO)
#           WHERE CURRENT OF C1;

```

However, for a positioned UPDATE statement, you cannot include a correlated reference to the column being updated in the select list of the subquery. That is, considering the preceding example, you cannot include a correlated reference to the BONUS column.

The FETCH FIRST *n* ROWS ONLY clause

A new SELECT statement clause can improve performance of your distributed applications and limit the result set size for local applications.

For a distributed query, you can use the FETCH FIRST *n* ROWS ONLY clause to limit the number of rows that DB2 prefetches and returns. When you execute a

SELECT statement that contains the FETCH FIRST n ROWS ONLY clause, DB2 prefetches only n rows. This feature can improve performance when the result table of a query is large, but you are interested in only a few rows.

For a local query, the FETCH FIRST n ROWS ONLY clause limits the size of the result table to n rows. When you use FETCH FIRST n ROWS ONLY with scrollable cursors, you can improve performance by limiting the number of rows that the cursor scrolls through.

If you specify FETCH FIRST n ROWS ONLY in a SELECT statement, OPTIMIZE FOR n ROWS is implied. If OPTIMIZE FOR m rows is specified, DB2 optimizes for the lower of n or m .

You can also specify the FETCH FIRST ROW ONLY clause in a SELECT INTO statement when you know that you want to retrieve only one row. The FETCH FIRST ROW ONLY clause in a SELECT INTO statement prevents potential SQL errors that occur when the result table contains more than one row.

Example: FETCH FIRST n ROWS ONLY: In a DRDA environment, suppose that the following SQL statement causes DB2 to prefetch 16 rows of the result table:

```
SELECT * FROM EMP
  OPTIMIZE FOR 1 ROW;
```

Now suppose that you need only one row of the result table. To avoid 15 unnecessary prefetches, add the FETCH FIRST 1 ROW ONLY clause:

```
SELECT * FROM EMP
  OPTIMIZE FOR 1 ROW
  FETCH FIRST 1 ROW ONLY;
```

Because FETCH FIRST 1 ROW ONLY implies OPTIMIZE FOR 1 ROW, you can simplify the query to this one:

```
SELECT * FROM EMP
  FETCH FIRST 1 ROW ONLY;
```

Fast implicit close

Fast implicit close means that during a distributed query, the DB2 server automatically closes the cursor after it prefetches the n th row if you specify FETCH FIRST n ROWS ONLY, or when there are no more rows to return. Fast implicit close can improve performance because it saves an additional network transmission between the client and the server.

DB2 uses fast implicit close when the following conditions are true:

- The query uses limited block fetch.
- The query retrieves no LOBs.
- The cursor is not a scrollable cursor.
- Either of the following conditions is true:
 - The cursor is declared WITH HOLD, and the package or plan that contains the cursor is bound with the KEEP DYNAMIC(YES) option.
 - The cursor is not defined WITH HOLD.

When you use FETCH FIRST n ROWS ONLY, and DB2 does a fast implicit close, the DB2 server closes the cursor after it prefetches the n th row, or when there are no more rows to return.

Even though DB2 closes the cursor, you should still include a CLOSE statement in your applications. The CLOSE statement returns SQLCODE 0 even if DB2 does a

fast implicit close. In addition, including a CLOSE statement in your application ensures that the cursor is closed, even if the query does not meet the conditions for a fast implicit close.

Support for USER and USING keywords on the CONNECT statement

The new options USER and USING in a new authorization clause for CONNECT statements allow you to provide a user ID and password for remote servers or for a local DB2. The password is used to verify that the user ID is authorized to connect. USER *host-variable* identifies the user ID, and USING *host-variable* identifies the password of the user ID. The *host-variable* can have a length attribute not greater than 255 characters, must be a character string variable, and must not include an indicator variable. For connecting to a local DB2, neither the user ID or the password can be greater than 8 characters.

The options allow you to port your applications that are developed on the workstation to DB2 for OS/390, allow applications running under WebSphere, for example, to reuse DB2 connections for different users, and enable DB2 for OS/390 to check passwords. Authorization cannot be specified when the connecting type is IMS or CICS.

FOR UPDATE enhancement

For positioned updates, the FOR UPDATE clause of the SELECT statement of the cursor can be specified without a list of columns. When no columns are specified, all updatable columns of the table or view that is identified in the first FROM clause of the fullselect are included.

ORDER BY *expression* of the SELECT statement

The new option ORDER BY *expression* allows you to specify an expression with operators as the sort key for the result table of a SELECT statement (that is, not simply a column name or integer). The query to which ordering is applied must be a subselect to use this form of the sort key for the ORDER BY clause.

IN and OUT parameters of an SQL procedure

To enhance compatibility with DB2 on other operating systems, DB2 modifies the behavior of IN and OUT parameters within an SQL procedure. You can place an IN parameter on the left or right side of an assignment statement. When control returns to the caller, the original value of an IN parameter is passed to the caller. You can also place an OUT parameter on the left or right side of an assignment statement. When control returns to the caller, the last value that is assigned to an OUT parameter is returned to the caller.

SQL scalar functions

DB2 introduces SQL scalar functions, which provide a fast and easy way to write simple user-defined functions. You write code for an SQL scalar function when you define it, which eliminates the need to write and prepare a host-language program. In addition, because the source code for an SQL scalar function is stored in the DB2 catalog, an SQL scalar function performs better than an external user-defined function that provides the same function.

You define an SQL scalar function using a CREATE FUNCTION statement and alter the definition using an ALTER FUNCTION statement. The source code for an SQL function consists of a RETURN clause that contains a single SQL expression. For more information on SQL scalar functions, see Chapter 5 of *DB2 SQL Reference*.

To make DB2 applications more compatible with other database applications,
 # untyped parameters are now permitted. For a function invocation that contains
 # untyped parameter markers, the data types of those parameter markers are
 # considered to match or be promotable to the data types of the parameters of the
 # function instance.

Alternative datetime functions that use ISO rules

New functions have been added that return the day of the week with Monday as day 1 and every week with seven days. The DAYOFWEEK_ISO function returns an integer between 1 and 7 that represents the day of the week with 1 representing Monday and 7 representing Sunday.

The WEEK_ISO function returns the week of the year as an integer value in the range of 1 to 53. The week starts with Monday and always includes seven days. Week 1 is the first week of the year to contain a Thursday. Thus, it is possible that up to three days at the beginning of the year appear as the last week of the previous year or up to three days at the end of the year appear in the first week of the next year.

Compatibility for C long data type

The *long* data type is not defined consistently for the C compilers that are used on
 # these DB2 hardware platforms:

- # • On 64-bit UNIX hardware platforms, *long* is a 64-bit integer.
- # • On 64-bit Windows operating systems and 32-bit hardware platforms, *long* is a
 # 32-bit integer.

For C compilers for all of these, *sqlint32* is defined as a 32-bit integer. To enhance
 # compatibility within the DB2 family, the DB2 on OS/390 SQL precompiler now
 # recognizes *sqlint32* as a synonym for *long*.

ODBC enhancements

DB2 for OS/390 and z/OS provides ODBC 3.0 support in Version 7. The addition of many ODBC 3.0 functions aligns DB2 ODBC with industry standards and increases application portability. Table 3 lists the new functions.

Table 3. ODBC 3.0 function support

Function name	Purpose
SQLAllocHandle()	Obtains an environment, connection, or statement handle
SQLCloseCursor()	Closes a cursor that has been opened on the statement handle.
SQLColAttribute()	Returns descriptor information for a column in the result set.
SQLEndTran()	Commits or rolls back a transaction.
SQLFreeHandle()	Frees an environment, connection, or statement handle.
SQLGetConnectAttr()	Returns a value of a connection attribute.
SQLGetDiagRec()	Returns additional diagnostic information.
SQLGetStmtAttr()	Returns a value of a statement attribute.
SQLSetConnectAttr()	Sets a value of a connection attribute.
SQLSetStmtAttr()	Sets a value of a statement attribute.

In addition, the following functions support ODBC 3.0 argument values:

- SQLGetEnvAttr()
- SQLGetInfo()
- SQLSetEnvAttr()

See *DB2 ODBC Guide and Reference* for more information.

SQLJ and JDBC enhancements

DB2 provides the following enhancements to JDBC and SQLJ support:

#

- # Interpreted Java stored procedures and user-defined functions
- # You can now create Java stored procedures and user-defined functions that run in a Java Virtual Machine, as specified in Part 1 of the ANSI standard, Database Languages – SQL – Part 10: Object Language Bindings (SQL/OLB). These routines can contain JDBC calls, SQLJ statements, or a combination of both. You can create a JAR file that contains all the methods for your routine, install that JAR file in DB2, and then run the routine from the JAR file. Java routines have special run-time requirements, so they need to run in a WLM-established stored procedure address space that is tailored for Java routines. For more information about interpreted Java stored procedures, see *DB2 Application Programming Guide and Reference for Java*.
- # JDBC/SQLJ driver for OS/390 with JDBC 2.0 support
- # This version of the JDBC/SQLJ driver for OS/390 is fully compliant with the JDBC 1.2 and SQLJ – Part 0 specification, and includes many of the functions of the JDBC 2.0 specification.
- # To use this version of the JDBC/SQLJ driver for OS/390, you need the SDK for OS/390, Version 1.3 or higher.
- # You need this driver if you use any of the following functions:
- # – The JDBC 2.0 DataSource function
- # – Global transactions that run under WebSphere Application Server Version 4.0 and above

#

The JDBC/SQLJ driver for OS/390 with JDBC 1.2 support is still available. You select the version of the JDBC/SQLJ driver for OS/390 that you want to use by specifying the associated file name in your CLASSPATH environment variable.

- # Data source and connection pooling support
- # The DB2 for OS/390 and z/OS JDBC/SQLJ driver for OS/390 with JDBC 2.0 support includes JDBC 2.0 data source (DataSource) support, Java Naming and Directory Interface (JNDI) support, and connection pooling support. When you connect to a data source using the JDBC 2.0 data source support, your application can reference a data source using a logical name, rather than an explicit driver class name and URL. In addition, you can define or modify the data source attributes without changing the JDBC application program.
- # DataSource support is a complete replacement for the previous JDBC driver manager support. You can use both types of support in the same application, but it is recommended that you use data source support to obtain connections, regardless of whether you use connection pooling or distributed transactions.
- # With DataSource support, an application uses JNDI to associate a logical name with a specific data source implementation. This DataSource object contains all of the information that is necessary to determine the correct JDBC driver and return a `java.sql.Connection` object to the specified data source.
- # Connection pooling is part of DataSource support. Connection pooling is a framework for caching physical data source connections, which are equivalent to

DB2 threads. When JDBC reuses physical data source connections, the
expensive operations that are required for the creation and subsequent closing of
`java.sql.Connection` objects are minimized. Connection pooling is a built-in part
of JDBC 2.0 data source support. Connection pooling support is completely
transparent to the JDBC application.

The JDBC 2.0 connection pooling framework lets a single physical data source
connection be serially reused by logical `java.sql.Connection` instances. The
application can use the logical `java.sql.Connection` object in exactly the same
manner as it uses a `java.sql.Connection` object when there is no connection
pooling support.

- # • Global transaction support

JDBC and SQLJ global transaction support lets Enterprise Java Beans (EJB) and
Java servlets that run under WebSphere Application Server Version 4.0 or later
access DB2 for OS/390 and z/OS relational data within global transactions.
WebSphere Application Server provides the environment to deploy EJBs and
servlets, and RRS provides the transaction management.

With global transactions, you do not execute the `commit` or `rollback` methods on
a `Connection` object.. You let WebSphere manage when transactions begin and
end. Alternatively, you can use specialized Java Transaction API (JTA) interfaces
to indicate the boundaries of transactions. Although DB2 for OS/390 and z/OS
does not implement the JTA specification, the methods for delimiting transaction
boundaries are available.

LOAD utility enhancements

Enhancements to the LOAD utility let you load the output of an SQL SELECT statement directly into a table. First, you specify a cursor for the input data set in the LOAD utility; then you declare the cursor and specify the dynamic SQL statement in the EXEC SQL control statement.

See *DB2 Utility Guide and Reference* for more information on the LOAD enhancement and EXEC SQL statement.

Precompiler services

DB2 Version 7 includes a new component called Precompiler Services. Compiler writers can modify their compilers to invoke Precompiler Services, and thereby produce an *SQL statement coprocessor*. An SQL statement coprocessor performs the same functions as the DB2 precompiler, but it performs those functions at compile time. If your compiler has an SQL statement coprocessor, you can eliminate the precompile step in your batch program preparation jobs.

Using an SQL statement coprocessor has the following advantages over using the DB2 precompiler:

- An SQL statement coprocessor removes restrictions on DB2 host variables.
The DB2 precompiler does not allow you to fully qualify the names of structured host variables. With an SQL statement coprocessor, a structured host language variable can be fully qualified.
- An SQL statement coprocessor makes debugging a program easier.
The DB2 precompiler translates your original program into modified source code. Source language debuggers use that modified source code as input. However, when the compiler includes an SQL statement coprocessor, you can use your original source code as input to the debugger.

- The format of the source file no longer has special restrictions other than the restrictions of the compiler.

The DB2 precompiler places restrictions on the source file. For example, the record length of the source file can be no more than 80 bytes. When you use an SQL statement coprocessor, the compiler controls the source file format.

Using the C SQL statement coprocessor

IBM Enterprise C for z/OS and OS/390 Version 1 Release 2 or later has an SQL statement coprocessor, which performs DB2 precompiler functions at compile time. The DB2 precompiler has some restrictions that the SQL statement coprocessor does not have. When you process SQL statements with the SQL statement coprocessor, you can do the following things in your program:

- # • Use fully-qualified names for structured host variables.
- # • Include SQL statements at any level of a nested C program, instead of in only the top-level source file.
- # • Use nested SQL INCLUDE statements.

To process SQL statements as you compile your C program, you specify the SQL compiler option, followed by a list of precompiler options in parentheses. For example, suppose that you want to process SQL statements as you compile a C program. In your program, the apostrophe is the string delimiter in SQL statements, and the SQL statements conform to DB2 rules. This means that you need to specify the APOSTSQL and STDSQL(NO) precompiler options. Therefore, you need to include the following compiler option in your compile step:

```
# SQL("APOSTSQL STDSQL(NO)")
```

Using the COBOL SQL statement coprocessor

IBM COBOL for OS/390 & VM Version 2 Release 2 has an SQL statement coprocessor, which performs DB2 precompiler functions at compile time. The DB2 precompiler has some restrictions that the SQL statement coprocessor does not have. When you process SQL statements with the SQL coprocessor, you can do the following things in your program:

- Use fully-qualified names for structured host variables.
- Include SQL statements at any level of a nested COBOL program, instead of in only the top-level source file.
- Use nested SQL INCLUDE statements.
- Use COBOL REPLACE statements to replace text strings in SQL statements.

To process SQL statements as you compile your COBOL program, you specify the SQL compiler option, followed by a list of precompiler options in parentheses. For example, suppose that you want to process SQL statements as you compile a COBOL program. In your program, the apostrophe is the string delimiter in SQL statements, and the SQL statements conform to DB2 rules. This means that you need to specify the APOSTSQL and STDSQL(NO) precompiler options. Therefore, you need to include the following compiler option in your compile step:

```
SQL("APOSTSQL STDSQL(NO)")
```

Controlling the CCSID: IBM Enterprise COBOL for z/OS & OS/390 Version 3 Release 2 or later, and the SQL statement coprocessor for the COBOL compiler, support:

- # • The NATIONAL data type that is used for declaring Unicode values in the UTF-16 format (that is, CCSID 1200)

• The COBOL CODEPAGE compiler option that is used to specify the default
EBCDIC CCSID of character data items

You can use the NATIONAL data type and the CODEPAGE compiler option to
control the CCSID of the character host variables in your application.

Using the PL/I SQL statement coprocessor

IBM Enterprise PL/I for z/OS and OS/390 Version 3 Release 1 or later has an SQL
statement coprocessor, which is called the SQL preprocessor. The SQL
preprocessor lifts some of the DB2 precompiler's restrictions on SQL programs.
When you process SQL statements with the SQL preprocessor, you can do the
following things in your program:

- # • Use fully-qualified names for structured host variables
- # • Include SQL statements at any level of a nested PL/I program, instead of in only
the top-level source file
- # • Use nested SQL INCLUDE statements

To process SQL statements as you compile your PL/I program, you specify the
compiler option PP(SQL('option, ...')), which consists of a list of SQL processing
options following the SQL keyword. For example, suppose that you want to process
SQL statements as you compile a PL/I program. In your program, the DATE data
types require USA format, and the SQL statements conform to DB2 rules. This
means that you need to specify the DATE(USA) and STDSQL(NO) options.
Therefore, you need to include this option in your compile step:

```
#                   PP(SQL('DATE(USA), STDSQL(NO)'))
```

#

Unicode support for international data and e-business

In prior releases, DB2 provided support for data that is encoded in ASCII and EBCDIC. The ASCII and EBCDIC encoding schemes represent character data in various environments.

DB2 allowed one set of ASCII CCSIDs and one set of EBCDIC CCSIDs for each system. ASCII and EBCDIC CCSIDs support either one specific geography or one generic geography, such as Western Europe. However, generic CCSIDs were not available for other geographies, such as the Far East. These encoding schemes did not contain enough characters to fully address the needs of users in many different geographies that interact with one DB2 server.

In Version 7, DB2 for OS/390 and z/OS adds support for Unicode, a solution that supports all geographies. Unicode is a powerful encoding scheme that has the capacity to represent the characters of geographies and languages internationally. With Unicode support, a geographically disparate group of users can interact with a central server to store or retrieve data. Unicode meets the needs of the data centers of multinational corporations and of e-commerce. This encoding scheme solves complex user needs, such as working with multilingual text and using technical characters, such as mathematical symbols.

The Unicode consortium published the Unicode standard. Extensive information about Unicode, the Unicode consortium, the Unicode standard, and standards conformance requirements is available at the following Web site: www.unicode.org

How Unicode works

The goal of Unicode is to provide a single, unique definition (code point) for every character in the world. To keep character coding simple and efficient, the Unicode standard assigns each character a unique numeric value and name, thereby treating alphabetic characters, ideographic characters, and symbols equally. These unique code points are independent of any specific operating system, application, or language. With this technique, referred to as *internationalization*, Unicode eliminates the data corruption problems that existed in older code pages because of inconsistent definitions for characters and mixed code sets.

Although the design of Unicode is based on ASCII, it goes far beyond ASCII's limited ability to encode only the Latin alphabet. To encode all of the characters that are used for the written languages of the world, Unicode uses more than 1 byte to represent a character.

Unicode provides a 16-bit encoding that makes code values available for more than 65 000 characters. These 65 000 characters are sufficient for encoding most of the many thousands of characters used in the major languages of the world.

In addition, Unicode provides an extension mechanism called UTF-16. The UTF-16 extension mechanism uses paired code values (called *surrogates* in the Unicode standard) that allow for encoding of as many as one million additional characters, without the use of escape codes.

With this capacity, the complex scripts of Asia and right-to-left scripts the Middle East are represented. Unicode is sufficient for all known character encoding requirements, including full coverage of all historic scripts of the world. It includes punctuation marks, mathematical symbols, technical symbols, geometric shapes, and dingbats.

Three common encoding forms are:

UTF-8 Unicode Transformation Format, 8-bit encoding form, which is designed for ease of use with existing ASCII-based systems.

UCS-2 Universal Character Set coded in 2 octets, meaning that characters are represented in 16-bits for each character.

UTF-16

Unicode Transformation Format, 16-bit encoding form. UTF-16 is a superset of UCS-2.

Unicode characters are usually shown in hexadecimal form following the prefix "U". For example, the code point U+0041 is the hexadecimal number 0041 (decimal 65). It represents the character "A" in the Unicode standard.

DB2 for OS/390 and z/OS implements UTF-8 and UTF-16. UTF-8 data is supported in *mixed data* fields, which are character strings that contain both single-byte and multi-byte characters. UTF-16 data is supported in graphic data fields.

Now, you can store data in a DB2 database using the ASCII, EBCDIC, or Unicode encoding scheme when you create a database, a table space, and a table. You specify the encoding scheme option on SQL statements that include:

- CREATE DATABASE
- CREATE TABLESPACE
- CREATE TABLE
- CREATE GLOBAL TEMPORARY TABLE

- DECLARE GLOBAL TEMPORARY TABLE
- CREATE DISTINCT TYPE
- CREATE FUNCTION
- CREATE PROCEDURE

All data in a table must use the same encoding scheme and all tables in a table space (with the exception of global temporary tables) must use the same encoding scheme.

Attention: DB2 for OS/390 and z/OS does not support changing from one CCSID to another CCSID during migration. Doing so can result in data corruption or other unpredictable results.

Related enhancements

Version 7 adds several enhancements that support Unicode specifically and encoding schemes in general:

- Changes to installation panel DSNTIPF
 - The DEF ENCODING SCHEME option accepts UNICODE as a value.
 - The option UNICODE CCSID is added.
The default is the UTF-8 CCSID, 1208.
 - The option APPLICATION ENCODING is added.
This field specifies a default application encoding scheme; the default is EBCDIC. This value determines how DB2 converts data that is received in SQL statements and host variables. For example, if you set your default application encoding scheme to 37, and your EBCDIC coded character set to 500, DB2 converts all data entering the system from 37 to 500 before using it.
- New CURRENT APPLICATION ENCODING SCHEME special register
This special register specifies the encoding scheme to use for dynamic statements. The value of the ENCODING bind option determines the initial value of CURRENT APPLICATION ENCODING SCHEME if the bind option is specified. Otherwise, the initial value is the value of the default application encoding scheme that is specified on the DSNTIPF installation panel.
- New SQL statements
 - SET CURRENT APPLICATION ENCODING SCHEME assigns a value to the CURRENT APPLICATION ENCODING SCHEME special register.
 - DECLARE VARIABLE defines a CCSID for a host variable.
- New PARAMETER CCSID clause for CREATE FUNCTION and CREATE PROCEDURE statements.
A PARAMETER CCSID clause in all CREATE FUNCTION and CREATE PROCEDURE statements now makes it easier to specify the same encoding scheme for all string parameters. You can specify PARAMETER CCSID to define an encoding scheme other than the default for all string parameters instead of having to specify a CCSID clause for each parameter. In addition, system-generated parameters use the same encoding scheme as user-specified parameters.

#

For more information about the installation options, see *DB2 Installation Guide*. For more detailed information about these SQL statements and the special register, see *DB2 SQL Reference*.

For more information about encoding schemes and character conversion, see Appendix A of *DB2 Installation Guide* and Chapter 1 of *DB2 SQL Reference*.

Chapter 5. Improvements in connectivity

IBM improves e-business connectivity with:

- “Support for COMMIT and ROLLBACK in stored procedures”
- “Support for Windows Kerberos security” on page 46
- “Support for encrypted password or user ID and password” on page 46
- “Support for encrypted change password” on page 47
- “Global transaction support for distributed applications” on page 47
- “Reporting server-elapsed time at the workstation” on page 47
- “Additional enhancements for connectivity” on page 48

Support for COMMIT and ROLLBACK in stored procedures

In previous releases of DB2, you could not include COMMIT statements in stored procedures. You could include ROLLBACK statements in stored procedures, but when DB2 encountered the ROLLBACK statement, it put the unit of work in a *must-rollback* state and returned an SQL error to the calling application. In DB2 Version 7, you can include COMMIT and ROLLBACK statements in a stored procedure. When DB2 executes the COMMIT or ROLLBACK statement, it commits or rolls back all changes within the unit of work. Those changes can include DB2 work that the calling application does before it calls the stored procedure, as well as DB2 work that the stored procedure does.

A stored procedure that includes COMMIT or ROLLBACK statements must be defined with the CONTAINS SQL, READS SQL DATA, or MODIFIES SQL DATA clause. There is no interaction between the COMMIT ON RETURN clause in a stored procedure definition and COMMIT or ROLLBACK statements in the stored procedure code. If you specify COMMIT ON RETURN YES when you define the stored procedure, DB2 issues a COMMIT when control returns from the stored procedure. This occurs, regardless of whether the stored procedure contains COMMIT or ROLLBACK statements.

A ROLLBACK statement has the same effect on cursors in a stored procedure as it has on cursors in stand-alone programs. A ROLLBACK statement closes all open cursors. A COMMIT statement in a stored procedure closes cursors that are not declared WITH HOLD, and leaves cursors open that are declared WITH HOLD. The effect of COMMIT or ROLLBACK on cursors applies to cursors that are declared in the calling application, as well as cursors that are declared in the stored procedure.

Under the following conditions, you *cannot* include COMMIT or ROLLBACK statements in a stored procedure:

- The stored procedure is nested within a trigger or a user-defined function.
- The stored procedure is called by a client that uses two-phase commit processing.
- The client program uses a type 2 connection to connect to the remote server that contains the stored procedure.

You *cannot* include ROLLBACK statements in a stored procedure if DB2 is not the commit coordinator.

If a COMMIT or ROLLBACK statement in a stored procedure violates any of the previous conditions, DB2 puts the transaction in a must-rollback state, and the CALL statement returns a -751 SQLCODE.

Putting transactions in a must-rollback state: In previous releases of DB2, you might have used ROLLBACK statements in stored procedures to indicate to the client program that some kind of error had occurred in the stored procedures. Executing a ROLLBACK in a stored procedure caused DB2 to send an SQLCODE –751 to the client program. In DB2 Version 7, executing a ROLLBACK statement in a stored procedure might not result in SQLCODE –751. If you want to continue to send the same error to the client application, you can write a user-defined function that executes a ROLLBACK statement and invoke that user-defined function from the stored procedure. The user-defined function then returns SQLCODE –751 to the stored procedure, and the stored procedure returns SQLCODE –751 to the client application.

Support for Windows Kerberos security

Kerberos security is a network security technology developed at the Massachusetts Institute of Technology. DB2 for OS/390 and z/OS can use Kerberos security services to authenticate remote users on clients using DRDA, such as a client using DB2 Connect Version 7. With Kerberos security services, remote end users access DB2 for OS/390 and z/OS when they issue their Kerberos name and password. This same name and password is used for access throughout the network, so a separate MVS password to access DB2 for OS/390 and z/OS is not necessary. This single logon capability uses the Kerberos name as the global identity for the end user. The end user avoids the need to supply a user ID and password when connecting to each server.

The Kerberos security technology uses encrypted tickets instead of flowing user IDs and passwords across a network. A Kerberos Authentication Server, where both clients are registered, issues tickets. A client's keys are derived from the user ID and password. The Kerberos protocol is transparent to DB2 for OS/390 and z/OS, so no changes to application programs are required.

DB2 for OS/390 and z/OS support for Kerberos security requires the OS/390 SecureWay Security Server (formerly known as Resource Access Control Facility (RACF)) and the Security Server Network Authentication and Privacy Service, or the functional equivalent. The Network Authentication and Privacy Service provides Kerberos support and relies on a security product, such as RACF, to provide registry support. The OS/390 Security Server allows administrators who are already familiar with RACF commands and RACF ISPF panels to define Kerberos configuration and principal information.

Data sharing environment: Data sharing Sysplex environments that use Kerberos security must have a Kerberos Security Server instance that runs . The instances must either be in the same realm and share the same RACF database, or have different RACF databases and be in different realms.

For more information on the Kerberos security protocol, see the Web site at:
<http://web.mit.edu/kerberos/www/>

Support for encrypted password or user ID and password

Depending on the DRDA level, clients can encrypt passwords or both user IDs and passwords when they send them to a DB2 for OS/390 and z/OS server.

To enable DB2 Connect to flow encrypted passwords, you must set database connection services (DCS) authentication to DCS_ENCRYPT in the DCS directory entry. When the workstation application issues an SQL CONNECT statement, the

workstation negotiates encryption support with the database server. If encryption is supported, the client and server generate a shared private key by using the Diffie-Hellman distribution algorithm (a public key technology), and the password is encrypted by applying the 56-bit Data Encryption Standard (DES) algorithm to the shared private key. The encrypted password cannot be reused, and the shared private key is generated on every connection. If the server does not support password encryption, the application receives SQLCODE -30073.

Support for encrypted change password

When the DRDA requester receives notification that the RACF password has expired, and the requester has implemented function that allows passwords to be changed, then the requester can prompt the end user for the old password and a new password. The requester then uses DB2 Connect support to send the old and new passwords to the DB2 server.

With the extended security option in Version 7, DB2 can now pass the old and new passwords to RACF. If the old password is correct, and the new password meets the installation's password requirements, then the end user's password is changed, and the DRDA connection request is honored. When a user changes a password, the user ID and the old password and the new password are sent to DB2. Depending on the DRDA levels, these three tokens can be encrypted when they are sent to a DB2 for OS/390 and z/OS server.

Global transaction support for distributed applications

In Version 6, enhancements were made to the Recoverable Resource Manager Services attachment facility (RRSAF) to support global transactions. However, global transactions were not supported in a distributed environment.

Distributed applications can now take advantage of global transaction support. Independent DB2 agents can share locks and access the same data. One or more of those DB2 agents can perform update operations. Your transaction manager must coordinate commit operations among the various DB2 agents using a two-phase commit protocol.

#

The SET_ID connection function for the RRSAF sets end-user information that is passed to DB2 when the next SQL request is processed. SET_ID establishes a new value for a program ID, which is defined by the calling program, that can be used to identify the end user. DB2 places the program ID into IFCID 316 records so that you can identify which program is associated with a particular SQL statement. For information on RRSAF calls, see Part 6 of *DB2 Application Programming and SQL Guide*.

Reporting server-elapsed time at the workstation

DB2 Connect clients running on a workstation can use the System Monitor enhancements in DB2 Connect Version 7 to determine the amount of time that it takes DB2 to process a request (server-elapsed time). When the System Monitor statement switch has been turned on, DB2 returns server-elapsed time information as a new element through the regular Snapshot Monitoring APIs. The server-elapsed time includes the actual amount of time it takes for DB2 to parse a remote request, process any SQL statements required to satisfy the request, and generate the reply. Because server-elapsed time does not include any of the network time used to receive the request or send the reply, workstation clients can

use the information to quickly isolate poor response times to the network or to the DB2 server without you having to perform traces on the server.

Additional enhancements for connectivity

Other enhancements that enhance connectivity are:

- Support for IBM DB2 Connect connection pooling and transaction pooling

Establishing a connection from a DB2 Connect server to a host system requires computing resources and time. In an environment where thousands of clients frequently connect to and disconnect from the host through the DB2 Connect server, a substantial portion of processing time is spent in establishing connections and dropping connections.

IBM DB2 Connect supports transaction pooling and connection pooling, which provide a significant performance improvement in such environments.

With connection pooling, DB2 Connect maintains open connections to the database in an available pool. When a client requests a connection, it can be provided from this pool of ready connections. Connection pooling significantly reduces the overhead that is used for opening and closing these connections.

With transaction pooling, multiple applications can share a single physical connection to the server. A physical connection to the server is reusable when an application reaches the end of a unit or work.

DB2 for OS/390 and z/OS Version 7 provides host support for DB2 Connect connection pooling and transaction pooling.

- DB2 Call Level Interface (CLI) support for bookmarks on DB2 UDB for Linux, UNIX and Windows

DB2 CLI on DB2 UDB for Linux, UNIX and Windows supports bookmarks. A bookmark is a pointer to a specific row in a result set. An application can store a bookmark, continue to move throughout the result set, and then return to the bookmarked row to generate a rowset.

You can use bookmarks in DB2 UDB for Linux, UNIX and Windows CLI applications that access data on DB2 for OS/390 and z/OS Version 7.

- MQSeries[®] user-defined functions

DB2 for OS/390 and z/OS Version 7 adds support for user-defined functions that you can use to perform MQSeries tasks.

MQSeries is the core of the WebSphere MQSeries product family. It is a flexible messaging system that lets applications communicate in a distributed environment. The DB2 MQSeries user-defined functions let you easily integrate MQSeries messaging with database applications.

The MQSeries user-defined functions support the following types of operations:

- Send and forget, called a *datagram*, where no reply is needed
- Read or receive, where one or all messages are either read without removing them from the queue, or received and removed from the queue
- Request/response, where a sending application needs a response to a request

You can use the DB2 MQSeries functions to send messages to a message queue or to receive messages from the message queue. In addition, you can send a request to a message queue and receive a response.

The MQSeries server is located on the same OS/390 system as the DB2 database server. The MQSeries user-defined functions provide access to the MQSeries server by using the Application Messaging Interface (AMI).

#

Chapter 6. Features of DB2 for OS/390 and z/OS

Version 7 of DB2 for OS/390 and z/OS offers several features that help you integrate, analyze, summarize, and share data across your enterprise. The following sections contain information about those features:

“Reporting and governing your enterprise using QMF”

“Managing your enterprise with the DB2 Management Clients Package” on page 52

“Web-ready applications” on page 60

“Integrating and analyzing your business information using the DB2 Warehouse Manager” on page 62

“Data Management Tools for your database” on page 64

Reporting and governing your enterprise using QMF

Query Management Facility (QMF) is the tightly integrated, powerful, and reliable tool for query and reporting within IBM's DB2 family. QMF provides new capabilities that improve user productivity and enhance functionality. With QMF, you can work with any DB2 data from DB2 for OS/390 and z/OS to DB2 for VSE and VM, to DB2 for AS/400 to DB2 on workstation servers that run OS/2, Windows NT, AIX and other UNIX operating systems. QMF supports DB2 on large parallel processors. When coupled with DB2 DataJoiner, QMF allows access to nonrelational and other vendor data sources.

QMF is a family of integrated tools that offers a total solution for your enterprise query and reporting needs. With these tools, you can access large amounts of data, share central repositories of queries and reports, and publish reports to the Web for viewing with a browser. You can also allow a broader set of users to access QMF, knowing that your applications are secure.

QMF Version 7

QMF enables your users to issue queries on OS/390 and z/OS. QMF Version 7 enhancements include support for the following functions:

- New default edit codes for current DB2 date (DD) and time (TT) formats in date columns and time columns
- Direct navigation to the QMF HOME panel with the SHOW command
- Comprehensive defaults for object types in many commands, including:
 - CONVERT
 - EDIT
 - EXPORT
 - PRINT
 - RESET
 - RUN
 - SAVE
- Functions that are easier to use with prompting for commands and defaults that are preloaded in prompts so users do less typing
- Extended text option in the MSG command, eliminating the need to add double quotation marks when the text of a message has single quotes
- Two DB2 data types: ROWID and limited LOB support
- Remote-unit-of-work application request in DB2 for VSE using Customer Information Control System (CICS)
- Server support for DB2 for AS/400 Version 4 Release 4 or later releases

- Installation improvements so that you can use one QMF installation base that you bind on other platforms

QMF High Performance Option

QMF HPO provides a set of integrated tools for performance management within the QMF environment and facilitates the administration of QMF. QMF HPO includes the QMF HPO/Manager and the QMF HPO/Compiler. QMF HPO/Manager consists of a group of facilities that improve governing and object management capabilities. The program includes a preemptive governor to analyze QMF queries. The governing capabilities enable you to establish controls that protect production applications while delivering on-demand information. QMF HPO/Manager provides:

- Scheduling for different user groups by time of day and day of week
- Control over resource consumption, including by numbers of rows and bytes fetched
- The ability to allow and disallow SQL verbs and QMF commands
- Extensive object tracking to gain knowledge about your QMF environment so you can train users and improve performance

QMF HPO/Compiler automatically converts queries and reports into efficient programs in COBOL. The converted programs use static SQL in place of dynamic SQL, which reduces processor usage, DB2 catalog contention, DB2 optimizer overhead, and security concerns.

Version 7 of QMF HPO includes the following enhancements:

- Improved tracking of QMF objects to support very large numbers of QMF users simultaneously
- Preemptive governing to guard QMF workloads from runaway adhoc queries
- Preemptive governing of QMF batch processes using QMF HPO/Manager
- Ability to bind programs that are generated by QMF HPO Compiler into packages, in addition to or instead of, plans

QMF for Windows

For customers with DB2 databases of many sizes, QMF for Windows provides a query tool with standard Windows capabilities. QMF for Windows delivers several benefits in one package, which includes an intuitive quick-start user interface. You can automate tasks and develop powerful native Windows applications. QMF for Windows provides the following features:

- A full-window table editor that you can use to update DB2 data
- Centralized control over resource use
- Web publishing capabilities and Java-based query and Web publishing capabilities
- A powerful WinSock application programming interface (API) to automate database query, update, and report distribution tasks

QMF for Windows is based on Distributed Relational Database Architecture (DRDA). QMF for Windows provides access to DB2 database servers on all platforms through TCP/IP, Systems Network Architecture (SNA), or Call Level Interface (CLI), as Figure 9 on page 51 shows. To provide this access, QMF uses direct DRDA connectivity without database gateways, middleware, or Open Database Connectivity (ODBC) drivers.

Enhancements to QMF for Windows Version 7 include:

- An improved user interface:

- QMF for Windows provides a point-and-click and a drag-and-drop interface to QMF Form creation.
You can automatically populate a QMF Form definition by using aggregation, grouping, and formatting.
- You can convert the resulting display options to QMF form definitions.
- You can more intuitively navigate and locate QMF objects with a Windows-style tree control.
- You can create job schedules by calendar, time of day, and more from within QMF for Windows.
- Java servlet support enables you to perform a Java-based query from a browser.
- QMF for Windows includes QMF Report Center as an optional install. QMF Report Center was previously a separate download named "Personal Portal,".
- When you invoke the QMF for Windows Table Editor, you can use the DB2 Forms option if you have it installed.

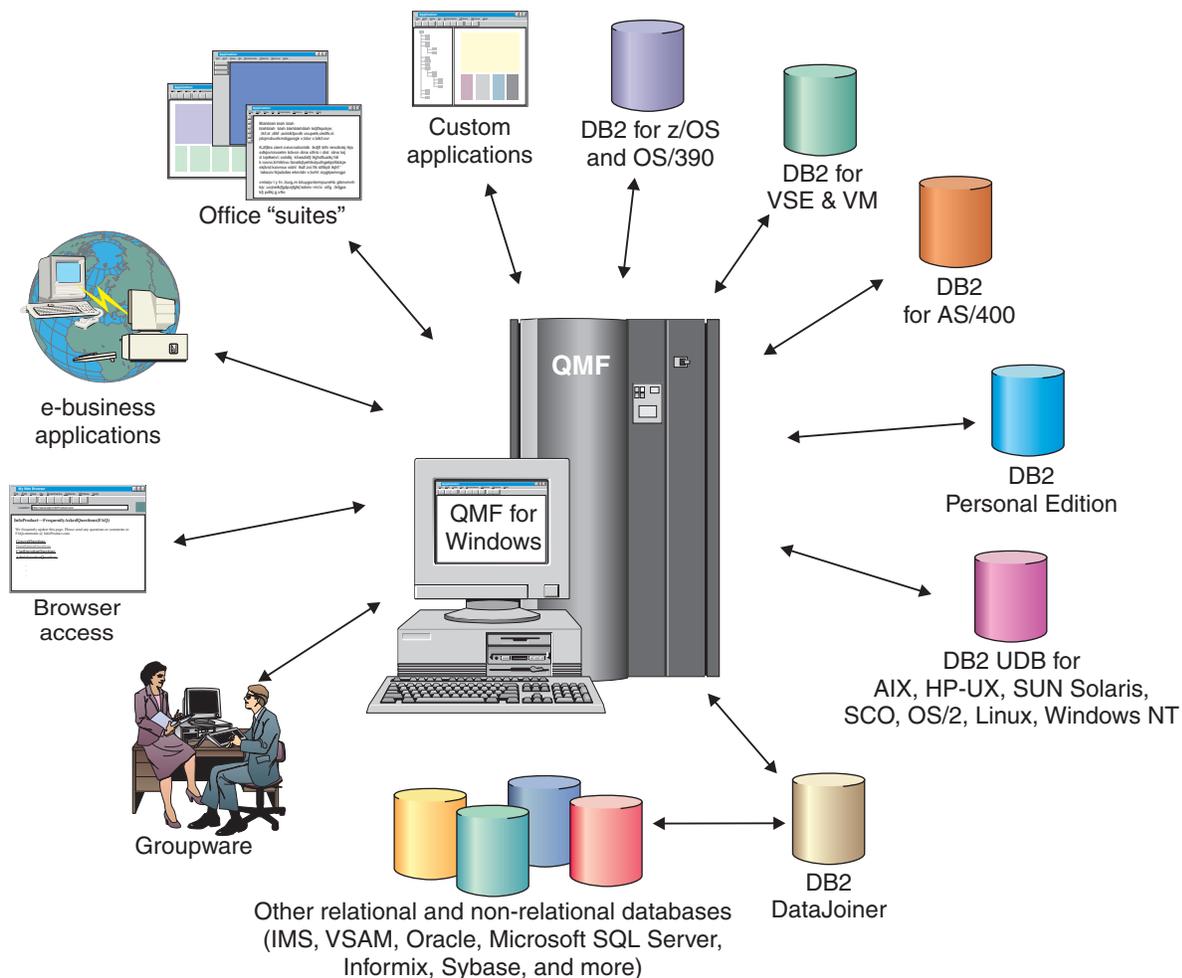


Figure 9. Accessing data using QMF for Windows

- More command icons are available so that you can customize the toolbar.
- QMF for Windows can remember DB2 passwords when that are used in the Windows NT environment.

- You can sort query results on more than one noncontiguous column and in day-of-week or month-of-year order.
- The Visual Warehouse object library now integrates QMF for Windows objects
- Enhanced database connectivity for distributed applications:
 - Support for stored procedures that return multiple result sets by using the CLI interface
 - Large object support
 - Enhanced support for DB2 for OS/390 and z/OS data sharing
 - Display of CLI-specific information when a CLI connection to the server exists
 - New trace options include CLI, TCP/IP, embedded SQL, SQLAM, REXX, CPI-C, and DDM
- More support for QMF for OS/390:
 - Support for all types of QMF for OS/390 procedures, including procedures with REXX logic
 - A new docking toolbar for entering and issuing single QMF commands
- Several new QMF commands:
 - Form conditions and form column definitions
 - A RESET GLOBAL command
 - PF keys that are defined to match QMF for OS/390
- A user interface with national language support for:
 - 19 language translations, which in Version 7 include Belgian French, Canadian French, Swiss French, Swiss Italian, Arabic, and Traditional Chinese
 - Unicode

Managing your enterprise with the DB2 Management Clients Package

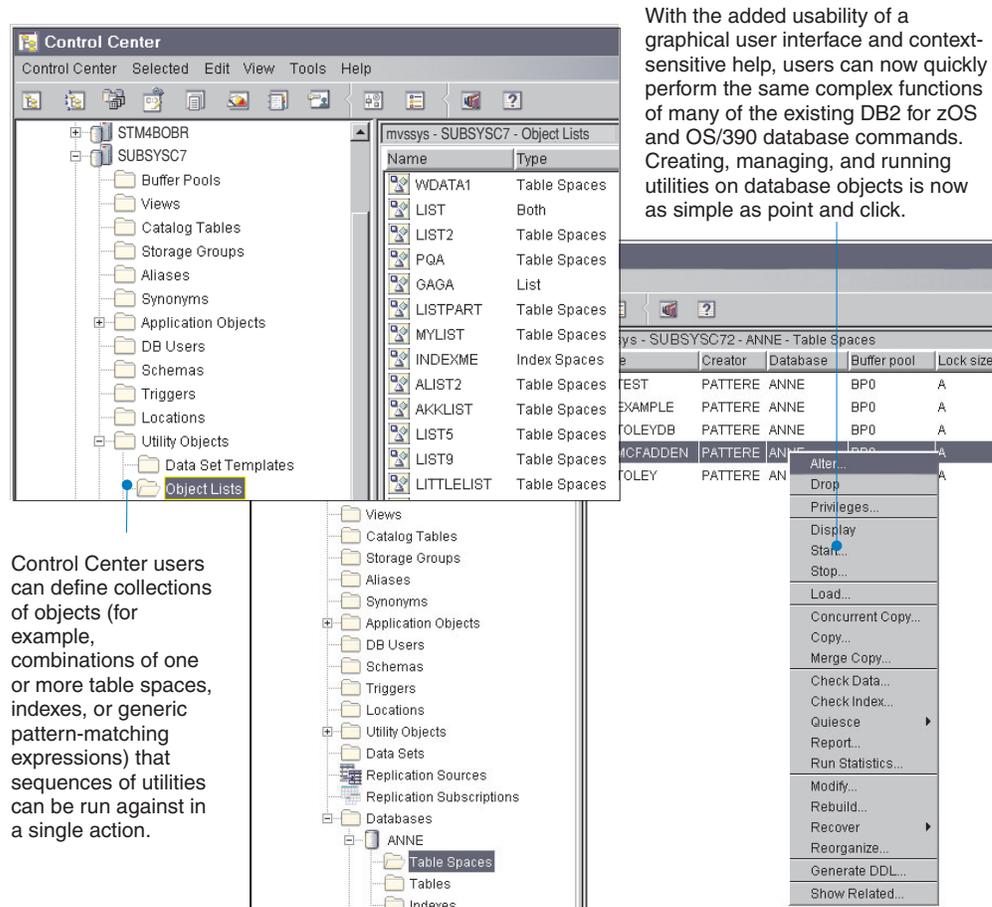
The DB2 Management Clients Package is a no-charge feature of DB2 for OS/390 and z/OS Version 7. The Management Clients Package is a collection of workstation-based tools that you can use to work with and manage your DB2 for OS/390 and z/OS environment. The elements of the DB2 Management Clients Package are:

- DB2 Control Center
- DB2 Stored Procedure Builder
- DB2 Installer
- DB2 Visual Explain
- DB2 Estimator

Managing DB2 for OS/390 and z/OS from your workstation with Control Center

The IBM DB2 Universal Database Control Center is a graphical interface that is designed to help you manage DB2 databases on different operating systems. The Control Center can run either as a Java application or as an application on your Web server that your Web browser can access.

With the Control Center for DB2 for OS/390 and z/OS, you can manage DB2 for OS/390 Version 5, Version 6, and Version 7 objects. These objects display on the Control Center main window, along with DB2 Universal Database objects. See Figure 10 on page 53. Using the Control Center, you can create, alter, and drop objects. You can also run utilities that reorganize or load your data in your existing DB2 for OS/390 databases.



Control Center users can define collections of objects (for example, combinations of one or more table spaces, indexes, or generic pattern-matching expressions) that sequences of utilities can be run against in a single action.

With the added usability of a graphical user interface and context-sensitive help, users can now quickly perform the same complex functions of many of the existing DB2 for z/OS and OS/390 database commands. Creating, managing, and running utilities on database objects is now as simple as point and click.

Figure 10. A DB2 for OS/390 and z/OS subsystem and its objects as displayed in the main navigation window of Control Center

The Control Center for DB2 for OS/390 and z/OS enhances the usability and administration of DB2 for OS/390 objects from the workstation. In Version 7, IBM significantly improves the usability of the DB2 for OS/390 and z/OS utilities.

Managing DB2 for OS/390 and z/OS data sets from the Control Center

You can now retrieve a list of data sets that reside on an OS/390 or a z/OS system from the Control Center. The interface to the Multiple Virtual Storage (MVS) file system allows you to work with four types of data sets: A physical sequential data set, a partitioned data set, a partitioned data set extended, or a generation data group. Depending on the data set type, you can rename, delete, or display members of the data set through the Control Center.

Restarting DB2 for OS/390 and z/OS utilities from the Control Center

You can now restart utilities that were originally started from the Control Center. You can execute restart from the last committed point or from the last committed phase. You can define a custom utility identifier from the Control Center, and associate the identifier with any utilities as they run. Customized utility identifiers help you find the utilities that you started among others that are running on the system.

Generating data definitions

You can use the DB2 Administration Tool with Control Center to extract object definitions from the DB2 for OS/390 and z/OS catalog tables. You use the

definitions to recreate databases, table spaces, tables, procedures, schemas (for granting privileges), user-defined functions, and distinct types. This feature is useful for creating a copy of a particular object before making changes to it when fallback might be necessary. You can also use this feature to move or copy an object and its environment to another subsystem. For example, you can copy the table space object, including all its objects, such as tables, indexes, and aliases that are associated with that object, to another subsystem.

Object lists and utility procedures

In the Version 7 Control Center, you can define collections of objects, called object lists. These object lists are combinations of one or more table spaces or index spaces that you can easily generate using pattern-matching characters. With a single command, you can run utilities (like COPY, CONCURRENT COPY, QUIESCE, and REORGANIZE) on every object in the list. By creating a utility procedure, you can define a sequence of object lists or utility combinations that you can run with a single command.

Data set templates

You no longer need to specify a data set name, a data set device, and data set space allocation for each utility. Now you can use data set templates to dynamically allocate data sets that DB2 for OS/390 and z/OS utilities use. Use data set templates to automatically generate the data set naming conventions, the data set parameters, and the media allocation parameters. The default data set templates available with DB2 for OS/390 and z/OS are shown in Figure 11.

These are the utilities currently supported by Data set templates.

Users can select from sample templates, or build their own to set as defaults for Data sets when running utilities.

This field displays the Data set naming convention that the template defines.

Utility	Data set description	Template name	Data set name
Concurrent copy	Primary copy data set at local site	CCCOPY	&USERID..DB2.&SSID..&DB..&SN..D&JDAY.&HOUR.&MINUTE
Concurrent copy	Backup copy data set at local site		
Concurrent copy	Filter data set	CCFILTER	&USERID..DB2.&SSID..&DB..&SN..D&JDAY.&HOUR.&MINUTE
Concurrent copy	Primary copy data set at recovery site		
Concurrent copy	Backup copy data set at recovery site		
Copy	Primary copy data set at local site	CCCOPY	&USERID..DB2.&SSID..&DB..&SN..D&JDAY.&HOUR.&MINUTE
Copy	Backup copy data set at local site		
Copy	Primary copy data set at recovery site		
Copy	Backup copy data set at recovery site		
Reorganize index	Unload data set	CCUNLOAD	&USERID..DB2.&SSID..&DB..&SN..D&JDAY.&HOUR.&MINUTE
Reorganize table space	Primary copy data set at local site	CCCOPY	&USERID..DB2.&SSID..&DB..&SN..D&JDAY.&HOUR.&MINUTE
Reorganize table space	Backup copy data set at local site		
Reorganize table space	Discard data set	CCDISCRD	&USERID..DB2.&SSID..&DB..&SN..DISCARD(+1)
Reorganize table space	Punch data set	CCPUNCH	&USERID..DB2.&SSID..&DB..&SN..D&JDAY.&HOUR.&MINUTE
Reorganize table space	Primary copy data set at recovery site		
Reorganize table space	Backup copy data set at recovery site		
Reorganize table space	Unload data set		
Reorganize table space	Input sort data set	CCSORTIN	&USERID..DB2.&SSID..&DB..&SN..D&JDAY.&HOUR.&MINUTE
Reorganize table space	Output sort data set	CCSRTOUT	&USERID..DB2.&SSID..&DB..&SN..D&JDAY.&HOUR.&MINUTE

Figure 11. The default data set templates available with DB2 for OS/390 and z/OS

In addition to providing standardization, data set templates allow DB2 to calculate the size of output data sets and work data sets. The use of data set templates can reduce utility failures that occur because of miscalculations. In the Control Center, you can associate each utility with default data set templates. You do not need a

data set entry when you create utility commands. The initial installation of the DB2 for OS/390 and z/OS enablement for the Control Center includes several templates.

With the default table for the data set templates, you can map the data sets that each utility needs to the default data set templates. The Control Center uses the default table each time you display the utility window. See Figure 11 on page 54, which shows the window in the Control Center that lists the sample default templates that are installed with DB2 for OS/390 and z/OS. Edit these templates to reflect the conventions that your production subsystem uses.

Information Center

The Control Center includes an Information Center for quick access to DB2 family product information, including:

- Database tasks
- Reference material
- DB2 documentation
- Warehouse administration information
- Troubleshooting aids
- Sample programs for application development on workstations
- DB2 Web-related links

You can use the Information Center to link to current DB2 for OS/390 and z/OS publications in Portable Document Format (PDF) on the Web. You can download the publications to your local server and update the Information Center links to access DB2 for OS/390 and z/OS books directly from your intranet.

Building DB2 stored procedures from your workstation

The IBM DB2 Stored Procedure Builder, an element of the DB2 Management Clients Package, provides an easy-to-use development environment for creating, installing, and testing stored procedures. With the DB2 Stored Procedure Builder, you can focus on creating your stored procedure logic rather than on the details of registering, building, and installing stored procedures on a DB2 server. The Stored Procedure Builder provides a single development environment that supports the entire DB2 family ranging from the workstation to System/390. Figure 12 on page 56 depicts the flexibility that is provided by the tool for building and using stored procedures. For example, you can develop stored procedures using IBM Visual Basic™, Microsoft Visual Basic®, or Microsoft Visual Studio®.

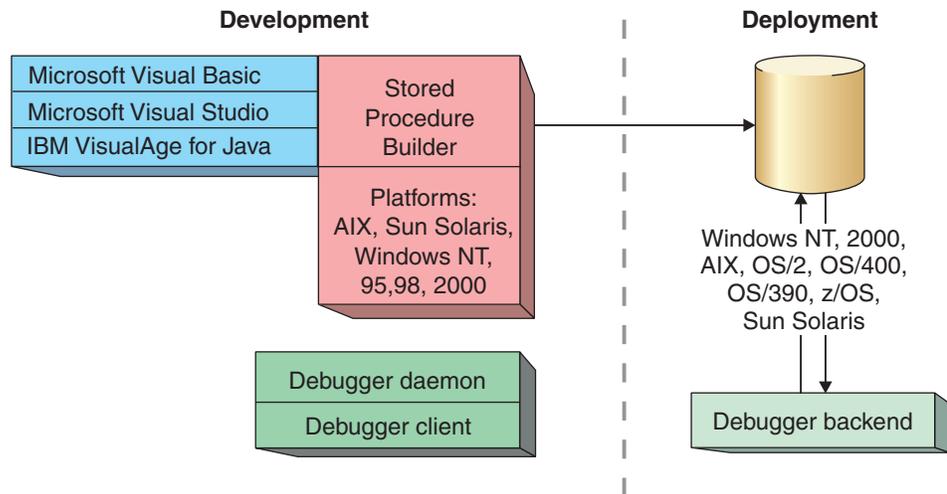


Figure 12. Using Stored Procedure Builder

With the DB2 Stored Procedure Builder, you can develop stored procedures on one operating system and deploy them on other server operating systems. The Stored Procedure Builder supports two commonly used languages for stored procedures: SQL Procedures language and Java.

The DB2 Stored Procedure Builder has a graphical user interface that guides you through tasks with the help of basic design patterns, SQL assistants, and costing information. Use the Stored Procedure Builder to perform a variety of tasks that are associated with stored procedures, such as:

- Viewing existing stored procedures
- Modifying existing stored procedures
- Creating new stored procedures
- Running existing stored procedures
- The copying and pasting of stored procedures across connections
- Building stored procedures on target databases in a single step
- Customizing the settings to enable remote debugging of installed stored procedures

The Stored Procedure Builder requires DB2 Connect. The DB2 Management Clients Package provides a restricted-use copy of DB2 Connect Version 7 to satisfy this functional dependency.

Installing DB2 from a workstation

DB2 Installer is a workstation client that is delivered as an element of the DB2 Management Clients Package. DB2 Installer enhances your productivity whether you are installing DB2 for OS/390 and z/OS for the first time or you are an experienced installer. From your workstation, using a graphical user interface shown in Figure 13 on page 57, you can perform the following tasks:

- Install, migrate, or update DB2 for OS/390 and z/OS from a graphical interface. The graphical interface illustrates the overall installation process and keeps a graphical record of how you defined each subsystem, as shown; see Figure 13 on page 57.
- Customize your DB2 subsystem as much as you need to. You can define a basic subsystem quickly, or you can customize every installation option. The main windows display the parameters that you must specify, and secondary windows display the advanced options.

- Easily control DB2 parameters and run SMP/E, installation, migration, update, fallback, and sample jobs if you have a TCP/IP connection to the DB2 Universal Database Server for OS/390 and z/OS. You receive job status dynamically, and you can edit JCL and examine job output from the workstation.
- Install optional features of DB2 for OS/390 and z/OS, including DB2 Performance Monitor and DB2 DataPropagator.

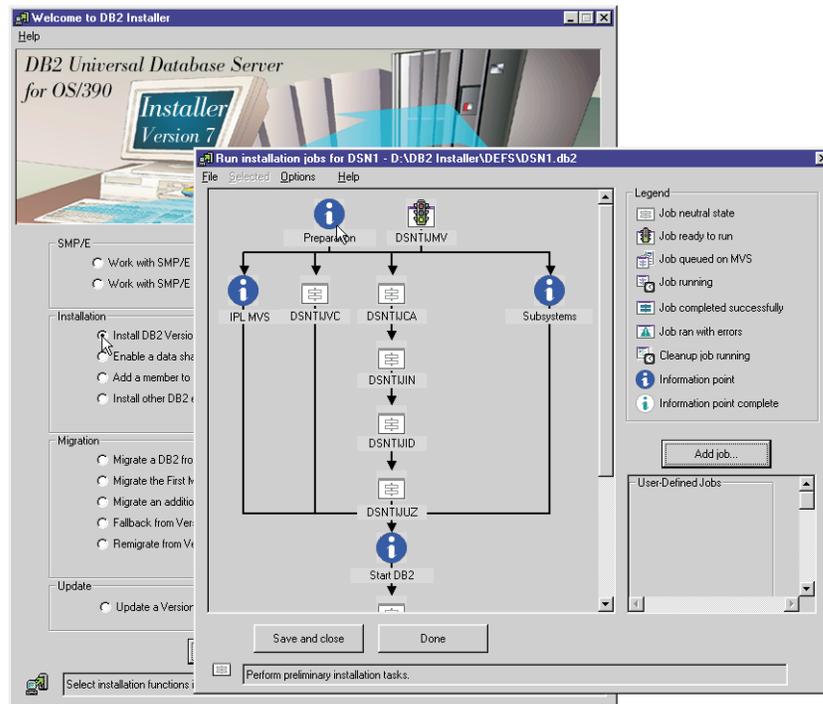


Figure 13. DB2 Installer

Enhancements for Version 7 of DB2 Installer include:

- The ability to use an IBM-supplied stored procedure to download subsystem parameters from your operational DB2 subsystem. You can use the subsystem parameters to migrate or update your subsystem.
- Improved usability of several DB2 Installer windows, including the **File Dialog** windows, the **Before You Begin** window, and the **Welcome** window.
- Support for Windows 95, Windows NT, and Windows 2000.

Using workstation views of DB2 Explain output

DB2 Visual Explain is a workstation client that is an element of the DB2 Management Clients Package. DB2 Visual Explain is an easy-to-use workstation tool that presents the output from DB2 EXPLAIN and dynamic EXPLAIN in a graphical format. Relationships between database objects, such as tables and indexes, are instantly clear as are various operations, such as table space scans and sorts. DB2 Visual Explain also includes a browser for viewing DB2 subsystem parameters. Figure 14 on page 58 displays a graph of an access path.

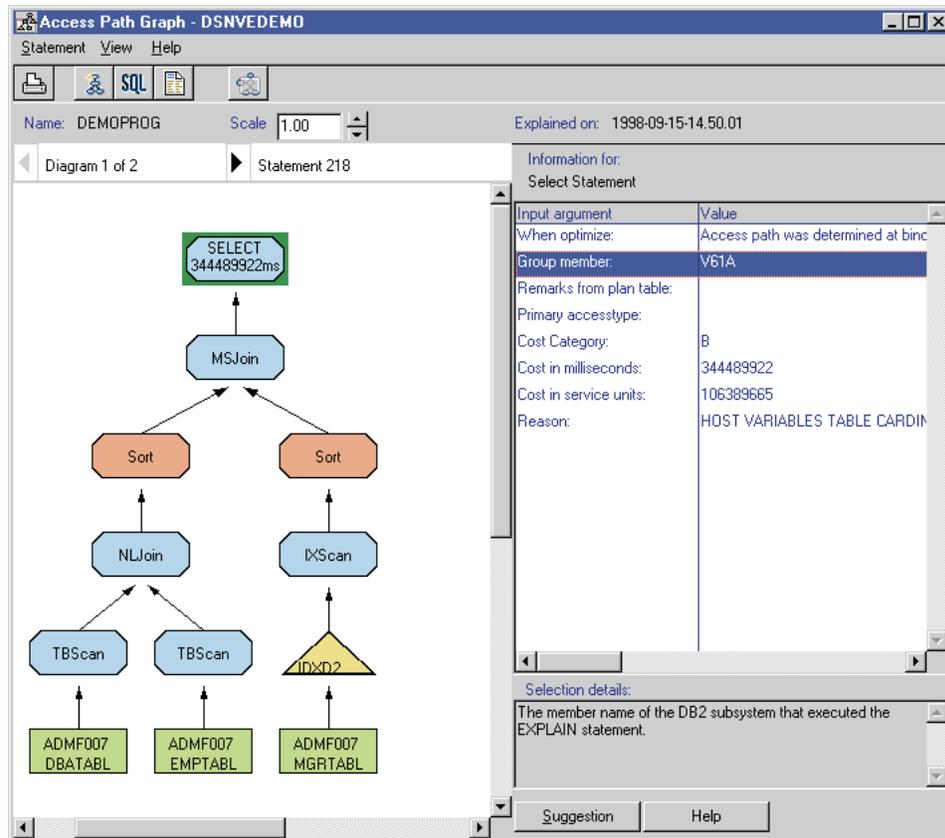


Figure 14. A graph of an access path, displayed by DB2 Visual Explain

If you are using the Control Center, you can start Visual Explain directly from the Control Center.

Use Visual Explain to perform management tasks for DB2 for OS/390 and z/OS:

- View statement costs in milliseconds and service units. The graph of the access path includes the DB2 estimated cost. You can include the cost when viewing explainable statements. Then you can either sort your statements by cost or filter out statements that are based on their costs.
- Filter explainable SQL statements from multiple plans and packages. You can list explainable statements from many plans and packages and then filter them by criteria that you specify. The criteria include statement cost, access path steps (such as table space scans and sorts and either tables or indexes that your sql statements reference). You can save filter criteria specifications for later use or modification.
- Quickly generate customized reports. The Visual Explain report wizard helps you create customized reports on one or more explainable statements. You choose how detailed a report you want. The report can include the statement cost, a description of the access path, and catalog statistics on the indexes and tables. You can print the report or save it as a text file.
- Specify your own qualifier for catalog tables. You can avoid contention against your DB2 catalog for Visual Explain queries by first copying the catalog tables. Then you can specify your own qualifier to access the copied catalog tables when Visual Explain retrieves DB2 catalog information.

DB2 Visual Explain saves statistics history for comparison with new variations that you enter so that you can improve your access paths. Version 7 of DB2 Visual Explain also provides:

- The ability to display information from PLAN_TABLE columns that are new for Version 7 of DB2 for OS/390 and z/OS.
- Better graphical display of information for statements with multiple query blocks.
- A like filter that gives you more flexibility in retrieving data from the PLAN_TABLE, DSN_STATEMENT_TABLE and DSN_FUNCTION_TABLE.

Estimating DB2 performance with DB2 Estimator

DB2 Estimator is an easy-to-use, standalone tool for estimating the performance of applications for DB2 for OS/390 and z/OS. DB2 Estimator is an element of the DB2 Management Clients Package. Run it on your desktop personal computer, or take it with you on your laptop computer.

With DB2 Estimator, you can model a partial DB2 application or a complete real or planned DB2 application without requiring an actual DB2 system. By providing simple table sizings to a detailed performance analysis of an entire DB2 application, DB2 Estimator saves time, lowers costs, and reduces risk. You can use DB2 Estimator to investigate the impact to your production system of new or modified applications before you implement them. You can do what-if analysis to assess the impact of changes you are considering. Answer many questions, such as:

- What is the impact on your system if the transaction volume doubles?
- What is the impact if your databases increase in size?
- What is the effect on response time if you use a faster processor?
- Is your batch window large enough for your utilities?
- How much storage do I need for the new table and its indexes?

Use DB2 Estimator during all life-cycle phases of a DB2 application. During the initial design phase, you can easily:

- Determine whether your design is optimal and feasible
- Investigate alternative database designs
- Assess the impact of using triggers and different ways of structuring queries and transactions
- Eliminate problems early in the design phase

When creating database objects, use the models that you specified in DB2 Estimator as a guide for naming columns and for specifying attribute values. You can model your system using actual DB2 information by importing information from the DB2 catalog and, if available, from DB2 EXPLAIN or the DB2 Performance Monitor (DB2 PM).

When your application is in production, use DB2 Estimator with tools such as DB2 PM to solve application performance problems. You can evaluate alternative SQL designs, without any risk to your production environment, before you change any production database objects. DB2 Estimator also helps you determine the impact of changes to hardware or workload.

You can use DB2 Estimator for Versions 5, 6, and 7 of DB2 for OS/390 and z/OS. It runs in any environment that supports Windows 95, Windows 98, Windows 2000, or Windows NT. Use DB2 Estimator on any data that you imported from your DB2 for OS/390 and z/OS subsystems. Model an application even when none of the tables, SQL statements, transactions, or configurations exist.

Version 7 of DB2 Estimator adds support for scrollable cursors, FETCH FIRST *n* ROWS, and statistics history.

Web-ready applications

Designed for use with the Web, DB2 supports all the key Internet standards. Built-in Java support, an Extensible Markup Language (XML) Extender, and a Web server make it easy to deploy e-business applications.

Text search capability for the Internet with Net Search Extender

Net Search Extender delivers the latest in high-speed search technology. Use its powerful search to provide rapid query responses for your e-business applications. Net Search Extender combines in-memory database technology with text search technology. This new extender is ideal for your e-commerce and Web self-service applications. Net Search Extender searches text data contained in DB2 and can handle the heavy text search demands of large, text-intensive Web sites. It rapidly searches data without locking database tables and delivers excellent query performance and scalability.

Functions available in Net Search Extender

Net Search Extender delivers the following capabilities:

Indexing

You can use a command line interface to create your search indexes in main memory. Flexible index options permit the use of the same text column in one or more indexes. Net Search indexing proceeds without placing locks on the text data.

Searching

You invoke text searches via a stored procedure interface on your server. The powerful search algorithms permit searching by word, phrase, stem, or by fuzzy search. Tags define limited sections of the text document to restrict the range of the search. You can combine conditions for the search argument, using Boolean expression and wildcard operations.

Manageable search results

You specify sort definitions for the search results during index creation. Use result subsets and limits on search terms for your search to manage search performance where large data volumes might be involved. Cursor positioning permits you to navigate through the result set.

In-memory technology

At index creation time, you specify the parts of the table that you want to store in main memory, such as the table columns that you want to sort for a text index. For example, you could presort a text index on a book abstract column by the value in the price column.

Here is an example of how Net Search Extender performs these functions. A customer accesses an online bookstore on the Web that has information about their merchandize in a DB2 for OS/390 and z/OS database. The customer requests a list of all books about relational database. The DB2 application responds with a list that is ordered by price and includes the name of the author. Net Search Extender completes the transaction as shown in Figure 15 on page 61.

- The Web site customer enters their query into the order application
- The application does a stored procedure call to the database
- Net Search issues the text query "relational databases" against the text indexes, saving time and locks by not searching through standard SQL query

- The search engine preserves the order specified during index creation in the result list
- Net Search Extender returns corresponding results from the main memory table, listing the lowest priced books first

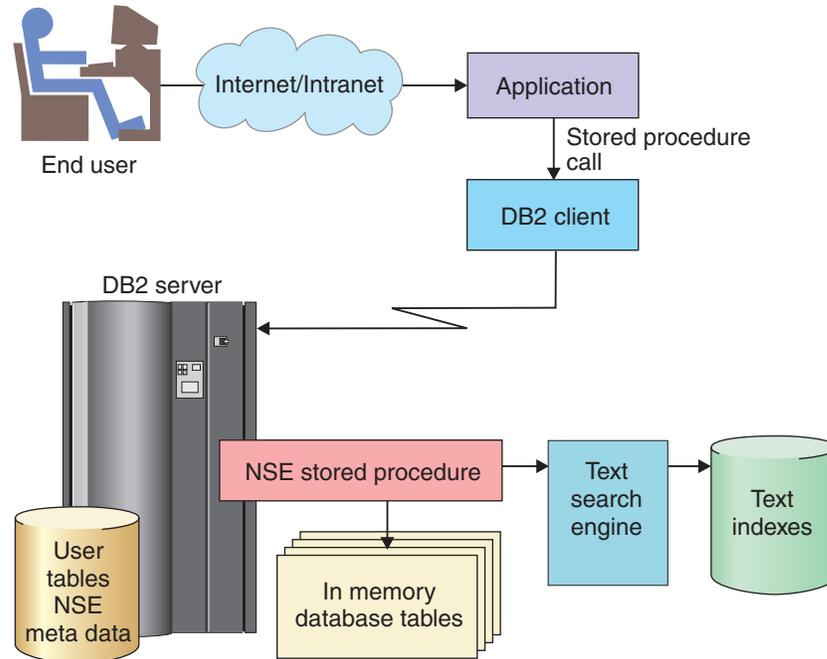


Figure 15. Searching a Web site using Net Search Extender

Using Net Search Extender

The powerful technology of Net Search Extender complements the search function in Text Extender. Net Search Extender provides a single stored procedure for searching large amounts of text.

Text Extender provides text search function that is fully integrated into SQL. You can imbed these functions inside SQL statements. Use Text Extender with your XML documents, for thesaurus support, and to perform integrated multimedia searches, such as searching images.

Net Search is ideal for searching Web sites that are text-intensive and that receive a high level of concurrent queries. You can use the stored procedure in Net Search Extender with any Net.Data, Java, or DB2 CLI client application.

Net.Data for secure Web applications

Net.Data takes advantage of the System/390 capabilities as a premier platform for electronic commerce and Internet technology. Net.Data Web applications provide continuous application availability, scalability, security, and high performance. Using Net.Data for OS/390 and z/OS for Web-based applications in the enterprise, you can:

Publish interactive data and provide universal access to dynamic data

Web applications need access to the key data in the enterprise. Net.Data provides dynamic access to DB2 family data, DRDA-enabled data sources, and HFS (hierarchical file system) flat-file data.

Benefit from high-performance Web-enabled data applications

Net.Data on OS/390 and z/OS benefits from the proximity of Web applications to the DB2 data source. Net.Data eliminates the need to transfer large volumes of data over a network to a workstation server. Net.Data applications can call DB2 stored procedures to take advantage of DB2 performance and to reduce network traffic.

Rely on System/390 availability, scalability, and security

Net.Data relies on the strength of the System/390 platform as a highly available, scalable network server. Net.Data works with DB2 and the Web server (IBM Internet Connection Secure Server and Domino Go Webserver) for secure Web applications.

Use the robust application development function

Net.Data provides a powerful macro capability for robust Web application development. The Net.Data application can call DB2 stored procedures, as well as scripts and programs that are written in Perl, REXX, and C/C++. The application can also call Java applets and JavaScripts. Net.Data also provides an ODBC interface for DB2.

Net.Data Version 7 offers all of the capability of previous releases of Net.Data and much more, including many new features and performance, scalability, tracing, and service enhancements. Using Version 7, you can :

- Execute Net.Data as a FastCGI application
- Use Support for Web page caching as well as the manual management of LOBs and cached Web pages when Net.Data is configured for CGI
- Upload files to the server
- Generate XML compliant documents using the new XMLBlock
- Call SQL functions from the REPORT and ROW blocks of other SQL functions
- Call SQL functions from the REPORT and ROW blocks of other SQL functions
- Use a new language environment for running COBOL applications
- Use several new Net.Data built-in functions
- Use the DB2 PREPARE CACHE capability
- Write user-specified messages to the Net.Data error log and the Net.Data trace log through built-in functions and user-written Language Environments
- Use support for the DTW_DEFAULT_MACRO configuration variable

Integrating and analyzing your business information using the DB2 Warehouse Manager

The DB2 Warehouse Manager brings together the tools to build, manage, govern, and access DB2 for OS/390-based data warehouses. The DB2 Warehouse Manager uses proven technologies with new enhancements that are not available in previous releases. The DB2 Warehouse Manager delivers tightly integrated components that enable you to do the following tasks:

- Simplify prototyping, development, and deployment of your warehouse
- Help your users find, understand, and access information
- Give you more flexibility in the tools and techniques you use to build, manage, and access the warehouse
- Meet the most common reporting needs for enterprises of any size

The DB2 Warehouse Manager for OS/390 provides the following components:

- A restricted license for Version 7 of DB2 Universal Database Enterprise Edition, which delivers the database warehouse management infrastructure and OLAP Starter Kit. This infrastructure replaces Visual Warehouse with enhancements that include:
 - The Data Warehouse Center, as shown in Figure 16, is a new graphical interface that is integrated with the DB2 Control Center. The Data Warehouse Center which provides the administration for building and managing data warehouses. New features include:
 - A navigator window to display and browse warehouse objects
 - A schema modeler that visualizes table relationships
 - The ability to extend the environment with user-written stored procedures or user-defined functions
 - A process modeler that provides a canvas and palette for defining warehouse steps and control flow between them
 - Integration with the DB2 OLAP Integration Server
 - Integration with DB2 DataPropagator to define subscriptions for warehouse population
 - An integrated tutorial for fast learning
 - A warehouse manager, which runs on Microsoft Windows NT and controls all the operations that are defined through the Data Warehouse Center.
 - A warehouse agent for Windows NT, which executes local operations on behalf of the components of the DB2 Warehouse Manager.
 - The OLAP Starter Kit, which is a limited user license for DB2 OLAP Server and the OLAP Integration Server for building and deploying OLAP applications.

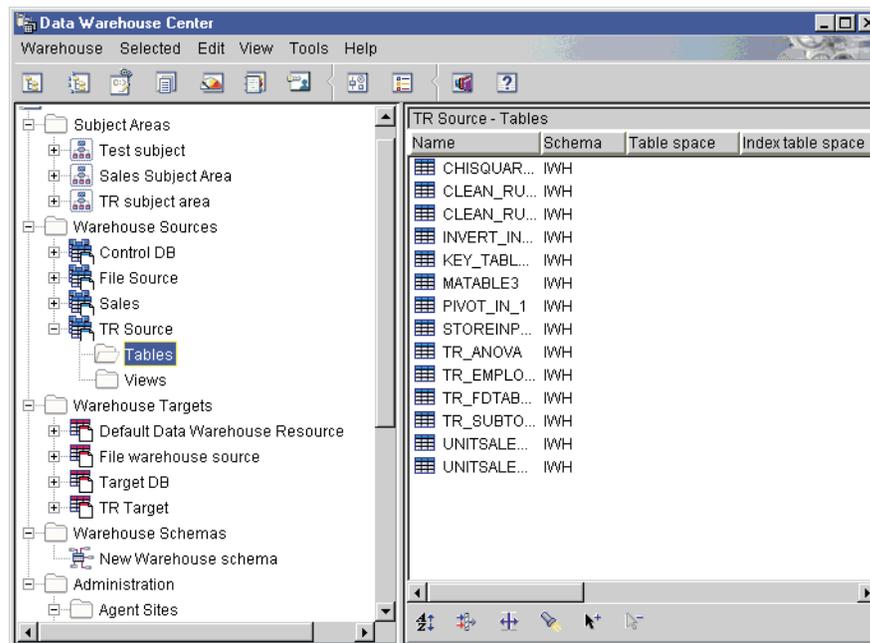


Figure 16. DB2 Data Warehouse Center

- Warehouse agents for OS/390, Unix, AS/400, and Windows NT that execute processes on behalf of the components of the DB2 Warehouse Manager.

- Prebuilt programs for performing a variety of OS/390–specific tasks, such as executing file transfer protocol (FTP), submitting job control language (JCL), and triggering server or client programs.
- Warehouse transformers for OS/390, which are stored procedures or user-defined functions. These transformers perform complex transformations that are commonly used in warehouse development including data manipulation, data cleansing, key generation, and statistical calculations. These transformers augment the transformations that are available with built-in SQL, including SQL string manipulation, Boolean operations, mathematical calculations, data type conversions, and utility invocations.
- The enhanced Information Catalog Manager, previously available with Visual Warehouse, improves the Web interface. The Information Catalog Manager provides tighter integration with DB2 OLAP Server, QMF from any DB2 database, Hyperion’s Essbase, other front-end tools, and IBM Enterprise Information Portal (EIP).

Data Management Tools for your database

IBM is investing heavily in developing tools for your DB2 databases. Now you can select the tools that you need at a price that you like from a group of products that will help you meet your business challenges. A wide variety of tools are available:

IBM DB2 Administration for OS/390

A tool that provides a comprehensive set of database management functions and includes two new functions. ALTER supports modification of tables and their attributes. MIGRATE provides the facilities to copy data and objects to other DB2 subsystems.

Program number: 5655-E70

IBM DB2 Archive Log Compression Tool for z/OS

Restores data directly from highly compressed, off-site copies of DB2 logs for a complete disaster recovery support system. Database administrators can reduce the volume of archived logs, resulting in shorter I/O and recovery times. The tool also helps lower storage costs by helping you store logs on affordable direct access storage device (DASD) instead of tape.

Program number: 5655-F54

IBM DB2 Automation Tool for z/OS

Continuously and automatically coordinates the execution of DB2 tools, so you can realize the full potential of your DB2 system. Capabilities include the automatic execution of DB2 tools against specified objects; manual, periodic or rules-based execution of any number of tools; easy creation of job specifications; and development of job profiles through intuitive interactive system productivity facility (ISPF) panels.

Program number: 5697-G63

IBM DB2 Bind Manager

Determines if a bind is required after an application has been precompiled. If not, the tool automatically resets the time stamp and bypasses the bind. The DBRM Checker function handles consistency checking between an existing DBRMLIB and a DB2 subsystem. The Path Checker function lets you quickly determine whether a bind of a DBRM will result in a changed access path.

Program number: 5655-D38

IBM DB2 Change Accumulation Tool for z/OS

Restores database objects in the most precise and least disruptive manner for point-in-time recovery of database objects. The tool allows recovery routines to focus on single objects and previous states, and it produces SHRLEVEL REFERENCE image copies without the associated overhead and data locking.

Program number: 5655-F55

IBM DB2 DataPropagator for OS/390

Lets you maintain consistent copies of relational data across the DB2 family of databases, and automatically capture and apply data changes.

Program number: 5655-E60

IBM DB2 Forms for OS/390

A multipurpose table editing environment that offers direct update and data creation operations on DB2 UDB for OS/390 databases from within Java or Windows-based interfaces. New features include enhanced data editing and referential integrity capabilities, a new full screen table editing interface and new form components. See 66, a follow-on product to IBM DB2 Forms for OS/390.

Program number: 5697-G52

IBM DB2 High Performance Unload for OS/390

Performs sequential reading and accessing of DB2 data at top speed. DB2 HP Unload can scan a table space and create output files in the format you need. It also lets you unload partitioned table spaces in parallel.

Program number: 5655-E69

IBM DB2 Log Analysis Tool for OS/390

Ensures high availability and complete control over data integrity. You can monitor and locate data changes, easily reverse undesired changes, and carry out re-do operations with precision.

Program number: 5655-E66

IBM DB2 Object Comparison Tool for z/OS

Compare objects and dependent objects from one source to those in another, so your test and development database system can be a mirror image of the production system. The tool runs as an extension to DB2 Administration Tool for OS/390, Version 2 Release 1. The tool includes a DB2 catalog extract function, a batch job generator, a dynamic link library (DDL) extract feature, an ISPF interface, and a batch compare capability.

Program number: 5697-G64

IBM DB2 Object Restore Tool for OS/390

Automatically restores previously dropped objects and all related dependencies. This tool saves DASD space because it eliminates the need for a duplicate shadow copy of the catalog to recover objects.

Program Number: 5655-E72

IBM DB2 Performance Monitor for OS/390

Helps you monitor, analyze and optimize your DB2 system performance. You can now take advantage of Data Sharing (Sysplex) Monitoring Online with group scope view, dynamic SQL statement cache monitoring, and an API that easily retrieves performance data and passes it on to an application program.

Program number: 5655-E61

IBM DB2 Query Monitor for OS/390

Helps you maximize DB2 availability. DB2 Query Monitor lets you determine what data is gathered during activity monitoring, when it is gathered, what database resources are required, and what kind of alerts or corrective actions to take.

Program number: 5655-E67

IBM DB2 Recovery Manager for OS/390

Simplifies and coordinates the recovery of both DB2 and IMS data to a common point, cutting the time and cost of data recovery and availability. DB2 Recovery Manager eliminates the error-prone complexity of managing different logs, utilities, and processes to do recovery from both databases.

Program Number: 5697-F56

IBM DB2 Row Archive Manager for OS/390

Saves storage, improves performance and reduces the overall costs of your DB2 environment by providing a simple method to control the separation of aged data from active DB2 data.

Program number: 5655-E65

IBM DB2 SQL Performance Analyzer for OS/390

Delivers performance analysis for all phases of database application design and development. DB2 SQL Performance Analyzer helps you find out how long queries will take and aids in preventing queries from running too long.

Program number: 5697-F57

IBM DB2 Table Editor for z/OS

Was preceded by DB2 Forms. This multipurpose table-editing environment offers enterprise-wide, direct update and data creation operations on DB2 Universal Database for z/OS from within Java, , Microsoft Windows or ISPF interfaces. Database administrators and developers can conduct robust table editing in full-screen, multirow, single row, or wizard mode across multiple end-user environments and DB2 databases.

Program number: 5697-G65

IBM DB2 Utilities Suite for OS/390 and z/OS

Is comprised of DB2 Operational Utilities and DB2 Diagnostic and Recovery Utilities. This suite extends the capabilities of the core DB2 for OS/390 and z/OS database utilities with advanced functions once available only through independent software vendors. The latest functional enhancements include object wildcarding and dynamic allocation, as well as a new UNLOAD utility and improvements to the LOAD and Online REORG utilities.

DB2 Utilities Suite Program number: 5697-E98

DB2 Operational Utilities Program number: 5655-E63

DB2 Diagnostic and Recovery Utilities Program number: 5655-E62

IBM DB2 Web Query Tool for OS/390

Allows end users and administrators to access enterprise data with speed, reliability and simplicity. The tool enables pervasive connectivity over the Internet to every desktop.

Program number: 5655-E71

For more information about these tools and how to order them, see the Data Management Tools Web site:

www.ibm.com/software/data/db2imstools

Chapter 7. Planning for migration and fallback

This chapter contains considerations for migration and fallback, and a directory of new and revised installation panels. You can migrate to Version 7 from Version 5 or Version 6. See *DB2 Installation Guide* for complete, step-by-step instructions for installing, migrating, or falling back. This chapter contains these sections:

- “Migration considerations (Version 5 to Version 7)”
- “Preparing for fallback to Version 5” on page 74
- “Migration considerations (Version 6 to Version 7)” on page 78
- “Preparing for fallback to Version 6” on page 81
- “Release incompatibilities” on page 83
- “Release coexistence” on page 93
- “Installation changes” on page 94

Migration considerations (Version 5 to Version 7)

This section includes items to consider before migrating to DB2 Version 7 from Version 5.

Make sure that your current subsystem is at the proper service level. See the *Version 7 IBM Database 2 Program Directory*, which is shipped with the product, for keyword specifications for preventive service planning (PSP). Check Information/Access or the ServiceLink facility of IBMLink for PSP information both before you migrate. Also check those facilities monthly to obtain the most current information about DB2.

Scrollable cursors are supported

Support for scrollable cursors enables random access to data in a table. Temporary result tables are stored in declared temporary tables. You must allow sufficient storage space for the new TEMP database and segmented table spaces.

Unicode support

If you plan to use Unicode, all members of a data-sharing group must convert to Version 7.

Utility enhancements

The online REORG performance has been enhanced. The performance improvements are explained in Part 5 (Volume 2) of *DB2 Administration Guide*. The enhancements use new calculations for the sort and unload data sets. You might need to increase the size of your unload and sort data sets. The new calculations are explained in *DB2 Utility Guide and Reference*.

More than 32 000 databases are supported

The maximum number of databases is no longer 32 000. The database identifier column of catalog table SYSIBM.SYSDATABASE can contain negative numbers to indicate that more than 32 000 databases are defined.

Increased maximum number of data sets open

The maximum number of concurrently allocated data sets increases to 32 767 for installations that are running OS/390 Version 2 Release 6. The practical limit for concurrently allocated data sets depends on virtual storage that is below the 16 M

line, OS/390 allocation control blocks, and some DB2 storage. For more details, refer to Part 5 (Volume 2) of *DB2 Administration Guide*.

Log buffer size increased

The maximum log output buffer size is 100 000 4-KB buffers (400 MB). The input read buffer size is increased to 60 KB. You will probably experience better log read and log write performance with these increases.

Change in recording soft errors in SYS1.LOGREC data set

In previous versions of DB2, abends that occur as a result of errors in SQL statements were handled by DB2 recovery routines and were recorded in SYS1.LOGREC. Examples of these errors include:

- An error in decimal arithmetic
- Overflow error
- Underflow error

In Version 7, by default, these abends are not recorded in SYS1.LOGREC. If you want these errors to be recorded in SYS1.LOGREC, specify YES in the RECORD SOFT ERRORS field of installation panel DSNTIPM.

Revoking SYSADM authorization

If an authorization ID with SYSADM authority grants USAGE on a JAR (Java ARchive) in Version 7, an attempt to revoke SYSADM from this ID in a prior release will fail and message DSNT501I will be issued. SYSADM authority can be revoked only from Version 7.

Changed messages from the LOAD utility

In prior releases of DB2, if the data types of input data and the column into which the data was loaded were incompatible, DB2 issued message DSNU390I during the UTILITY phase of utility processing and terminated the utility with return code 8. In Version 7, if the data types are incompatible, DB2 issues message DSNU299I during the RELOAD phase. If LOAD is not performing discard processing, LOAD abnormally terminates with reason code 00E40323. If LOAD is performing discard processing, LOAD issues message DSNU299I for every row of input data and puts the rejected records in the discard file. LOAD then terminates with return code 8.

Encoding schemes of string parameters for routines

The new PARAMETER CCSID clause allows you to define the encoding scheme of all string parameters for stored procedures and system-defined parameters at once. In prior versions, you had to define a CCSID for each string parameter if you wanted an encoding scheme other than the default. Also, EBCDIC is no longer the default encoding scheme for system-defined parameters. DB2 now uses the same encoding scheme for both user-specified and system-generated string parameters.

Increased size for generated code

In DB2 Version 7, you can specify CCSIDs for host variables. Because DB2 stores the CCSIDs with the host variables, the generated source code and object code for a DB2 application might be larger than in prior versions.

Consider increasing IDBACK and CTHREAD

Because utilities might use additional threads, you should consider increasing the IDBACK and CTHREAD subsystem parameters. Increasing these parameters can

help you avoid failure of some utilities due to increased thread usage. Increasing these parameters also allows additional parallelism to be performed by the utilities.

Customized DB2I defaults can be migrated

A DB2I TSO ISPF profile member from a prior release can be migrated to the current release. The DSNEMC01 CLIST uses the values that are specified on installation panel DSNTIPF and stores the results in the ISPF profile member DSNEPROF. Any customized DSNEPROF members can be migrated from Version 5 to Version 7. However, you need to examine any new or changed default panel values to ensure that your customized values are still valid.

Migrating a data sharing group

Migration of a data sharing group requires careful planning:

1. Read the information about migration considerations in this book and also in Chapter 3 of *DB2 Data Sharing: Planning and Administration*.
2. Plan to migrate the data sharing group in as short a time as possible.
3. Apply the fallback SPE to the Version 5 load library for each member in the data sharing group. For best availability, you can apply the SPE to one member at a time. You can have Version 5 systems with and without the SPE that run . Stop and restart each member to receive the change. Before migrating the first member of the data sharing group to Version 7, you must apply the fallback SPE to all members in the group.
4. Follow the procedure on migrating the data sharing group in Chapter 3 of *DB2 Data Sharing: Planning and Administration*. You must completely migrate the first member of the data sharing group to Version 7 before starting any other members at the Version 7 level.
5. To prepare for fallback, keep the subsystem parameter load module that is used by Version 5.

The CLIST edits different jobs for enabling data sharing and migrating a data sharing member. See Chapter 3 of *DB2 Data Sharing: Planning and Administration* for the list of jobs that are edited for data sharing and migration.

Creating tables with DBCS and mixed columns

You can no longer create EBCDIC tables with GRAPHIC, VARGRAPHIC, or DBCLOB columns when the setting for installation option MIXED DATA is NO. Similarly, you cannot alter EBCDIC tables to add GRAPHIC, VARGRAPHIC, DBCLOB columns when MIXED DATA is set to NO.

Consider enlarging the BSDS

The BSDS might need to be larger to accommodate additional buffer pools. To prevent the BSDS from going into secondary extents, change the record size of the primary allocation to 180 records. To increase the space allocation for the BSDS:

1. Rename existing BSDSs.
2. Define larger BSDSs with the original names.
3. Copy the renamed BSDSs into the new ones.

You can use access method services to enlarge the BSDS. See the Version 7 installation job, DSNTIJIN, for the definition that uses the larger primary extent size.

Stored procedures

In prior releases of DB2 the AUTHID and LUNAME columns of catalog table SYSIBM.SYSPROCEDURES were used to uniquely identify multiple instances of a

procedure. After migrating to Version 7, use the SCHEMA column in the SYSIBM.SYSROUTINES catalog table, the CURRENT PATH special register, and the PATH bind option to identify multiple instances of a procedure. During migration, DB2 generates CREATE PROCEDURE statements that populate SYSIBM.SYSROUTINES and SYSIBM.SYSPARMS. Rows in SYSIBM.SYSROUTINES that contain nonblank values for columns AUTHID or LUNAME are not used to generate the CREATE PROCEDURE statements. You can identify those rows by using the following statement:

```
SELECT * FROM SYSIBM.SYSPROCEDURES WHERE AUTHID <> ' ' OR LUNAME <> ' ';
```

DB2 also copies rows from SYSIBM.SYSPROCEDURES into table SYSIBM.SYSPARMS and propagates information from the PARMLIST column of SYSIBM.SYSPROCEDURES into SYSIBM.SYSROUTINES and SYSIBM.SYSPARMS.

Procedures that are migrated from Version 5 do not have an owner because they were not created with the CREATE PROCEDURE statement. DB2 authorization treats these migrated procedures differently from procedures that are created with CREATE PROCEDURE. The authorization for migrated procedures is unchanged.

Tables SYSIBM.SYSPSM and SYSIBM.SYSPSMOPTS are not used in Version 7. Those tables exist if you are currently using Version 5 SQL Procedures support. Job DSNTIJMP migrates the data from these tables into new catalog tables.

Catalog table SYSIBM.SYSPROCEDURES is not used after Version 5 to define stored procedures to DB2. The SYSCAT.PROCEDURES view of catalog table SYSIBM.SYSPROCEDURES is not used to list the stored procedures that are registered in DB2. Although these tables are no longer used in Version 7, DB2 does not drop them during migration because they are needed for data sharing coexistence and fallback.

For more information, refer to *DB2 Application Programming and SQL Guide*.

Size calculations for the work file database

The migration job DSNTIJTC creates and updates indexes on catalog tables. These indexes are created and updated *sequentially* during migration. The work file database is used for the sort of each index; DB2 needs enough work file storage to sort the largest of the indexes in Table 4. Migration fails if you do not have enough storage, so ensure that you have enough space before you begin.

Table 4 shows the indexes that are new and changed for existing catalog tables.

Table 4. Indexes that are added or updated sequentially using the work file database

Catalog table name	Index name	Column names
SYSIBM.SYSINDEXPART	SYSIBM.DSNDRX02	STORNAME
SYSIBM.SYSCOLUMNS	SYSIBM.DSNDCX02	TYPESCHEMA, TYPENAME
SYSIBM.SYSPACKDEP	SYSIBM.DSNKDX03	BQUALIFIER, BNAME, BTYPE, DTYPE
SYSIBM.SYSTABLEPART	SYSIBM.DSNPX02	STORNAME
SYSIBM.SYSVIEWDEP	SYSIBM.DSNGGX03	BSHEMA, BNAME, BTYPE

Table 4. Indexes that are added or updated sequentially using the work file database (continued)

Catalog table name	Index name	Column names
SYSIBM.SYSTABAUTH	SYSIBM.DSNATX02	GRANTEE, TCREATOR, TTNAME, GRANTEETYPE, UPDATECOLS, ALTERAUTH, DELETEAUTH, INDEXAUTH, INSERTAUTH, SELECTAUTH, UPDATEAUTH, CAPTUREAUTH, REFERENCESAUTH, REFCOLS, TRIGGERAUTH
SYSIBM.SYSUSERAUTH	SYSIBM.DSNAUH01	GRANTEE, GRANTEDETS
SYSIBM.SYSINDEXES	SYSIBM.DSNDXX02	DBNAME, INDEXSPACE, COPY, COPYLRSN
SYSIBM.SYSVIEWS	SYSIBM.DSNVX01	CREATOR, NAME, SEQNO, TYPE
SYSIBM.SYSTRIGGERS	SYSIBM.DSNOTX03	SCHEMA, TRIGNAME
SYSIBM.SYSROUTINES	SYSIBM.DSNOFX07	NAME, PARM_COUNT, ROUTINETYPE, SCHEMA, PARM_SIGNATURE, PARM1, PARM2, PARM3, PARM4, PARM5, PARM6, PARM7, PARM8, PARM9, PARM10, PARM11, PARM12, PARM13, PARM14, PARM15, PARM16, PARM17, PARM18, PARM19, PARM20, PARM21, PARM22, PARM23, PARM24, PARM25, PARM26, PARM27, PARM28, PARM29, PARM30
SYSIBM.SYSROUTINES	SYSIBM.DSNOFX08	JARSCHEMA, JAR_ID
SYSIBM.SYSVTREE	SYSIBM.DSNVTH01	CREATOR, NAME, TYPE

Shared read-only data function is removed

The shared read-only data function is not supported in Version 7. Data sharing is a more substantial and more usable function than shared read-only data. You can also use distributed data to share information. If any shared read-only data is found during catalog migration, you will receive a warning message.

Data set password protection is removed

Data set password protection is not supported in Version 7. Remove data set passwords before migrating to Version 7. Use the RACF function of the SecureWay Security Server for OS/390 or an equivalent security system to protect your data sets. If any data set passwords are found during catalog migration, you will receive a warning message.

Private protocol function not enhanced

No enhancements are planned for distributed data that use the private protocol function. Private protocol support might be removed in a subsequent release of DB2. The private protocol function cannot use type 2 inactive threads. To use type 1 inactive threads for private protocol, specify a non-zero value for MAXTYPE1. To take advantage of the enhancements to stored procedures, TCP/IP and new data types, you must use the DRDA protocol. To use DRDA without changing your applications by rebinding with the DBPROTOCOL option set to DRDA. See Chapter 2 of *DB2 Command Reference* for more information.

Type 2 indexes are required

Type 1 indexes are not supported. You must convert all indexes to type 2 before migrating to Version 7. If any type 1 indexes are found during catalog migration, catalog migration will fail. Type 1 indexes will not be functional in Version 7.

Preparing for fallback to Version 5

You can fall back to DB2 for OS/390 Version 5 after migrating to Version 7. This
section highlights fallback considerations for Version 7. If you experience a severe
application or performance error and want to return to Version 5, follow the detailed
step-by-step instructions in *DB2 Installation Guide*.

To avoid complications, do not use the new DB2 for OS/390 and z/OS Version 7 facilities until you are certain that you will not need to fall back. You can save your Version 7 TSO LOGON procedures and JCL for remigration to Version 7.

To fall back to Version 5:

1. Run Phase 0 of the Version 7 installation verification procedure.
2. Stop DB2 Version 7 activity.
3. Reactivate Version 5.
4. Reconnect TSO, IMS, and CICS to Version 5.
5. Start Version 5.
6. Verify fallback by running the DB2 sample applications or your own applications.

After fallback, plans or packages that were bound in Version 7 will be automatically rebound the first time they run in the previous release.

If you fall back and then try to use frozen plans or packages, the automatic rebind from the previous version fails. To make the plans and packages that were not automatically rebound on the previous version available, change the SQL statements or remove the reference to a frozen object, precompile the application programs, and explicitly BIND the plans and packages on the previous version.

Frozen objects

Falling back does not undo changes that are made to the catalog during migration to Version 7. The migrated catalog is used after fallback. Some objects in this catalog that have been affected by Version 7 function might become *frozen* objects after fallback. Frozen objects are unavailable, and they are marked with the release dependency marker I, J, or K. If an object is marked with a release dependency, it is never unmarked. The release dependency marker is listed in the IBMREQD column of catalog tables.

In general, objects that depend on the new facilities of DB2 for OS/390 and z/OS Version 7 are frozen after you fall back to Version 5 and remain frozen until you remigrate to Version 7. Table 5 on page 75 lists the objects that are frozen when falling back to Version 5. Frozen objects are marked with the release dependency markers I, J, or K.

Table 5. Objects that are frozen when falling back to DB2 for OS/390 Version 5

RELEASE DEPENDENT MARK = I, J, or K
<ul style="list-style-type: none"> • Plans, packages, or views that use any new syntax or objects¹ • Tables and views with triggers • Tables with check constraints that reference a cast function • 8-KB and 16-KB table spaces and associated indexes² • Plans or packages that reference user-defined functions • Plans or packages that reference a Unicode object • Plans or packages that use any new bind options such as DBPROTOCOL(DRDA)³ • Plans or packages that have statements that use a default encoding scheme, and the default is Unicode • Trigger packages • Queries with scrollable cursor syntax • Tables with LOB columns • LOB table spaces • Auxiliary tables • Indexes on auxiliary tables • Tables with columns that have the ROWID data type • Tables that are defined with a user-defined distinct type • Tables, table spaces, or databases that are defined with the Unicode encoding scheme • Tables and table spaces that are defined with identity columns • Table spaces that contain a table with an identity column • Any table space with a DSSIZE greater than 4 GB and their associated partitioning indexes and nonpartitioning indexes with PIECESIZE greater than 4 GB

Notes:

1. If a plan or package references a declared temporary table that uses the qualifier SESSION, but does not use a DECLARE GLOBAL TEMPORARY TABLE statement, the plan or package does not become frozen.
2. The table spaces are marked with a release dependency. The indexes are not marked with a release dependency but are unavailable after fallback due to their association with the table spaces.
3. The release dependency is marked regardless of whether the DBPROTOCOL(DRDA) option is specified on bind or rebind, or is the default that is defined in the subsystem parameter.

Plans and packages become frozen objects when they use new SQL syntax, use new BIND options and attributes, or reference frozen objects. When plans and packages become frozen objects, the automatic rebind process is adversely affected. See *DB2 Installation Guide* for details.

Product-sensitive Programming Interface

While operating in Version 5, you can determine if any of your objects are frozen by issuing the following SELECT statements:

```

SELECT NAME FROM SYSIBM.SYSPLAN
  WHERE IBMREQD = 'I' OR IBMREQD = 'J' OR IBMREQD = 'K';
SELECT LOCATION, COLLID, NAME, VERSION FROM SYSIBM.SYSPACKAGE
  WHERE IBMREQD = 'I' OR IBMREQD = 'J' OR IBMREQD = 'K';
SELECT CREATOR, NAME FROM SYSIBM.SYSVIEWS
  WHERE IBMREQD = 'I' OR IBMREQD = 'J' OR IBMREQD = 'K';
SELECT CREATOR, NAME FROM SYSIBM.SYSINDEXES
  WHERE IBMREQD = 'I' OR IBMREQD = 'J' OR IBMREQD = 'K';
SELECT CREATOR, NAME, TYPE FROM SYSIBM.SYSTABLES
  WHERE IBMREQD = 'I' OR IBMREQD = 'J' OR IBMREQD = 'K';
SELECT DBNAME, NAME FROM SYSIBM.SYSTABLESPACE
  WHERE IBMREQD = 'I' OR IBMREQD = 'J' OR IBMREQD = 'K';

```

Other fallback considerations

Before you fall back to Version 5, you must be aware of the following considerations:

Key constraints: If DROP INDEX is run on Version 5, and the index enforces a unique key constraint, the unique key constraint is dropped when the index is dropped. This is true even if the unique key was created on Version 7.

ALTER partition: You should remove all restrictive states from a table space before falling back. However, if fallback occurs while a partition of a table space is in REORG PENDING (REORP) state, the entire table space is unavailable. You must run REORG TABLESPACE (SHRLEVEL NONE) on the entire table space from the fallback release to remove the REORP status and rebalance the partitions. After you fall back to Version 5, you cannot use the ALTER INDEX statement to modify the limit key boundaries. You also cannot specify a range of partitions by using the REORG utility.

Changes to the MODIFY utility: You should alter indexes with the COPY NO option before falling back to Version 5. In Version 5, you cannot run the MODIFY RECOVERY utility on table spaces with indexes that have the COPY YES option.

Stored procedures: If you created stored procedures with the Version 7 CREATE PROCEDURE statement, you cannot access those stored procedures if you fall back to Version 5. SQLCODE -204 is returned if you try to access a Version 7 stored procedure after fallback because the procedure name is not in catalog table SYSIBM.SYSPROCEDURES.

No predictive governing: You can use the Version 7 RLST after falling back, but you cannot take advantage of the predictive governing features. Rows with RLFFUNC = 6 or 7 are ignored.

LOB considerations: If any log records are found for LOBs, the applicable pages for the LOB table space are placed in the LPL.

Optimization hints: You can still use your Version 7 PLAN_TABLE after falling back, but you cannot take advantage of the optimization hints that you have added to the PLAN_TABLE.

Utilities COPY, REPORT, and RECOVER: You must use the Version 5 COPY and RECOVER utility jobs for backup and recovery after fallback. You can use the REBUILD INDEX utility because of an alias. SYSIBM.SYSCOPY and SYSIBM.SYSLGRNX rows that are inserted for indexes in Version 7 are ignored by backup and recovery processing after fallback.

UNLOAD utility: You cannot stop or start an UNLOAD utility job from Version 5. You also cannot issue the TERM utility command to a stopped UNLOAD utility job from a release prior to Version 7. You should terminate stopped UNLOAD utilities by using the TERM utility command before fallback occurs; otherwise, SYSUTIL records for those UNLOAD utility jobs remain in the system.

More consistent DB2 restart times: Falling back causes postponed abort units of recovery (URs) to be treated as inabort or inflight. Their backout is performed as

part of the first restart in Version 5 and all page sets are removed from their restart-pending (RESTP) state or advisory restart-pending (AREST) state. If outstanding backout work exists on the page set on another data sharing member, the pending states are not removed. Therefore, you should try to resolve postponed abort URs with the command RECOVER POSTPONED before falling back.

A Version 5 subsystem cannot request limited backout processing. However, if you enter a DISPLAY DATABASE command on Version 5, that display might show the advisory restart-pending (AREST) state of page sets that can result when Version 7 uses postponed abort processing. Similarly, if you issue a DISPLAY GROUP command on a Version 5 subsystem, a member that is both active and has indoubt URs is displayed as ACTIVE. (In a Version 7 subsystem they are displayed as 'A I'.) The 'A I' status means the object is active but has indoubt or postponed abort units of recovery outstanding. On fallback to Version 5, the 'A I' status is changed to ACTIVE even if indoubt URs exist. However, on remigration to Version 7, the ACTIVE status is changed back to 'A I' if the indoubt URs still exist.

New buffer pool sizes: A CREATE or ALTER DATABASE or TABLESPACE statement that specifies an 8-KB or 16-KB buffer pool name does not work after fallback. If BUFFERPOOL is not specified on the CREATE TABLESPACE statement and the default buffer pool for the database was an 8-KB or 16-KB buffer in Version 7, SQLCODE -204 is returned.

New buffer pool type: If a buffer pool is defined as VPTYPE(DATASPACE) in Version 7, and you fall back to Version 5, this buffer pool becomes a primary type buffer pool. It is allocated with the specified VPSIZE unless the VPSIZE is larger than the allowable 1.6 GB, in which case the default VPSIZE is used. The BSDS is not updated during fallback so that the old values can be used on remigration. On remigration to Version 7, the original VPTYPE and VPSIZE are used. However, if you alter these attributes using ALTER BUFFERPOOL while in fallback mode on Version 5, the BSDS is updated and the Version 7 attributes are lost.

New CLUSTERRATIOF column: When you fall back to Version 5, the value in CLUSTERRATIO is used. If you run RUNSTATS on Version 5, the CLUSTERRATIOF column is reset to the default value. When you remigrate to Version 7, DB2 uses CLUSTERRATIO until RUNSTATS is run on Version 7.

Changing column length in an index: Changing the length of a column of an index in Version 7 can cause an error after falling back to Version 5. A resource unavailable message with return code 00C9009E occurs if the APAR to enable immediate index access has not been applied.

Changes to cascading REVOKE privileges: When you fall back to Version 5 and attempt to revoke SYSADM or SYSCTRL privileges for a grantor, and that revoke cascades down to Version 7 objects or new privilege grants on Version 5 objects, the revoke fails with message DSNT501I.

Changes to IMMEDIATE function: If a Version 7 plan or package that is bound with IMMEDIATE(YES | PH1) is executed on a DB2 Version 5 member that does not have support for the IMMEDIATE (YES | PH1) attribute, the IMMEDIATE(YES | PH1) attribute is ignored at run time. The Version 5 plan or package uses the default IMMEDIATE(NO) behavior without needing to be rebound.

Table and index spaces with undefined data sets: You can use the DEFINE NO option on the CREATE TABLESPACE or CREATE INDEX statements to defer

creating the VSAM data set until it is used. If the underlying VSAM data sets have not been created when you fall back to Version 5, you will receive a -904 SQLCODE for any SQL statements that refer to the table spaces and index spaces with undefined data sets. Do not use the REPAIR DBD or STOSPACE utilities on the undefined data sets.

For more information on fallback considerations, refer to *DB2 Installation Guide*.

Migration considerations (Version 6 to Version 7)

This section includes items to consider before migrating to DB2 Version 7 from Version 6.

Make sure that your Version 6 subsystem is at the proper service level. The
recommended maintenance level is the DB2 UDB for OS/390 Version 6 refresh
tape. An equivalent service level can be obtained by ordering a service update via
PDO (level 0014) or ESO (level 0003) and applying service to PUT 0003. Refer to
IBM Database 2 Program Directory (shipped with the product) for keyword
specifications for preventive service planning (PSP). Check Information/Access or
the ServiceLink facility of IBMLink for PSP information before you migrate. Also
check those facilities monthly to obtain the most current information about DB2.

Be aware of the following changes that might affect your migration to Version 7.

Unicode support

If you plan to use Unicode, all members of a data-sharing group must convert to Version 7.

New default encoding scheme

In Version 6, the default encoding scheme for host variables in the SET host variable and VALUES INTO statements was EBCDIC. In Version 7, the default encoding scheme is the value in the APPLICATION ENCODING SCHEME bind option.

Scrollable cursors are supported

Support for scrollable cursors enables random access to data in a table. Temporary result tables are stored in declared temporary tables. You must allow sufficient storage space for the new TEMP database and segmented table spaces.

Revoking SYSADM authorization

If an authorization ID with SYSADM authority grants USAGE on a JAR (Java ARchive) in Version 7, an attempt to revoke SYSADM from this ID in a prior release will fail and message DSNT501I will be issued. SYSADM authority can be revoked only from Version 7.

Creating tables with DBCS and mixed columns

You can no longer create EBCDIC tables with GRAPHIC, VARGRAPHIC, or DBCLOB columns when the setting for installation option MIXED DATA is NO. Similarly, you cannot alter EBCDIC tables to add GRAPHIC, VARGRAPHIC, or DBCLOB columns when MIXED DATA is set to NO.

Changed messages from the LOAD utility

In prior releases of DB2, if the data types of input data and the column into which the data was loaded were incompatible, DB2 issued message DSNU390I during the UTILITY phase of utility processing and terminated the utility with return code 8. In Version 7, if the data types are incompatible, DB2 issues message DSNU299I during the RELOAD phase. If LOAD is not performing discard processing, LOAD abnormally terminates with reason code 00E40323. If LOAD is performing discard processing, LOAD issues message DSNU299I for every row of input data and puts the rejected records in the discard file. LOAD then terminates with return code 8.

Change in recording soft errors in SYS1.LOGREC data set

In prior versions of DB2, abends that occur as a result of errors in SQL statements were handled by DB2 recovery routines and were recorded in SYS1.LOGREC.

Examples of these errors include:

- An error in decimal arithmetic
- Overflow error
- Underflow error

In Version 7, by default, these abends are not recorded in SYS1.LOGREC. If you want these errors to be recorded in SYS1.LOGREC, you can specify YES in the RECORD SOFT ERRORS field of installation panel DSNTIPM.

Consider increasing IDBACK and CTHREAD

Because utilities may use additional threads, you should consider increasing the IDBACK and CTHREAD subsystem parameters. Increasing these parameters can help you avoid failure of some utilities due to increased thread usage. An increase also supports the additional parallelism to be performed by the utilities.

Increased size for generated code

In DB2 Version 7, you can specify CCSIDs for host variables. Because DB2 stores the CCSIDs with the host variables, the generated source code and object code for a DB2 application might be larger than in prior versions.

Encoding schemes of string parameters for routines

The new parameter CCSID clause allows you to define the encoding scheme of all string parameters for user-defined functions and stored procedures at once. In prior versions, you had to define a CCSID for each string parameter if you wanted an encoding scheme other than the default. Also, EBCDIC is no longer the default encoding scheme for system-defined parameters. DB2 now uses the same encoding scheme for both user-specified and system-generated string parameters.

Customized DB2I defaults can be migrated

A DB2I TSO ISPF profile member from a prior release can be migrated to the current release. The DSNEMC01 CLIST uses the values that are specified on installation panel DSNTIPF and stores the results in the ISPF profile member DSNEPROF. Any customized DSNEPROF members can be migrated from Version 6 to Version 7. However, you need to examine any new or changed default panel values to ensure that your customized values are still valid.

Migrating a data sharing group

Migrating a data sharing group requires a careful plan:

1. Read the information about migration considerations in this book and also in Chapter 3 of *DB2 Data Sharing: Planning and Administration*.
2. Plan to migrate the data sharing group over as short a time as possible.
3. Apply the fallback SPE to the Version 6 load library for each member in the data sharing group. For best availability, you can apply the SPE to one member at a time. You can have Version 6 systems with and without the SPE that run . Stop and restart each member to pick up the change. Before migrating the first member of the data sharing group to Version 7, the fallback SPE must be applied to all members in the group.
4. Follow the procedure on migrating the data sharing group in Chapter 3 of *DB2 Data Sharing: Planning and Administration*. You must migrate the first member of the data sharing group to Version 7 before starting any other members at Version 7.
5. To prepare for fallback, keep the subsystem parameter load module used by Version 6.

The CLIST edits different jobs for enabling data sharing and migrating a data sharing member. See Chapter 3 of *DB2 Data Sharing: Planning and Administration* for the list of jobs that are edited for each case.

Work file database size calculations

The migration job DSNTIJTC creates and updates indexes on catalog tables. These indexes are created and updated *sequentially* during migration. The work file database is used for the sort of each index; DB2 needs enough work file storage to sort the largest of the indexes in Table 6. The migration fails if you do not have enough storage, so ensure that you have enough space before you begin.

Table 6 shows the indexes that are new and changed for existing catalog tables.

Table 6. Indexes that are added or updated sequentially using the work file database

Catalog table name	Index name	Column names
SYSIBM.SYSTRIGGERS	SYSIBM.DSNOTX03	SCHEMA, TRIGNAME
SYSIBM.SYSROUTINES	SYSIBM.DSNOFX07	NAME, PARM_COUNT, ROUTINETYPE, SCHEMA, PARM_SIGNATURE, PARM1, PARM2, PARM3, PARM4, PARM5, PARM6, PARM7, PARM8, PARM9, PARM10, PARM11, PARM12, PARM13, PARM14, PARM15, PARM16, PARM17, PARM18, PARM19, PARM20, PARM21, PARM22, PARM23, PARM24, PARM25, PARM26, PARM27, PARM28, PARM29, PARM30
SYSIBM.SYSROUTINES	SYSIBM.DSNOFX08	JARSCHEMA, JAR_ID
SYSIBM.SYSVIEWS	SYSIBM.DSNVX01	CREATOR, NAME, SEQNO, TYPE
SYSIBM.SYSVTREE	SYSIBM.DSNVTH01	CREATOR, NAME, TYPE

SYSIBM.SYSPSM and SYSIBM.SYSPSMOPTS no longer used

Tables SYSIBM.SYSPSM and SYSIBM.SYSPSMOPTS are no longer used. Job DSNTIJMP migrates the data from these tables into new catalog tables.

Preparing for fallback to Version 6

You can fall back to DB2 for OS/390 Version 6 after migrating to Version 7. This
section highlights fallback considerations for Version 7. If you experience a severe
application or performance error and want to return to Version 6, follow the detailed
step-by-step instructions in *DB2 Installation Guide*.

To avoid complications, do not use the new DB2 for OS/390 and z/OS Version 7 facilities until you are certain that you will not need to fall back. You can save your Version 7 TSO LOGON procedures and JCL for remigration to Version 7.

To fall back to Version 6:

1. Run Phase 0 of the Version 7 installation verification procedure.
2. Stop DB2 Version 7 activity.
3. Reactivate Version 6.
4. Reconnect TSO, IMS, and CICS to Version 6.
5. Start Version 6.
6. Verify fallback by running the DB2 sample applications or your own applications.

After fallback, plans or packages that were bound in Version 7 will be automatically rebound the first time they run in Version 6.

If you fall back and then try to use frozen plans or packages, the automatic rebound in the previous version fails. To make the plans and packages that were not automatically rebound on Version 6 available:

1. Change the SQL statements or remove the reference to a frozen object.
2. Precompile the application programs.
3. Explicitly BIND the plans and packages on Version 6.

Frozen objects

Falling back does not undo changes that were made to the catalog during migration to Version 7. The migrated catalog is used after fallback. Some objects in this catalog that have been affected by Version 7 function might become *frozen* objects after fallback. Frozen objects are unavailable, and they are marked with the release dependency marker K. If an object is marked with a release dependency, it is never unmarked. The release dependency marker is listed in the IBMREQD column of catalog tables.

In general, objects that depend on the new facilities of DB2 for OS/390 and z/OS Version 7 are frozen after you fall back to Version 6 until you remigrate to Version 7. Table 7 on page 82 lists the objects that are frozen when falling back to Version 6. Frozen objects are marked with the release dependency marker K.

Table 7. Objects that are frozen when falling back to DB2 for OS/390 Version 6

RELEASE DEPENDENT MARK = K
<ul style="list-style-type: none">• Plans, packages, or views that use any new syntax or objects¹• Queries with scrollable cursor syntax• Plans or packages that use any new bind options such as DBPROTOCOL(DRDA)²• Plans or packages that reference a Unicode object• Plans or packages that have statements that use a default encoding scheme, and the default is Unicode• Stored procedures or user-defined functions with parameters that explicitly or implicitly use the Unicode encoding scheme• Distinct types for which the base type is a string with the Unicode encoding scheme• Tables, table spaces, or databases that are defined with the Unicode encoding scheme• User-defined functions and stored procedures that are written in Java• User-defined functions that are written in SQL
Notes: <ol style="list-style-type: none">1. If a plan or package references a declared temporary table that uses the qualifier SESSION, but does not use a DECLARE GLOBAL TEMPORARY TABLE statement, the plan or package does not become frozen.2. The release dependency is marked regardless of whether the DBPROTOCOL(DRDA) option is specified on bind or rebind, or is the default that is defined in the subsystem parameter.

Plans and packages become frozen objects when they use new SQL syntax, use new BIND options and attributes, or reference frozen objects. When plans and packages become frozen objects, the automatic rebind process is adversely affected.

Product-sensitive Programming Interface

While operating in Version 6, you can determine if any of your objects are frozen by issuing the following SELECT statements:

```
SELECT NAME FROM SYSIBM.SYSPLAN
  WHERE IBMREQD = 'K';
SELECT LOCATION, COLLID, NAME, VERSION FROM SYSIBM.SYSPACKAGE
  WHERE IBMREQD = 'K';
SELECT CREATOR, NAME FROM SYSIBM.SYSVIEWS
  WHERE IBMREQD = 'K';
SELECT CREATOR, NAME FROM SYSIBM.SYSINDEXES
  WHERE IBMREQD = 'K';
SELECT CREATOR, NAME, TYPE FROM SYSIBM.SYSTABLES
  WHERE IBMREQD = 'K';
SELECT DBNAME, NAME FROM SYSIBM.SYSTABLESPACE
  WHERE IBMREQD = 'K';
```

End of Product-sensitive Programming Interface

Other fallback considerations

Before you fall back to Version 6, you must be aware of the following considerations:

Key constraints: If DROP INDEX is run on Version 6, and the index enforces a unique key constraint, the unique key constraint is dropped when the index is dropped. This is true even if the unique key was created on Version 7.

Utilities COPY, REPORT, and RECOVER: You must use the Version 6 COPY and RECOVER utility jobs for backup and recovery after fallback. You can use the REBUILD INDEX utility because of an alias. SYSIBM.SYSCOPY and SYSIBM.SYSLGRNX rows inserted for indexes in Version 7 are ignored by backup and recovery processing after fallback.

UNLOAD utility: You cannot start or stop an UNLOAD utility job from Version 6. You also cannot issue the TERM utility command to a stopped UNLOAD utility job from a release prior to Version 7. You should terminate topped UNLOAD utilities by using the TERM utility command before fallback occurs; otherwise, SYSUTIL records for those UNLOAD utility jobs remain in the system.

For more information on fallback considerations, refer to *DB2 Installation Guide*.

Release incompatibilities

This section describes changes that might affect your DB2 operations after migrating to Version 7 of DB2.

Release incompatibilities (migration from Version 5)

DCE security authentication no longer supported

DCE support has been removed. Kerberos authentication has replaced DCE. For more information about creating Resource Access Control Facility (RACF) profiles, see Part 3 (Volume 1) of *DB2 Administration Guide*.

Implicit restart of utilities

The subsystem parameter UTLRSTRT in macro DSN6SPRM controls whether online restartable utilities can be restarted without having to specify the RESTART keyword. If you specify OFF, the default value, you must explicitly specify RESTART if you want to restart a restartable utility. Utilities are restarted from the beginning, where appropriate.

Fewer sort operations for queries

A performance enhancement to DB2 results in fewer sort operations for queries that have ORDER BY clauses and have WHERE clauses with predicates of the form COL=constant. This enhancement can result in the following changes from prior releases of DB2:

- Changes in access paths, which can result in changes to EXPLAIN output or the elapsed time of a query.
- Some queries that previously received SQL code -136 (sort key length exceeds maximum) may now execute successfully.

Sysplex query parallelism in the PLAN_TABLE

If a parallel group being rebound to Version 7 is capable of Sysplex query parallelism, and only one DB2 data sharing member is active, a single-CEC parallel plan is developed, and an **X** is placed in column PARALLEL_MODE of table PLAN_TABLE instead of the **C** that was used in Version 5. If you do not want the **X**, bind the plan or package on a member that is installed with COORDINATOR=NO. The **X** for column PARALLEL_MODE means that all available assisting DB2s are used at run time. You need to rebind all static plans that have a PARALLEL_MODE of X after migrating to take advantage of this change.

#

Limit backouts with system restarts

There is a release incompatibility by using the AUTO setting of LIMIT BACKOUT, selected from installation panel DSNTIPL. The purpose of the setting is so that you can indicate that you want DB2 to postpone some of the backout work that is usually performed during system restart.

Recommendation: Use the default setting, AUTO, but be aware that after completing restart, some objects (table spaces, index spaces or partitions) might be unavailable until the automatically started process completes the backout work. In releases prior to Version 6, all of DB2 was unavailable until the backout work was completed.

ALTER INDEX syntax

After Version 5, ALTER INDEX is more flexible in allowing multiple PART clauses to be specified in a single ALTER INDEX statement. Existing ALTER INDEX statements that specify partition-level options before the PART keyword have different results in Version 7 than in prior releases. In Version 7, when these options are specified before the PART keyword, they apply to all partitions in the index unless you override them with a PART specification.

For example, assume that you have the following statement to change the FREEPAGE value to 10 for partition 1:

```
ALTER INDEX ixname
  FREEPAGE 10
  PART 1;
```

In Version 7, this same statement changes the FREEPAGE value to 10 for all index partitions. If you want to obtain the same results for this statement as you did in prior releases, change the statement as follows:

```
ALTER INDEX ixname
  PART 1
  FREEPAGE 10;
```

For the ALTER INDEX statement syntax, see *DB2 SQL Reference*.

RECOVER INDEX becomes REBUILD INDEX

Before migrating to Version 7, ensure that you have applied APAR PQ09842, which enables you to change your RECOVER INDEX jobs to use the new syntax REBUILD INDEX. In Version 7, RECOVER INDEX means to use a copy of the index and apply log records. Thus, your old RECOVER INDEX jobs are likely to fail. Change any existing RECOVER INDEX utility jobs to use REBUILD INDEX. See APAR PQ09842 for more information.

Change to parameter in IRLMPROC startup procedure

The GROUP parameter in the IRLMPROC startup procedure has been changed to IRLMGRP. Be sure to use the IRLMGRP parameter instead of the GROUP parameter to prevent JCL errors during IRLM startup.

Default for FASTSWITCH parameter is YES

For some utilities, the FASTSWITCH parameter has a default of YES, which results in the creation of J0001 data sets. Because some tools and other utilities require data set names to match, you may experience incompatibilities. The fifth-level qualifier can vary from I0001 to J0001. For more information about renaming data sets, see Part 2 (Volume 1) of DB2 Administration Guide.

#

Adjust application programs

You might need to adjust your application programs because of the release incompatibilities that this section describes. This list applies only if you are migrating from Version 5. If you are migrating from Version 6, see “Adjust application programs” on page 90.

No colon on a host variable is an error: All host variable references must have the leading colon. If you neglect to use a colon to designate a host variable, the precompiler issues a DSNH104I message or interprets the host variable as an unqualified column name, which might lead to unintended results. The host variable without a colon is interpreted as a column name when the host variable is referenced in a context in which a column name can also be referenced.

To find out if you have host variables that are not preceded with a colon, precompile again in Version 5 and check for warning message DSNH315I. You can correct the problem by inserting colons in the source code, precompiling, and binding the new DBRM. If you do not have source code, read APARs II12100, PQ26922, and PQ30390. For more information about detecting missing colons on host variables and correcting the problem, see DB2 UDB Server for OS/390 Version 6 Technical Update, SG24-6108-00.**SQLCODE -101:** After migrating to DB2 Version 7, you might receive SQLCODE -101 on long or complicated SQL statements that previously executed successfully. This is possible because SQL statements and DB2 internal structures are buffered in the same local storage. Release changes in the internal structures can result in less available storage for the SQL statements.

Rewrite the unsuccessful SQL statements by using correlated references, breaking up unions or using outer joins.

SQLCODE -538: After migrating to DB2 Version 7, you might receive SQLCODE -538 on an SQL CREATE TABLE FOREIGN KEY or SQL ALTER TABLE ADD FOREIGN KEY statement that previously executed successfully. If the foreign key references a parent key that is not defined as a primary key or unique key in the parent table, the creation of the foreign key will fail. You must ensure that the parent key has been created as a primary key or unique key in the parent table.

SQLCODE -669: After migrating to DB2 Version 7, you might receive SQLCODE -669 on an SQL DROP INDEX statement that previously executed successfully. If the dropped index is a unique index used to enforce a unique (primary or unique key) constraint or referential constraint drop the constraint before you can drop the index.

You can drop a unique index (for a unique key only) without first dropping the unique key constraint if the unique key was created in a release of DB2 prior to Version 7, and the unique key has no associated referential constraints.

#

Bind plans and packages: Before DB2 Version 7, you did not need to bind a plan or a package in the following situations:

- For distributed applications, you did not need to bind a package at the requester for statements that were processed at the requester.
- For local applications, you did not need to bind a plan, or a package and a plan, if a program contained ONLY statements that are processed at the requester.

Appendix B of DB2 SQL Reference indicates which SQL statements are processed at the requester. Examples of those statements are CONNECT, COMMIT, ROLLBACK, and RELEASE.

As of DB2 Version 7, for distributed applications, you must bind a package at the
requester if your application contains statements that are processed at the
requester. For local applications, you must bind a plan, or a package and a plan,
even if your application contains only statements that are processed at the
requester.

Example 1: For a program that runs at location RQSTR and contains only these
SQL statements, in previous releases you only needed to bind a package at
location SRVR.

```
# EXEC SQL CONNECT TO SRVR;  
# EXEC SQL UPDATE TABLE_A SET COL1=:HV1;
```

Starting with DB2 Version 7, you need to bind a package at location RQSTR as well
as at location SRVR.

Example 2: For a local program that contains only the following SQL statement,
you did not need to bind a plan, or a package and a plan before DB2 Version 7.

```
# EXEC SQL DESCRIBE TABLE :HV_TBL INTO :SQLDA1;
```

Starting with DB2 Version 7, you need to bind a plan, or a package and a plan for
this program.

For the distributed case, after you bind the package at the requester, you need to
include that package in the PKLIST parameter when you bind the plan. If the
package that you execute at the requester does not include a SET CURRENT
PACKAGESET statement before other statements that are processed at the
requester, you need to explicitly include the collection ID for the requester's
package in the PKLIST parameter, rather than specifying an asterisk (*) for the
collection ID.

Failure to bind a package at the requester for the distributed case, or a plan or
package and plan for the local case, can result in an SQLCODE -805 or SQLCODE
-812 at run time.

To correct existing programs, perform the following actions:

- # • For the distributed case, to add a package for the statements that are executed
at the requester, bind a package at the requester, and then rebind the plan for
the program, with a PKLIST parameter that includes the new package. Ensure
that you grant the appropriate authorizations to execute the new package.
- # • For the local case, bind a plan, or a package and a plan, for the program that
contains only statements that execute at the requester. Ensure that you grant the
appropriate authorizations to execute the new plan.

To temporarily circumvent this restriction, you can set subsystem parameter
PKGLDTOL to YES. However, future releases of DB2 will not support this
subsystem parameter. For more on the PKGLDTOL subsystem parameter, see Part
2 of *DB2 Installation Guide*.

Modify applications that call DSNACCQC or DSNACCAV: You may need to
modify applications that call DB2-supplied stored procedures DSNACCQC or
DSNACCAV. The stored procedures are now defined to run in a WLM-established
stored procedures address space. The single result set that is returned by
DSNACCQC and the second result set that is returned by DSNACCAV contain one
more column than in prior releases of DB2.

Changed encoding scheme inheritance: Any CREATE TABLE, CREATE GLOBAL TEMPORARY TABLE, and DECLARE GLOBAL TEMPORARY TABLE statements with a LIKE clause, but no CCSID and IN clauses, will inherit the encoding scheme of the table being copied. Previously, these tables would have inherited the system default encoding scheme.

SQLCA changes: To provide additional information about the enhanced capability
of cursors in this version, several flags in the SQLCA are changed. New values are
added to the following flags to reflect the scrollability, sensitivity, and capability of
the cursor after an OPEN CURSOR or ALLOCATE CURSOR statement:

SQLWARN1
Contains an N for a non-scrollable cursor and a Y for a scrollable cursor.

SQLWARN4
Contains an I for an insensitive cursor and an S for a sensitive, static
cursor.

SQLWARN5
Contains a 1 for read-only cursors, 2 for cursors that can read and delete,
and a 4 for cursors that can read, delete, and update.

In addition, if no warnings exist, SQLWARN0 will remain blank even when
SQLWARN1, SQLWARN4, and SQLWARN5 contain any of these new values. Also,
Version 7 sets SQLWARN1 and SQLWARN5 for non-scrollable cursors. If you have
applications that check these flags, you may need to change these programs to
only check SQLWARN0 or set subsystem parameter DISABSCL to YES.

Changed format for DSN messages: The DSN message identifiers can now be eight or nine characters long. The message identifier formats are DSNcnnnl and DSNcnnnnl. To accommodate the longer message identifier, the message text begins in column 13 for all DSN messages. If you have applications that scan the text of DSN messages, you might need to modify them.

Changed format for message DSNU050I: Certain formatting problems in message DSNU050I have been corrected. The message formats differently in Version 7 than it did in Version 5. If you have applications that scan the text of message DSNU050I, you might need to modify them.

SQL reserved words: There are several new SQL reserved words in Version 7. Refer to *DB2 SQL Reference* for the list and adjust your applications accordingly.

Using the Euro symbol: New CCSIDs that support the Euro symbol have been added to DB2.

Using aliases: Prior to Version 7, aliases were known locally. Now, if remote aliases are used in SQL, they should be defined at the server site. SET statements that assign a host variable with the value of a special register will only assign the local value to the host variable.

QUIESCE return code: The QUIESCE utility command produces a return code of 4 for duplicate names in the list of table spaces that are to be quiesced. This is a change from a return code of 8. The QUIESCE processing continues, and the table space is quiesced only once.

DSNH message ID lengths: DSNH messages can be 8, 9, or 10 characters long. Existing messages in Version 7 will not be padded with zeroes to 10 characters, as they were in Version 5.

Positive SQLCODE from PREPARE: A return code greater than zero can be returned from a PREPARE statement if your migrated application is using predictive governing or optimization hints.

New DBPROTOCOL option: Applications might be affected by the new DBPROTOCOL option, which is both a subsystem parameter and a bind parameter. If the DBPROTOCOL option is not coded on the BIND, the value listed in the subsystem parameter DBPROTCL is used. For plans and packages to execute the same as they did in Version 5, they might need to be rebound, depending on the selected DBPROTOCOL value. In prior releases, remote access by name resulted in the use of DB2 private protocol to communicate with a remote DB2 for OS/390 and z/OS site. Remote access with the CONNECT statement used the DRDA database protocol. Users can now specify which protocol to use when communicating with the remote server.

Changed default for RELCURHL subsystem parameter: Use the RELCURHL subsystem parameter to indicate whether you want DB2 to release a data page or row lock at COMMIT for the row on which a cursor that is defined WITH HOLD is positioned. In Version 5, the default value was NO. Although this particular row or page lock is not necessary for maintaining cursor position, NO was provided as a default for existing applications that relied on that data lock. In Version 7, RELCURHL is a field on installation panel DSNTIP4. The field name is RELEASE LOCKS, and the default is changed to YES to provide for better concurrency. If you still require the lock, specify NO for RELEASE LOCKS.

Changed default for DYNRULS subsystem parameter: In earlier releases, the default for the DYNRULS subsystem parameter is NO. This means that the values of the precompiler options are used for dynamic SQL statements in plans or packages that are bound with DYNAMICRULES(BIND). For Version 7, the default is YES. YES means the application programming (DSNHDECP) defaults are used for dynamic SQL statements in plans or packages that are bound using DYNAMICRULES bind, define, or invoke behavior.

Using new column called CLUSTERRATIOF: The SYSINDEXES and SYSINDEXSTATS tables now have a new column, CLUSTERRATIOF. After migration, this column contains a default value until it is updated by RUNSTATS or some other application. If DB2 sees the default value in CLUSTERRATIOF, it uses the value in CLUSTERRATIO.

Support for large objects: The following incompatibilities result from the expansion of the SQLDA to support large objects:

- Because of the additional data types that are supported after Version 5, the host language statements that are generated by the precompiler for each SQL statement include a larger parameter list. As a result, the size of the object code that results from compiling the output of the precompiler increases. This increase varies according to the number of host variables that are specified in an SQL statement.
- INCLUDE SQLDA generates additional fields for applications that are written in assembler, PL/I, C, and C++. (See Appendix C of *DB2 SQL Reference* for descriptions of these fields.) If an SQL application program that is written in one

of those languages defines a name that matches any symbol in the additional fields, and the program also uses INCLUDE SQLDA, that program might not operate, assemble, or compile correctly.

A technique for reducing the number of matching columns no longer works: Part 5 (Volume 2) of *DB2 Administration Guide* formerly described a technique for reducing the number of matching columns for an index by changing equal predicates to BETWEEN predicates. Due to enhancements to BETWEEN predicate processing, this technique no longer reduces the number of matching columns. If you have queries that use this technique, consider using the alternative technique of discouraging the use of a particular index.

New reserved qualifier for tables, SESSION: DB2 uses a new implicit qualifier, SESSION, for a declared temporary table. All tables with a high-level qualifier of SESSION are treated as a declared temporary table. You must modify any existing tables with a high level qualifier of SESSION to use another qualifier.

Use the following query to identify existing tables, views, aliases, and synonyms that are inaccessible because of the SESSION qualifier:

```
SELECT 'TABLE', CREATOR, NAME
      FROM SYSIBM.SYSTABLES
      WHERE CREATOR = 'SESSION' AND TYPE IN ('T', 'G', 'X')
UNION
      SELECT 'VIEW', CREATOR, NAME
      FROM SYSIBM.SYSTABLES
      WHERE CREATOR = 'SESSION' AND TYPE = 'V'
UNION
      SELECT 'ALIAS', CREATOR, NAME
      FROM SYSIBM.SYSTABLES
      WHERE CREATOR = 'SESSION' AND TYPE = 'A'
UNION
      SELECT 'SYNONYM', CREATOR, NAME
      FROM SYSIBM.SYSSYNONYMS
      WHERE CREATOR = 'SESSION';
```

DB2 cannot know during the bind of the plan or package whether the static SQL statement that references a table name that is qualified by SESSION is for a persistent base table or for a declared temporary table. These static SQL statements are incrementally bound at run time when the static SQL statement is issued.

Release incompatibilities (migration from Version 6)

DCE security authentication no longer supported

DCE support has been removed. Kerberos authentication has replaced DCE. For more information about creating Resource Access Control Facility (RACF) profiles, see Part 3 (Volume 1) of *DB2 Administration Guide*.

Fewer sort operations for queries

A performance enhancement to DB2 results in fewer sort operations for queries that have ORDER BY clauses and have WHERE clauses with predicates of the form COL=constant. This enhancement can result in the following changes from prior releases of DB2:

- Changes in access paths, which can result in changes to EXPLAIN output or the elapsed time of a query.
- Some queries that previously received SQL code -136 (sort key length exceeds maximum) may now execute successfully.

Sysplex query parallelism in the PLAN_TABLE

If a parallel group being rebound to Version 7 is capable of Sysplex query parallelism, and only one DB2 data sharing member is active, a single-CEC parallel plan is developed, and an **X** is placed in column PARALLEL_MODE of table PLAN_TABLE. If you do not want the **X**, bind the plan or package on a member that is installed with COORDINATOR=NO. The **X** for column PARALLEL_MODE means that all available assisting DB2s are used at run time. You need to rebind all static plans that have a PARALLEL_MODE of X after migrating to take advantage of this change.

DISPLAY BUFFERPOOL changes

The DISPLAY BUFFERPOOL command syntax and output have changed. The new command provides more data sharing information required for understanding the level of inter-system sharing taking place. The DISPLAY BUFFERPOOL output no longer issues messages DSNB450I, DSNB451I, and DSNB452I. Several new messages with expanded information are generated by this command. For details on the changes see the DISPLAY BUFFERPOOL command in *DB2 Command Reference*.

Index changes

Indexes that have been altered, rebuilt, reorged, or newly created can increase in size by one page per nonpartitioned index or one page per index partition. The leaf distribution column (LEAFDIST) value in SYSINDEXPART may increase for indexes that have been altered, rebuilt, reorged, or newly created.

Work space formulas changed for utilities

Due to the larger table spaces and indexes available, the record header for several utility work data sets is lengthened by several bytes. The changed record headers which are used in several formulas for estimating work data set size affects the following utilities:

- CHECK DATA
- CHECK INDEX
- LOAD
- REBUILD INDEX
- REORG

The new calculations are explained in *DB2 Utility Guide and Reference* under the topic for defining work data sets for each utility.

Default for FASTSWITCH parameter is YES

For some utilities, the FASTSWITCH parameter has a default of YES, which results in the creation of J0001 data sets. Because some tools and other utilities require data set names to match, you may experience incompatibilities. The fifth-level qualifier can vary from I0001 to J0001. For more information about renaming data sets, see Part 2 (Volume 1) of DB2 Administration Guide.

Adjust application programs

You might need to adjust your application programs because of the following release incompatibilities. This list applies only if you are migrating from Version 6.

SQLCODE -538: After migrating to DB2 Version 7, you might receive SQLCODE -538 on an SQL CREATE TABLE FOREIGN KEY or SQL ALTER TABLE ADD FOREIGN KEY statement that previously executed successfully. If the foreign key references a parent key that is not defined as a primary key or unique key in the parent table, the creation of a foreign key will fail. Ensure that the parent key has been created as a primary key or unique key in the parent table.

#

SQLCODE -669: After migrating to DB2 Version 7, you might receive SQLCODE -669 on an SQL DROP INDEX statement that previously executed successfully. If the dropped index is a unique index that enforces a unique (primary or unique key) constraint or referential constraint, drop the constraint before you drop the index.

You can drop a unique index (for a unique key only) without first dropping the unique key constraint if the unique key was created in a release of DB2 prior to Version 7, and the unique key has no associated referential constraints.

Adjust user-defined function calls for new IBM functions: Several new functions have been added, such as DAYOFWEEK_ISO, WEEK_ISO, and IDENTITY_VAL_LOCAL. If you have user-defined functions, invoke them with a fully qualified name to avoid calling IBM functions that may have the same name. If the user-defined functions are not invoked with a fully qualified name and SYSIBM is first in the SQL path, the IBM function is selected instead of the user-defined function.

Changed default for CREATE FUNCTION statement: CREATE FUNCTION statements that executed in a prior release for table functions had FINAL CALL as the default. When these statements are executed in Version 7, NO FINAL CALL is the default.

Changed encoding scheme inheritance: Any CREATE TABLE, CREATE GLOBAL TEMPORARY TABLE, and DECLARE GLOBAL TEMPORARY TABLE statements with a LIKE clause, but no CCSID and IN clauses, inherit the encoding scheme of the table being copied. Prior to Version 7, these tables would have inherited the system default encoding scheme.

Bind plans and packages: Before DB2 Version 7, you did not need to bind a plan
or a package in the following situations:

• For distributed applications, you did not need to bind a package at the requester
for statements that were processed at the requester.

• For local applications, you did not need to bind a plan, or a package and a plan,
if a program contained ONLY statements that are processed at the requester.

Appendix B of DB2 SQL Reference indicates which SQL statements are processed
at the requester. Examples of those statements are CONNECT, COMMIT,
ROLLBACK, and RELEASE.

As of DB2 Version 7, for distributed applications, you must bind a package at the
requester if your application contains statements that are processed at the
requester. For local applications, you must bind a plan, or a package and a plan,
even if your application contains only statements that are processed at the
requester.

Example 1: For a program that runs at location RQSTR and contains only these
SQL statements, in previous releases you only needed to bind a package at
location SRVR.

```
# EXEC SQL CONNECT TO SRVR;  
# EXEC SQL UPDATE TABLE_A SET COL1=:HV1;
```

Starting with DB2 Version 7, you need to bind a package at location RQSTR as well
as at location SRVR.

Example 2: For a local program that contains only the following SQL statement,
you did not need to bind a plan, or a package and a plan before DB2 Version 7.

```

# EXEC SQL DESCRIBE TABLE :HV_TBL INTO :SQLDA1;
#
# Starting with DB2 Version 7, you need to bind a plan, or a package and a plan for
# this program.
#
# For the distributed case, after you bind the package at the requester, you need to
# include that package in the PKLIST parameter when you bind the plan. If the
# package that you execute at the requester does not include a SET CURRENT
# PACKAGESET statement before other statements that are processed at the
# requester, you need to explicitly include the collection ID for the requester's
# package in the PKLIST parameter, rather than specifying an asterisk (*) for the
# collection ID.
#
# Failure to bind a package at the requester for the distributed case, or a plan or
# package and plan for the local case, can result in an SQLCODE -805 or SQLCODE
# -812 at run time.
#
# To correct existing programs, perform the following actions:
#
# • For the distributed case, to add a package for the statements that are executed
# at the requester, bind a package at the requester, and then rebind the plan for
# the program, with a PKLIST parameter that includes the new package. Ensure
# that you grant the appropriate authorizations to execute the new package.
#
# • For the local case, bind a plan, or a package and a plan, for the program that
# contains only statements that execute at the requester. Ensure that you grant the
# appropriate authorizations to execute the new plan.
#
# To temporarily circumvent this restriction, you can set subsystem parameter
# PKGLDTOL to YES. However, future releases of DB2 will not support this
# subsystem parameter. For more on the PKGLDTOL subsystem parameter, see Part
# 2 of DB2 Installation Guide.
#
# Modify applications that call DSNACCQC or DSNACCAV: You might need to
# modify applications that call DB2-supplied stored procedures DSNACCQC or
# DSNACCAV. The stored procedures are now defined to run in a WLM-established
# stored procedures address space. The single result set that is returned by
# DSNACCQC and the second result set that is returned by DSNACCAV contain one
# more column than in prior releases of DB2.
#
# SQL reserved words: Version 7 has several new SQL reserved words. Refer to
# DB2 SQL Reference for the list and adjust your applications accordingly.
#
# Changed return code for message DSNU185I: The return code for DSNU185I
# has changed from return code 8 to return code 0. If you have applications that scan
# the return code of this message, you might need to modify them. In Version 6, utility
# processing stopped upon encountering an object created with the DEFINE NO
# attribute. These undefined objects were identified with message DSNU185I and
# return code 8. In Version 7, undefined objects are skipped and utility processing
# continues. Message DSNU185I is issued, but with return code 0.
#
# Changed default for identity columns: In Version 7, the default for the START
# WITH value of an identity column is the specified value for MINVALUE (for an
# ascending sequence) or MAXVALUE (for a descending sequence). In prior versions,
# no MINVALUE or MAXVALUE options existed, so if a value was not explicitly
# specified for START WITH when an identity column was specified, the default was
# 1.

```

New reserved qualifier for tables, *SESSION*: DB2 uses a new implicit qualifier, *SESSION*, for a declared temporary table. All tables with a high-level qualifier of *SESSION* are treated as a declared temporary table. You must modify any existing tables with a high-level qualifier of *SESSION* to use another qualifier.

Use the following query to identify existing tables, views, aliases, and synonyms that are inaccessible because of the *SESSION* qualifier:

```
SELECT 'TABLE', CREATOR, NAME
      FROM SYSIBM.SYSTABLES
      WHERE CREATOR = 'SESSION' AND TYPE IN ('T', 'G', 'X')
UNION
SELECT 'VIEW', CREATOR, NAME
      FROM SYSIBM.SYSTABLES
      WHERE CREATOR = 'SESSION' AND TYPE = 'V'
UNION
SELECT 'ALIAS', CREATOR, NAME
      FROM SYSIBM.SYSTABLES
      WHERE CREATOR = 'SESSION' AND TYPE = 'A'
UNION
SELECT 'SYNONYM', CREATOR, NAME
      FROM SYSIBM.SYSSYNONYMS
      WHERE CREATOR = 'SESSION';
```

DB2 cannot know during the bind of the plan or package whether the static SQL statement that references a table name that is qualified by *SESSION* is for a persistent base table or for a declared temporary table. These static SQL statements are incrementally bound at run time when the static SQL statement is issued.

Release coexistence

This section describes changes that might affect your DB2 operations after migrating to Version 7 of DB2 for OS/390 and z/OS.

IRLM

As you apply IRLM service to members of a data sharing group, some members run with the newer service level and some with the older service level. Even though each member uses the same release level of IRLM, the different service levels can raise issues you must consider. For more information about IRLM coexistence, see *DB2 Data Sharing: Planning and Administration*.

Data sharing

DB2 can support Version 6 and Version 7 or Version 5 and Version 7 members in a data sharing group, but not all three versions at once. DB2 supports the coexistence of only two releases at a time. To support two releases, you must first apply the fallback SPE to all members of the group from the previous version as described in “Migrating a data sharing group” on page 79. Release coexistence begins when you migrate the first data sharing member to Version 7. You must successfully migrate the first data sharing member to Version 7 before attempting to migrate the other data sharing members.

For the best availability, you can migrate the members to the new release one member at a time. When developing your migration plan, remember that new functions introduced in this release are not available to members of the group that have not yet migrated.

Recommendation: Use one of the following approaches:

- Migrate all members to the new release before you use new function.
- Restrict the execution of packages and plans bound on Version 7 to members of the group that have already migrated.

This restriction serves two purposes. First, if new plans or packages use new functions, you avoid the application errors that can occur if the plan or package tries to execute an SQL statement that is not allowed in the previous version. Second, you avoid the automatic rebind that occurs when any plan or package that is bound on Version 7 is run on the previous version. It also avoids the automatic rebind that occurs when a Version 7-bound plan or package that was automatically rebound on the previous version is subsequently run on Version 7. If you cannot control which member a plan or package runs on, consider how to handle binds and automatic rebinds while two releases are coexisting. For more information, see the description of the AUTOBIND option of installation panel DSNTIPO.

For detailed information about data sharing release coexistence considerations, see *DB2 Data Sharing: Planning and Administration*.

TSO and CAF logon procedures: You can attach to either release of DB2 with your existing TSO or CAF logon procedures, without changing the load libraries for your applications. After you migrate completely to the latest level of DB2, you **must** update those procedures and jobs to point to the latest level of DB2 load libraries. If you forget to update the procedures and jobs before migrating to any release subsequent to Version 7, they can no longer work in that release.

For a detailed list of considerations for a data sharing group with multiple DB2 releases, see Chapter 4 of *DB2 Data Sharing: Planning and Administration*.

Distributed environment

DB2 for OS/390 and z/OS communicates in a distributed data environment with DB2 Version 2 Release 3 and later, using either DB2 private protocol access or DRDA access. However, the distributed functions that are introduced in this release of DB2 for OS/390 and z/OS can be used only when using DRDA access.

Other DRDA partners at DRDA level 4 can also take advantage of the functions that are introduced in this release of DB2 for OS/390 and z/OS.

Installation changes

This section shows the panels that are used by the installation CLIST to customize the jobs you use to install or migrate to Version 7. This section also lists the changes to SMP/E jobs and sample jobs.

You can also install DB2 for OS/390 and z/OS from a Windows NT or OS/2 workstation with the DB2 Installer feature.

Version 7 panels

Table 8 on page 95 shows the panels for DB2 for OS/390 and z/OS Version 7 installation and migration. With the addition of the new functions in Version 7, several panels have been modified and new fields have been added. The new and modified panels have a Yes listed under the **Panel modified** column in Table 8 on page 95.

Table 8. Version 7 installation and migration panels

Panel ID	Panel title	Panel modified
DSNTIPA0	Online Book Data Set Names	
DSNTIPA1	Main Panel	Yes
DSNTIPA2	Data Parameters	
DSNTIPK	Define Group or Member ⁽¹⁾	
DSNTIPH	System Resource Data Set Names	
DSNTIPT	Data Set Names Panel 1	
DSNTIPU	Data Set Names Panel 2	
DSNTIPQ	Data Set Names Panel 3	
DSNTIPG	Data Set Names Panel 4	
DSNTIPW	Data Set Names Panel 5	
DSNTIPD	Sizes Panel 1	
DSNTIP7	Sizes Panel 2	
DSNTIPE	Thread Management	Yes
DSNTIP1	Buffer Pool Sizes Panel 1	
DSNTIP2	Buffer Pool Sizes Panel 2	
DSNTIP6	Buffer Pool Sizes Panel 3	
DSNTIPN	Tracing and Checkpoint Parameters	Yes
DSNTIPO	Operator Functions	Yes
DSNTIPF	Application Programming Defaults Panel 1	Yes
DSNTIP4	Application Programming Defaults Panel 2	Yes
DSNTIPI	IRLM Panel 1	
DSNTIPJ	IRLM Panel 2	
DSNTIPP	Protection	Yes
DSNTIPM	MVS PARMLIB Updates	Yes
DSNTIPL	Active Log Data Set Parameters	Yes
DSNTIPA	Archive Log Data Set Parameters	
DSNTIPS	Databases and Spaces to Start Automatically	
DSNTIPR	Distributed Data Facility Panel 1	
DSNTIP5	Distributed Data Facility Panel 2	
DSNTIPX	Routine Parameters	
DSNTIPZ	Data Definition Control Support	
DSNTIPY	Job Editing	
DSNTIPC	DB2 CLIST Calculations Panel 1	Yes
DSNTIPC1	DB2 CLIST Calculations Panel 2	
DSNTIPB	Update Selection Menu	

Notes:

1. DSNTIPK is for installing and migrating in data sharing mode.

Subsystem parameters added to installation panels

Two important subsystem parameters have been added to installation panels (see Table 9). As a result, the values you choose for these parameters are used during migration to a new release of DB2.

Table 9. Subsystem parameters that have been added to installation panels

Subsystem parameter	Panel	Field name
IMMEDWRI	DSNTIP4	IMMEDIATE WRITE
EDMBFIT	DSNTIP4	LARGE EDM BETTER FIT

Changes to sample jobs

With the addition of the new functions in Version 7, several existing sample jobs have been modified and several new jobs added. The new and changed sample jobs are listed in Table 10.

Table 10. New and modified sample jobs

Sample job	New or modified
DSNTEJ0	Modified
DSNTEJ1	Modified
DSNTEJ1L	Modified
DSNTEJ1P	Modified
DSNTEJ1U	New
DSNTEJ2A	Modified
DSNTEJ6V	New
DSNTEJ6W	New
DSNTEJ6Z	New
DSNTEJ80	New

Appendix A. Changes to commands

This appendix provides an overview of the new and changed commands in Version 7 of DB2 for OS/390 and z/OS. The purpose of the appendix is to highlight the major changes. For complete information on all the changes, such as the syntax for new or changed commands see *DB2 Command Reference*.

New commands

Table 11 shows the new commands in Version 7.

Table 11. New commands

Command name	Description
-SET SYSPARM	Load a new subsystem parameters load module while DB2 is running
-DISPLAY DDF	Display DDF status information

Changed commands

Table 12 shows that several existing commands have new and changed options.

Table 12. Changes to existing commands

Command	Description of enhancements and notes
CANCEL THREAD (DB2)	<p>New option: NOBACKOUT</p> <p>NOBACKOUT specifies that DB2 will not attempt to back out the data during transaction rollback processing. Cancelling the thread with NOBACKOUT leaves objects in an inconsistent state. Do not issue this command with NOBACKOUT unless you have a plan to resolve the data inconsistency.</p>
DISPLAY THREAD (DB2)	<p>New option: RRSURID(rrs-urid)</p> <p>Specifies that only threads that match the specified RRSURID selection criteria are to be displayed.</p>
MODIFY irlmproc,DIAG (MVS IRLM)	<p>New options: PLOCK ALL NONE</p> <p>PLOCK directs IRLM to generate a dump the first time it detects that P-lock negotiation is taking longer than two minutes. Dumps of the IRLM and DB2 address spaces are taken and placed in the SYS1.DUMPxx data set.</p> <p>ALL directs IRLM to generate diagnostic dumps for IRLM or DBMS subsystems in a data sharing group for the following unusual conditions:</p> <ul style="list-style-type: none">• P-lock negotiation takes longer than two minutes.• Child-lock propagation takes longer than 45 seconds. <p>NONE disables generation of all diagnostic dumps.</p>

Table 12. Changes to existing commands (continued)

Command	Description of enhancements and notes
RECOVER POSTPONED (DB2)	<p>New option: CANCEL</p> <p>CANCEL directs DB2 to stop processing of all postponed abort units of recovery immediately. Cancelling postponed abort units of recovery leaves objects in an inconsistent state. Do not issue this command with NOBACKOUT unless you have a plan to resolve the data inconsistency.</p>
SET LOG (DB2)	<p>New options: CHKTIME SUSPEND RESUME</p> <p>CHKTIME specifies the number of minutes between the start of successive checkpoints. This option overrides log records that are specified by installation options or the LOGLOAD option, which are based on checkpoint frequency.</p> <p>SUSPEND directs DB2 to suspend logging and update activity for the current DB2 subsystem until SET LOG RESUME is issued. DB2 externalizes unwritten log buffers, takes a system checkpoint (in non-data sharing environments), updates the BSDS with the high-written RBA, then suspends the update activity. Message DSNJ372I is issued and remains on the console until update activity resumes.</p> <p>RESUME specifies logging and update activity are to resume for the current DB2 subsystem and removes the message DSNJ372I from the console.</p>
START DB2 (DB2)	<p>New option: LIGHT</p> <p>LIGHT specifies whether a light restart is to be performed in a data sharing environment.</p>
START irlmproc (MVS IRLM)	<p>New option: PGPROT</p> <p>PGPROT specifies whether the IRLM load modules that are resident in common storage are placed in MVS page-protected storage.</p>

Appendix B. Changes to utilities

This appendix summarizes the changes to utilities in Version 7 of DB2 for OS/390 and z/OS:

“New utilities”

“Changed utilities” on page 100

“Other utility changes” on page 105

New utilities

Table 13 shows the new utilities. In addition to new utilities, Version 7 adds some new utility control statements. The new utility control statements are:

EXEC SQL

The EXEC SQL utility control statement declares cursors or executes dynamic SQL statements.

LISTDEF

The LISTDEF utility control statement defines a list of database objects (table spaces and index spaces) to be processed by utilities. Lists contain explicitly name objects, use “pattern matching” characters to define object, or both.

TEMPLATE

The TEMPLATE utility control statement defines a template for dynamically allocating data sets. The template eliminates the need to create JCL DD cards for data sets and when used in combination with the LISTDEF control statement, greatly reduces the maintenance of utility job streams.

OPTIONS

The OPTIONS utility control statement alters the performance characteristics of utility jobs. You can:

- Alter the way errors are handled
- Alter the return code for warning messages
- Modify the DD names of LISTDEF and TEMPLATE libraries
- Activate the utility PREVIEW process

Table 13. Overview of new utilities

Utility name	Description
COPYTOCOPY	<p>COPYTOCOPY improves availability by decreasing the time an object is unavailable when making image copy data sets for recovery. Use COPY, LOAD, or REORG to make a primary image copy and then use COPYTOCOPY at a more convenient time to make additional copies. COPYTOCOPY makes up to three additional backup copies from either the local site primary or recovery site primary image copy and registers these copies in the DB2 catalog.</p> <p>With COPYTOCOPY, you can copy a list of image copy data sets using pattern-matching characters and dynamically allocate your data sets using the TEMPLATE control statement.</p>
MODIFY STATISTICS	<p>MODIFY STATISTICS improves performance by increasing usable catalog space and facilitating processes that access the catalog history tables. The MODIFY STATISTICS online utility deletes user-specified statistical records from catalog history tables.</p> <p>You can delete all or portions of statistics rows for a table space, index space, or index. You can delete rows by specified criteria such as date or age.</p>

Table 13. Overview of new utilities (continued)

Utility name	Description
UNLOAD	<p>UNLOAD improves the unload process with greater speed and versatility. The UNLOAD online utility unloads data from one or more source objects to one or more BSAM sequential data sets in an external format. The source can be either DB2 table spaces or DB2 image copy data sets.</p> <p>You can unload rows from an entire table space, or select specific partitions or tables to unload. You can also select columns using the field specification list. For a partitioned table space, you can specify that all selected partitions are loaded into a single data set, or you can unload each partition in parallel into physically distinct data sets.</p> <p>UNLOAD provides the basic functions of REORG UNLOAD EXTERNAL with the following enhancements:</p> <ul style="list-style-type: none"> • Unloads data from an image copy data set • Unloads multiple partitions in parallel • Provides row samples by table • Provides field selection, ordering, and formatting options • Supports SHRLEVEL CHANGE or REFERENCE

Changed utilities

Table 14 lists the new and changed options for many existing DB2 for OS/390 and z/OS utilities.

Table 14. New and changed utility options

Utility name	Description of enhancements and notes
CHECK DATA	<p>Changed options: ERRDDN WORKDDN</p> <p>The ERRDDN and WORKDDN options now support the template construct that is used for dynamic data set allocation. The templates contain:</p> <ul style="list-style-type: none"> • The data set naming convention • DFSMS parameters • Disk or tape allocation parameters <p>Templates are not used for sort work data sets.</p>
CHECK INDEX	<p>New option: LIST</p> <p>Changed options: ERRDDN WORKDDN</p> <p>The new LIST option uses a list of specific objects, a generic pattern-matching expression, or both. The list is defined using a LISTDEF control statement.</p> <p>The ERRDDN and WORKDDN options now support the template construct that is used for dynamic data set allocation. The templates contain:</p> <ul style="list-style-type: none"> • The data set naming convention • DFSMS parameters • Disk or tape allocation parameters <p>Templates are not used for sort work data sets.</p>

Table 14. New and changed utility options (continued)

Utility name	Description of enhancements and notes
CHECK LOB	<p>Changed option: WORKDDN</p> <p>The WORKDDN option now supports the template construct that is used for dynamic data set allocation. The templates contain:</p> <ul style="list-style-type: none"> • The data set naming convention • DFSMS parameters • Disk or tape allocation parameters <p>Templates are not used for sort work data sets.</p>
COPY	<p>New option: CHECKPAGE LIST TAPEUNITS</p> <p>Changed options: COPYDDN FILTERDDN RECOVERYDDN</p> <p># The CHECKPAGE option if specified, checks each page in the table space or index space for validity. The new LIST option uses a list of specific objects, a generic pattern-matching expression, or both. The list is defined using a LISTDEF control statement.</p> <p># The TAPEUNITS option supports parallel processing of image copies on tape drives. This option specifies the number of tape drives that are dynamically allocated for the list of objects that are to be processed in parallel. TAPEUNITS must be specified with the PARALLEL option.</p> <p># The COPYDDN , FILTERDDN, and RECOVERYDDN options now support the template construct that is used for dynamic data set allocation. The templates contain:</p> <ul style="list-style-type: none"> • The data set naming convention • DFSMS parameters • Disk or tape allocation parameters <p>Templates are not used for sort work data sets.</p>

Table 14. New and changed utility options (continued)

Utility name	Description of enhancements and notes
LOAD	<p>New options:</p>
#	<p>FORCEROLLUP HISTORY INCURSOR SHRLEVEL UNICODE</p>
#	<p>Changed options: COPYDDN DISCARD DDN ERRORDDN KEEPDICTIONARY MAPDDN RECOVERYDDN WORKDDN INTO TABLE</p>
#	<p>The new FORCEROLLUP keyword determines if aggregation or rollup of statistics takes place when RUNSTATS is executed in spite of some parts being empty. The new HISTORY option records catalog table inserts and updates that DB2 makes to the catalog history tables. The INCURSOR option specifies a cursor for the input data set. The SHRLEVEL option specifies the extent to which applications can concurrently access the table space or partition during LOAD utility processing. The UNICODE option specifies that the input data set is Unicode format.</p>
#	<p>The COPYDDN, DISCARD DDN, ERRORDDN, MAPDDN, RECOVERYDDN, and WORKDDN options now support the template construct that is used for dynamic data set allocation. The templates contain:</p> <ul style="list-style-type: none"> • The data set naming convention • DFSMS parameters • Disk or tape allocation parameters
#	<p>Templates are not used for sort work data sets. The KEEPDICTIONARY option can be specified with the LOAD RESUME YES and LOAD RESUME NO requests.</p>
#	<p>The INTO TABLE option adds the INDDN and DISCARD DDN keywords for partitioned table spaces. These two keywords with partition parallelism loads each partition from a separate input data set, and writes discarded records to a separate data set.</p>
MERGECOPY	<p>New option:</p>
	<p>LIST</p>
	<p>Changed options: COPYDDN RECOVERYDDN</p>
	<p>The new LIST option uses a list of specific objects, a generic pattern-matching expression, or both. The list is defined using a LISTDEF control statement.</p>
	<p>The COPYDDN and RECOVERYDDN options now support the template construct that is used for dynamic data set allocation. The templates contain:</p> <ul style="list-style-type: none"> • The data set naming convention • DFSMS parameters • Disk or tape allocation parameters
	<p>Templates are not used for sort work data sets.</p>

Table 14. New and changed utility options (continued)

Utility name	Description of enhancements and notes
MODIFY RECOVERY	<p data-bbox="542 268 683 327">New option: LIST</p> <p data-bbox="542 352 1386 411">The new LIST option uses a list of specific objects, a generic pattern-matching expression, or both. The list is defined using a LISTDEF control statement.</p>
QUIESCE	<p data-bbox="542 441 683 499">New option: LIST</p> <p data-bbox="542 525 1386 583">The new LIST option uses a list of specific objects, a generic pattern-matching expression, or both. The list is defined using a LISTDEF control statement.</p>
REBUILD INDEX	<p data-bbox="149 636 818 722"># # # #</p> <p data-bbox="542 615 818 722">New options: FORCEROLLUP HISTORY LIST</p> <p data-bbox="542 743 764 802">Changed option: WORKDDN</p> <p data-bbox="542 823 1435 995">The new FORCEROLLUP keyword determines if aggregation or rollup of statistics takes place when RUNSTATS is executed in spite of some parts being empty. The new HISTORY option records catalog table inserts and updates that DB2 makes to the catalog history tables. The new LIST option uses a list of specific objects, a generic pattern-matching expression, or both. The list is defined using a LISTDEF control statement.</p> <p data-bbox="542 1020 1456 1079">The WORKDDN option now supports the template construct that is used for dynamic data set allocation. The templates contain:</p> <ul data-bbox="542 1083 938 1167" style="list-style-type: none"> • The data set naming convention • DFSMS parameters • Disk or tape allocation parameters <p data-bbox="542 1184 1049 1209">Templates are not used for sort work data sets.</p>
RECOVER	<p data-bbox="149 1266 857 1415"># # # # # # # # #</p> <p data-bbox="542 1245 857 1415">New options: LIST TOLASTCOPY TOLASTFULLCOPY TAPEUNITS LOGRANGES</p> <p data-bbox="542 1440 1435 1558">The new LIST option uses a list of specific objects, a generic pattern-matching expression, or both. The list is defined using a LISTDEF control statement. With the TOLASTCOPY and TOLASTFULLCOPY options, the image copy data set name no longer needs to be coded in the RECOVER utility statement.</p> <p data-bbox="542 1583 1456 1696">The TAPEUNITS option supports parallel processing of image copies on tape drives. This option specifies the number of tape drives that are dynamically allocated for the list of objects that are to be processed in parallel. TAPEUNITS must be specified with the PARALLEL option.</p> <p data-bbox="542 1722 1435 1803">If you specify LOGRANGES NO, RECOVER does not use SYSLGRNX information for the LOGAPPLY phase. Use this option only under the direction of IBM Software Support.</p>

Table 14. New and changed utility options (continued)

Utility name	Description of enhancements and notes
REORG INDEX	<p>New options:</p> <p># DRAIN_WAIT</p> <p># FASTSWITCH</p> <p># FORCEROLLUP</p> <p># HISTORY</p> <p># LIST</p> <p># RETRY</p> <p># RETRY_DELAY</p> <p>Changed option:</p> <p>WORKDDN</p> <p># Use the DRAIN_WAIT option to specify how long to wait for drains. Use the RETRY and RETRY_WAIT keywords to customize the drain process. The new FASTSWITCH option keeps the data set name unchanged and updates the catalog to reference the newly reorganized data set. The default value is YES. The new FORCEROLLUP keyword determines if aggregation or rollup of statistics takes place when RUNSTATS is executed in spite of some parts being empty. The new HISTORY option records catalog table inserts and updates that DB2 makes to the catalog history tables. The new LIST option uses a list of specific objects, a generic pattern-matching expression, or both. The list is defined using a LISTDEF control statement.</p> <p>The WORKDDN option now supports the template construct that is used for dynamic data set allocation. The templates contain:</p> <ul style="list-style-type: none"> • The data set naming convention • DFSMS parameters • Disk or tape allocation parameters <p>Templates are not used for sort work data sets.</p>

Table 14. New and changed utility options (continued)

Utility name	Description of enhancements and notes
REORG TABLESPACE	<p>New options:</p>
#	DRAIN_WAITFASTSWITCH
#	FORCEROLLUP
#	HISTORY
#	LIST
#	RETRY
#	RETRY_DELAY
	<p>Changed options:</p>
	COPYDDN
	DISCARDN
	PUNCHDDN
	RECOVERYDDN
	UNLDDN
	WORKDDN
#	<p>Use the DRAIN_WAIT option to specify how long to wait for drains. Use the RETRY and RETRY_WAIT keywords to customize the drain process. The new FASTSWITCH option keeps the data set name unchanged and updates the catalog to reference the newly reorganized data set. The default value is YES. The new FORCEROLLUP keyword determines if aggregation or rollup of statistics takes place when RUNSTATS is executed in spite of some parts being empty. The new HISTORY option records catalog table inserts and updates that DB2 makes to the catalog history tables. The new LIST option uses a list of specific objects, a generic pattern-matching expression, or both. The list is defined using a LISTDEF control statement.</p>
#	<p>The COPYDDN, DISCARDN, PUNCHDDN, RECOVERYDDN, UNLDDN, and WORKDDN options now support the template construct that is used for dynamic data set allocation. The templates contain:</p> <ul style="list-style-type: none"> • The data set naming convention • DFSMS parameters • Disk or tape allocation parameters <p>Templates are not used for sort work data sets.</p>
REPORT	<p>New option:</p> <p>LIST</p>
	<p>The new LIST option uses a list of specific objects, a generic pattern-matching expression, or both. The list is defined using a LISTDEF control statement.</p>
RUNSTATS	<p>New options:</p>
#	FORCEROLLUP
#	HISTORY
#	LIST
#	<p>The new FORCEROLLUP keyword determines if aggregation or rollup of statistics takes place when RUNSTATS is executed in spite of some parts being empty. The new HISTORY option records catalog table inserts and updates that DB2 makes to the catalog history tables. The new LIST option uses a list of specific objects, a generic pattern-matching expression, or both. The list is defined using a LISTDEF control statement.</p>
#	

Other utility changes

Other changes to utilities in Version 7 are:

- A new method for switching access between the original object and the shadow object reduces the time for online REORG when using SHRLEVEL CHANGE or SHRLEVEL REFERENCE. The new FASTSWITCH method toggles the fifth qualifier, referred to as the instance qualifier, of the data set names between I0001 and J0001. The default value of YES causes the instance qualifier to toggle with each REORG SHRLEVEL CHANGE or SHRLEVEL REFERENCE. The FASTSWITCH method can be used on table spaces or index spaces that are either DB2 managed or user managed.
- You can improve data availability during BUILD2 phase of online REORG for a partition or range of partitions. Multiple subtasks process the logical partitions asynchronously.
- The availability of some user objects can be improved:
 - Objects marked with refresh-pending status (REFP) can be recovered to a point in time using the RECOVER or REBUILD INDEX utility.
 - You can load user-defined table spaces that are marked with the refresh-pending status by using the REPLACE option of the LOAD utility.
 - You can specify LOAD RESUME YES SHRLEVEL CHANGE to give users read and write access to data while the LOAD utility runs.
- You can load data into a partitioned table space by using multiple input data sets in the same job step. Table space partitions can be loaded in parallel within a single job step when you define a separate input data set for each partition.
- All catalog and directory indexes must be converted to Type 2 when on Version 5 prior to migrating to Version 7. The CATMAINT UPDATE utility abnormally terminates if a Type 1 directory or catalog index is found.
- You can restart utility jobs without specifying the RESTART keyword. If the UTLRSTRT system parameter is set to ON and you resubmit a job that finished abnormally, DB2 recognizes the job and restarts it if possible. If the UTLRSTRT system parameter is set to OFF, you must specify the RESTART parameter for the job to be restarted.

#

Appendix C. Changes to SQL

This appendix provides an overview of the new and changed SQL statements in Version 7 of DB2 for OS/390 and z/OS:

“New SQL statements”

“Changed SQL statements”

“New functions” on page 117

“Changed functions” on page 119

“Other SQL language changes” on page 119

For complete information on all changes, such as the syntax for new or changed SQL statements, comprehensive descriptions of keywords, and examples of usage, see *DB2 SQL Reference*.

New SQL statements

Table 15 shows the new SQL statements in Version 7.

Table 15. New SQL statements

SQL statement	Description
ALTER FUNCTION (SQL scalar function)	Changes the description of an SQL scalar function at the current server
CREATE FUNCTION (SQL scalar function)	Defines an SQL scalar function
DECLARE VARIABLE	Defines a CCSID for a host variable and the subtype of the variable and causes DB2 to tag a host variable with a specific CCSID
SET CURRENT APPLICATION ENCODING SCHEME	Assigns a value to the CURRENT APPLICATION ENCODING SCHEME special register to allow control of which encoding scheme is used for dynamic SQL statements
SET <i>host-variable</i>	Assigns values, either expressions or NULL values, to host variables (previously part of SET Assignment statement)
SET <i>transition-variable</i>	Assigns values, either expressions or NULL values, to transition variables (previously part of SET Assignment statement)

Changed SQL statements

As shown in Table 16 on page 108, many existing SQL statements have new and changed clauses.

Table 16. Changes to existing SQL statements

SQL statement	Description of enhancements and notes
ALTER FUNCTION	<p>New clauses: INHERIT SPECIAL REGISTERS DEFAULT SPECIAL REGISTERS STATIC DISPATCH</p> <p>Changed clauses: LANGUAGE PARAMETER STYLE CCSID <i>encoding-scheme</i></p> <p>INHERIT SPECIAL REGISTERS indicates that special registers should be inherited according to rules for characteristics of special registers in a user-defined function. DEFAULT SPECIAL REGISTERS indicates that special registers should be initialized to the default values for such functions.</p> <p>At function resolution, DB2 chooses a function based on the static (or declared) types of function parameters.</p> <p>LANGUAGE is extended to allow for JAVA. LANGUAGE JAVA cannot be specified with a PARAMETER STYLE other than JAVA, or with DBINFO, SCRATCHPAD, FINAL CALL, PROGRAM TYPE MAIN, or RUN OPTIONS.</p> <p>PARAMETER STYLE is extended to allow for JAVA. When LANGUAGE JAVA is specified, PARAMETER STYLE JAVA must also be specified.</p> <p>The UNICODE option is added to the CCSID <i>encoding-scheme</i> clause. The UNICODE keyword specifies that the data must be encoded using the CCSIDs for Unicode data.</p>
# ALTER INDEX	<p>Changed clause: PIECESIZE PART and VALUES</p> <p>A warning emphasizes that when the PIECESIZE value is altered, the index is placed into REBUILD-pending (PSRBD) status and the entire index space becomes inaccessible until REBUILD INDEX or REORG TABLESPACE is run.</p> <p>If PART and VALUES are specified to change the limit key values of a partitioned index, the plans and packages that are dependent on that index are marked INVALID and go through automatic rebind the next time they are run.</p>

Table 16. Changes to existing SQL statements (continued)

SQL statement	Description of enhancements and notes
ALTER PROCEDURE (external procedure)	<p>New clauses: INHERIT SPECIAL REGISTERS DEFAULT SPECIAL REGISTERS CALLED ON NULL INPUT</p> <p>Changed clauses: LANGUAGE PARAMETER STYLE CCSID <i>encoding-scheme</i></p> <p>INHERIT SPECIAL REGISTERS indicates that special registers should be inherited according to rules for characteristics of special registers in a user-defined function. DEFAULT SPECIAL REGISTERS indicates that special registers should be initialized to the default values for such functions.</p> <p>CALLED ON NULL INPUT can now be explicitly specified if you want to specify all the options that will be in effect.</p> <p>LANGUAGE is extended to allow for JAVA. LANGUAGE JAVA cannot be specified without PARAMETER STYLE JAVA and the EXTERNAL NAME clause. It cannot be specified with DBINFO, NO WLM ENVIRONMENT, PROGRAM TYPE MAIN, or RUN OPTIONS.</p> <p>PARAMETER STYLE is extended to allow for JAVA. When LANGUAGE JAVA is specified, PARAMETER STYLE JAVA must also be specified.</p> <p>The UNICODE option is added to the CCSID <i>encoding-scheme</i> clause. The UNICODE keyword specifies that the data must be encoded using the CCSIDs for Unicode data.</p>
ALTER PROCEDURE (SQL procedure)	<p>New clauses: INHERIT SPECIAL REGISTERS DEFAULT SPECIAL REGISTERS CALLED ON NULL INPUT</p> <p>INHERIT SPECIAL REGISTERS indicates that special registers should be inherited according to rules for characteristics of special registers in a user-defined function. DEFAULT SPECIAL REGISTERS indicates that special registers should be initialized to the default values for such functions.</p> <p>CALLED ON NULL INPUT can now be explicitly specified if you want to specify all the options that will be in effect.</p>

Table 16. Changes to existing SQL statements (continued)

SQL statement	Description of enhancements and notes
ALTER TABLE	<p>New clause: <i>ADD unique-constraint</i></p> <p>Changed clause: AS IDENTITY DROP CONSTRAINT ADD PRIMARY KEY</p> <p><i>ADD unique-constraint</i> defines a primary key that is composed of identified columns or a unique key composed of the identified columns with a specified constraint name.</p> <p>The AS IDENTITY clause has new keywords: CYCLE/NO CYCLE (indication of whether values for the column continue to be generated after the minimum or maximum value is reached), MAXVALUE (maximum value allowed for the column), and MINVALUE (minimum value allowed for the column).</p> <p>The DROP CONSTRAINT clause, which used to drop only foreign key and check constraints, now also drops primary key and unique key constraints. The DROP CONSTRAINT clause must not be used on a ALTER TABLE statement that includes DROP FOREIGN KEY, DROP CHECK, DROP PRIMARY KEY, or DROP UNIQUE clauses.</p> <p>In the ADD PRIMARY KEY clause, you can now specify a <i>constraint-name</i>.</p>
# ALTER TABLESPACE # #	<p>Changed clauses: PRIQTY SECQTY</p> <p>The maximum values that can be specified for PRIQTY is 64GB if the running DB2 is under DFP 1.5; otherwise, the existing maximum value of 4GB applies. The maximum value allowed for SECQTY is 4GB.</p>
COMMENT	<p>New clause: <i>INDEX_index name</i></p> <p>Changed clause: <i>CCSID encoding-scheme</i></p> <p><i>INDEX_index name</i> identifies the index to which the comment applies. The name identifies an index that exists at the current server.</p> <p>The UNICODE option is added to the <i>CCSID encoding-scheme</i> clause. The UNICODE keyword specifies that the data must be encoded using the CCSIDs for Unicode data.</p>
CONNECT (TYPE 1 and 2)	<p>New clause: authorization clause (USER and USING)</p> <p>The authorization clause specifies an authorization ID (USER) and a password (USING) to verify a user ID that is being used to connect to a database server.</p>
CREATE DATABASE	<p>Changed clause: <i>CCSID encoding-scheme</i></p> <p>The UNICODE option is added to the <i>CCSID encoding-scheme</i> clause. The UNICODE keyword specifies that the data must be encoded using the CCSIDs for Unicode data.</p>

Table 16. Changes to existing SQL statements (continued)

SQL statement	Description of enhancements and notes
CREATE DISTINCT TYPE	<p>Changed clause: <i>CCSID encoding-scheme</i></p> <p>The UNICODE option is added to the <i>CCSID encoding-scheme</i> clause. The UNICODE keyword specifies that the data must be encoded using the CCSIDs for Unicode data.</p>
CREATE FUNCTION (external scalar)	<p>New clauses: INHERIT SPECIAL REGISTERS DEFAULT SPECIAL REGISTERS PARAMETER CCSID STATIC DISPATCH</p> <p>Changed clauses: LANGUAGE PARAMETER STYLE <i>CCSID encoding-scheme</i></p> <p>INHERIT SPECIAL REGISTERS indicates that special registers should be inherited according to rules for characteristics of special registers in a user-defined function. DEFAULT SPECIAL REGISTERS indicates that special registers should be initialized to the default values for such functions.</p> <p>PARAMETER CCSID provides a convenient way to specify an encoding scheme for string parameters other than the default encoding scheme. You don't have to explicitly specify one for each string parameter. In addition, system-generated parameters use the same encoding scheme as user-specified parameters.</p> <p>LANGUAGE is extended to allow for JAVA. LANGUAGE JAVA cannot be specified with a PARAMETER STYLE other than JAVA or with DBINFO, SCRATCHPAD, FINAL CALL, PROGRAM TYPE MAIN, or RUN OPTIONS.</p> <p>PARAMETER STYLE is extended to allow for JAVA. When LANGUAGE JAVA is specified, PARAMETER STYLE JAVA must also be specified.</p> <p>The UNICODE option is added to the <i>CCSID encoding-scheme</i> clause. The UNICODE keyword specifies that the data must be encoded using the CCSIDs for Unicode data.</p> <p>At function resolution, DB2 chooses a function based on the static (or declared) types of function parameters.</p>

Table 16. Changes to existing SQL statements (continued)

SQL statement	Description of enhancements and notes
CREATE FUNCTION (external table)	<p>New clauses: INHERIT SPECIAL REGISTERS DEFAULT SPECIAL REGISTERS PARAMETER CCSID STATIC DISPATCH</p> <p>Changed clauses: CCSID <i>encoding-scheme</i> FINAL CALL</p> <p>INHERIT SPECIAL REGISTERS indicates that special registers should be inherited according to rules for characteristics of special registers in a user-defined function. DEFAULT SPECIAL REGISTERS indicates that special registers should be initialized to the default values for such functions.</p> <p>PARAMETER CCSID provides a convenient way to specify an encoding scheme for string parameters other than the default encoding scheme. You don't have to explicitly specify one for each string parameter. In addition, system-generated parameters use the same encoding scheme as user-specified parameters.</p> <p>The UNICODE option is added to the CCSID <i>encoding-scheme</i> clause. The UNICODE keyword specifies that the data must be encoded using the CCSIDs for Unicode data.</p> <p>The default for NO FINAL CALL/FINAL CALL has changed from FINAL CALL to NO FINAL CALL.</p> <p>At function resolution, DB2 chooses a function based on the static (or declared) types of function parameters.</p>
CREATE FUNCTION (sourced)	<p>New clause: PARAMETER CCSID</p> <p>Changed clause: CCSID <i>encoding-scheme</i></p> <p>PARAMETER CCSID provides a convenient way to specify an encoding scheme for string parameters other than the default encoding scheme. You don't have to explicitly specify one for each string parameter. In addition, system-generated parameters use the same encoding scheme as user-specified parameters.</p> <p>The UNICODE option is added to the CCSID <i>encoding-scheme</i> clause. The UNICODE keyword specifies that the data must be encoded using the CCSIDs for Unicode data.</p>
CREATE GLOBAL TEMPORARY TABLE	<p>Changed clauses: CCSID <i>encoding-scheme</i> LIKE</p> <p>The UNICODE option is added to the CCSID <i>encoding-scheme</i> clause. The UNICODE keyword specifies that the data must be encoded using the CCSIDs for Unicode data.</p> <p>The default encoding scheme for tables that are created with the LIKE clause has changed. If the CCSID is not specified, the temporary table that is being defined inherits the encoding scheme of the table that is being copied.</p>

Table 16. Changes to existing SQL statements (continued)

SQL statement	Description of enhancements and notes
CREATE PROCEDURE (external procedure)	<p>New clauses: INHERIT SPECIAL REGISTERS DEFAULT SPECIAL REGISTERS PARAMETER CCSID</p> <p>Changed clauses: LANGUAGE PARAMETER STYLE</p> <p>INHERIT SPECIAL REGISTERS indicates that special registers should be inherited according to rules for characteristics of special registers in a user-defined function. DEFAULT SPECIAL REGISTERS indicates that special registers should be initialized to the default values for such functions.</p> <p>PARAMETER CCSID provides a convenient way to specify an encoding scheme for string parameters other than the default encoding scheme. You don't have to explicitly specify one for each string parameter. In addition, system-generated parameters use the same encoding scheme as user-specified parameters.</p> <p>LANGUAGE is extended to allow for JAVA. LANGUAGE JAVA cannot be specified with a PARAMETER STYLE other than JAVA or with DBINFO, PROGRAM TYPE MAIN, or RUN OPTIONS.</p> <p>PARAMETER STYLE is extended to allow for JAVA. When LANGUAGE JAVA is specified, PARAMETER STYLE JAVA must also be specified.</p>
CREATE PROCEDURE (SQL procedure)	<p>New clauses: INHERIT SPECIAL REGISTERS DEFAULT SPECIAL REGISTERS PARAMETER CCSID</p> <p>Changed clause: CCSID <i>encoding scheme</i> <i>parameter-declaration</i> <i>built-in-data-type</i></p> <p>INHERIT SPECIAL REGISTERS indicates that special registers should be inherited according to rules for characteristics of special registers in a user-defined function. DEFAULT SPECIAL REGISTERS indicates that special registers should be initialized to the default values for such functions.</p> <p>PARAMETER CCSID provides a convenient way to specify an encoding scheme for string parameters other than the default encoding scheme. You don't have to explicitly specify one for each string parameter. In addition, system-generated parameters use the same encoding scheme as user-specified parameters.</p> <p>The UNICODE option is added to the CCSID <i>encoding-scheme</i> clause. The UNICODE keyword specifies that the data must be encoded using the CCSIDs for Unicode data.</p> <p>The way in which you can use IN and OUT parameters has changed. See "IN and OUT parameters of an SQL procedure" on page 37.</p> <p>SQL procedures can be defined with BLOB, CLOB, and DBCLOB parameters.</p>

Table 16. Changes to existing SQL statements (continued)

SQL statement	Description of enhancements and notes
DECLARE CURSOR	<p>Changed clause: DECLARE <i>cursor-name</i></p> <p>SCROLL can now specify that a cursor is scrollable. Scrollable cursors need to be defined as INSENSITIVE (changes made to a database are not visible to the cursor) or SENSITIVE STATIC (changes made to a database are visible to the cursor, but the size of the result table does not change).</p>
DECLARE GLOBAL TEMPORARY TABLE	<p>Changed clauses: AS (<i>fullselect</i>) DEFINITION ONLY AS IDENTITY CCSID <i>encoding-scheme</i> LIKE</p> <p>Fullselect can now be used instead of subselect. The fullselect specifies that the columns of the table are to have the same name and description as the columns that would appear in the derived result table of the fullselect if the fullselect were to be executed.</p> <p>The AS IDENTITY clause has new keywords: CYCLE/NO CYCLE (indication of whether values for the column continue to be generated after the minimum or maximum value is reached), MAXVALUE (maximum value allowed for the column), and MINVALUE (minimum value allowed for the column).</p> <p>The UNICODE option is added to the CCSID <i>encoding-scheme</i> clause. The UNICODE keyword specifies that the data must be encoded using the CCSIDs for Unicode data.</p> <p>The default encoding scheme for tables created with the LIKE clause has changed. If the CCSID is not specified, the temporary table that is being defined inherits the encoding scheme of the table that is being copied.</p>
DELETE	<p>Changed clause: WHERE</p> <p>In the WHERE clause, the object of a searched DELETE statement can be the object of a subquery.</p>
DROP	<p>Changed clauses: CCSID <i>encoding-scheme</i> INDEX RESTRICT</p> <p>The UNICODE option is added to the CCSID <i>encoding-scheme</i> clause. The UNICODE keyword specifies that the data must be encoded using the CCSIDs for Unicode data.</p> <p>To drop a unique index that enforces a primary key, unique key, or referential constraint, you must first drop the constraint.</p>
#	<p>The RESTRICT clause is now a default.</p>
FETCH	<p>Keywords can now define a FETCH statement as INSENSITIVE (returns a row from the result table as it is) or SENSITIVE (updates the fetched row in the result table, including current values and changes made outside the cursor). Keywords that define the position of the scrollable cursor include NEXT, PRIOR, FIRST, LAST, CURRENT, BEFORE, AFTER, ABSOLUTE, and RELATIVE.</p>

Table 16. Changes to existing SQL statements (continued)

SQL statement	Description of enhancements and notes
GRANT	<p>Changed clause: <i>CCSID encoding-scheme</i></p> <p>The UNICODE option is added to the <i>CCSID encoding-scheme</i> clause. The UNICODE keyword specifies that the data must be encoded using the CCSIDs for Unicode data.</p>
GRANT (distinct type privileges)	<p>Changed statement name: GRANT (distinct type or JAR privileges)</p> <p>New clause: <i>JAR jar-name</i></p> <p>The GRANT (distinct type privileges) statement is modified to include JAR privileges to grant privilege to use identified jars. The JAR <i>jar-name</i> clause identifies a unique jar that exists at the current server.</p>
INSERT	<p>Fullselect is now supported instead of subselect. Fullselect specifies a set of new rows in the form of the result table of a fullselect.</p>
PREPARE	<p>New clause: <i>ATTRIBUTES attr-host-variable</i></p> <p>The new ATTRIBUTES clause allows you to specify the attributes for a cursor that are in effect if a corresponding attribute has not been specified on the PREPARE statement. The variable identifies a host variable that is described in a program.</p> <p>The options that can be specified as part of the attribute-string include INSENSITIVE or SENSITIVE STATIC, SCROLL, WITH HOLD, WITH RETURN, fetch-first-clause, read-only clause, update-clause, optimize-clause, and isolation-clause.</p>
REVOKE	<p>Changed clause: <i>CCSID encoding-scheme</i> RESTRICT</p> <p>The UNICODE option is added to the <i>CCSID encoding-scheme</i> clause. The UNICODE keyword specifies that the data must be encoded using the CCSIDs for Unicode data.</p>
#	<p>RESTRICT clause is now a default.</p>
REVOKE (distinct type privileges)	<p>Changed statement name: REVOKE (distinct type or JAR privileges)</p> <p>New clause: <i>JAR jar-name</i></p> <p>The REVOKE (distinct type privileges) statement is modified to include JAR privileges to revoke privilege to use identified jars. The JAR <i>jar-name</i> clause identifies a unique jar that exists at the current server.</p>

Table 16. Changes to existing SQL statements (continued)

SQL statement	Description of enhancements and notes
SELECT	<p>New clause: FETCH FIRST <i>n</i> ROWS ONLY</p> <p>Changed clause: FOR UPDATE</p> <p>The FETCH FIRST <i>n</i> ROWS ONLY clause limits the number of rows that DB2 prefetches and returns. When you execute a SELECT statement that contains the FETCH FIRST <i>n</i> ROWS ONLY clause, DB2 prefetches only <i>n</i> rows.</p> <p>In the FOR UPDATE clause, specifying columns is now optional. If a clause is specified without columns, the columns that are updated include all updatable columns identified in the first FROM clause of the fullselect.</p>
SELECT INTO	<p>New clause: FETCH FIRST ROW ONLY</p> <p>The FETCH FIRST ROW ONLY clause indicates that only one row should be retrieved regardless of how many rows are in the result table. Using this clause helps you avoid using a cursor when you know that you want to retrieve only one row.</p>
# SET CURRENT PATH	SET PATH can be used instead of SET CURRENT PATH.
UPDATE	<p>Changed clauses: SET <i>assignment-clause</i> WHERE</p> <p>In the SET <i>assignment-clause</i>, <i>row-fullselect</i> and <i>scalar-fullselect</i> can now be specified instead of <i>row-select</i> and <i>scalar-subselect</i>. <i>row-fullselect</i> specifies a fullselect that returns a single row; <i>scalar-fullselect</i> specifies a fullselect that returns a single row with a single column.</p> <p>In the WHERE clause, the object of a searched UPDATE statement can be the object of a subquery.</p>

New functions

Table 17 shows the new functions in Version 7, which improve the power of the SQL language.

Table 17. New functions

Function name	Description
Column functions	
STDDEV_POP	Returns the standard deviation $/n-1$ of a set of numbers, is an alternative to STDDEV
STDDEV_SAMP	Returns the standard deviation $/n-1$ of a set of numbers
VAR_POP	Returns the variance of a set of numbers, is an alternative to VARIANCE or VAR
VARIANCE_SAMP or VAR_SAMP	Returns the sample variance of a set of numbers
Scalar functions	
ADD_MONTHS	Returns a date that represents <i>date-expression</i> plus <i>expression</i> months
CCSID_ENCODING	Enables an application to determine if a CCSID for ASCII, EBCDIC, or Unicode data

Table 17. New functions (continued)

Function name	Description
DAYOFWEEK_ISO	Returns an integer in the range of 1 to 7 that represents the day of the week where 1 is Monday and 7 is Sunday
# GENERATE_UNIQUE #	Returns a bit character string that is unique compared to any other execution of the function
IDENTITY_VAL_LOCAL	Nondeterministic function that returns the most recently assigned value for an identity column
LAST_DAY	Returns a date that represents the last day of the month indicated by <i>date-expression</i>
MAX	Returns the maximum value in a set of values
MIN	Returns the minimum value in a set of values
# MQPUBLISH	Publishes a message to the specified MQSeries publisher
# MQREAD #	Returns a message from a specified MQSeries location (return value of VARCHAR) without removing the message from the queue
# MQREADCLOB #	Returns a message from a specified MQSeries location (return value of CLOB) without removing the message from the queue
# MQRECEIVE #	Returns a message from a specified MQSeries location (return value of VARCHAR) with removal of message from the queue
# MQRECEIVECLOB #	Returns a message from a specified MQSeries location (return value of CLOB) with removal of message from the queue
# MQSEND	Sends data contained in msg-data to a specified MQSeries location
# MQSUBSCRIBE #	Registers a subscription to MQSeries messages that are published on a specified topic
# MQUNSUBSCRIBE #	Unregisters an existing subscription to MQSeries messages that are published on a specified topic
NEXT_DAY	Returns a timestamp that represents the first weekday, named by <i>expression</i> , that is later than the date <i>date-expression</i>
ROUND_TIMESTAMP	Returns the <i>timestamp-expression</i> rounded to the unit specified by the <i>format-string</i>
TRUNC_TIMESTAMP	Returns the <i>timestamp-expression</i> truncated to the unit specified by the <i>format-string</i>
WEEK_ISO	Returns an integer in the range of 1 to 53 that represents the week of the year, starting with Monday and starting with the first week of the year that includes a Thursday
# Table functions	
# MQREADALL # #	Returns a table containing the messages and message metadata from a specified MQSeries location with a VARCHAR column and without removing the messages from the queue
# MQREADALLCLOB # #	Returns a table containing the messages and message metadata from a specified MQSeries location with a CLOB column and without removing the messages from the queue
# MQRECEIVEALL # #	Returns a table containing the messages and message metadata from a specified MQSeries location with a VARCHAR column and with removal of messages from the queue
# MQRECEIVEALLCLOB # #	Returns a table containing the messages and message metadata from a specified MQSeries location with a CLOB column and with removal of messages from the queue

Changed functions

As shown in Table 18, many of the existing functions are enhanced to support

Table 18. Changes to existing functions

Function name	Newly supported argument types
Column functions	
COUNT	ALL option is supported for COUNT(ALL <i>expression</i>) to count duplicate values but not null values for a set of rows or values
COUNT_BIG	ALL option is supported for COUNT_BIG(ALL <i>expression</i>) to count duplicate values but not null values for a set of rows or values
Scalar functions	
CHAR	Support is added for Unicode encoded strings
CLOB	Support is added for Unicode encoded strings
CONCAT	Support is added for GRAPHIC UTF-16 Unicode data and for VARGRAPHIC UTF-16 Unicode data
DECIMAL	Extended to support the ability to specify a decimal separator character when the first argument is character data
DIGITS	Support is added for Unicode encoded strings
GRAPHIC	Support is added for Unicode encoded strings
HEX	Support is added for Unicode encoded strings
INSERT	Support is added for Unicode encoded strings
LEFT	Support is added for Unicode encoded strings
LOWER/LCASE	Support is added for Unicode encoded strings
LENGTH	Support is added for Unicode encoded strings
LTRIM	Support is added for Unicode encoded strings
POSSTR	Support is added for Unicode encoded strings
REPEAT	Support is added for Unicode encoded strings
REPLACE	Support is added for Unicode encoded strings
RIGHT	Support is added for Unicode encoded strings
RTRIM	Support is added for Unicode encoded strings
SPACE	Support is added for Unicode encoded strings
STRIP	Support is added for Unicode encoded strings
UPPER/UCASE	Support is added for Unicode encoded strings
VARCHAR	Support is added for Unicode encoded strings
VARGRAPHIC	Support is added for Unicode encoded strings

Other SQL language changes

In addition to the many new SQL statements and functions in Version 7, Table 19 on page 120 shows some of the other enhancements to the SQL language.

Table 19. Other changes to SQL language

Item	Description
Special registers	CURRENT APPLICATION ENCODING SCHEME specifies which encoding scheme is to be used for dynamic statements that are executed after the set of the special register. CURRENT MEMBER specifies the member name of the current DB2 data sharing group on which the statement is executing.
Expressions	Nested table expressions in a FROM clause are changed to allow a fullselect where a subselect was used in previous releases.
Predicates	Basic, quantified, EXISTS, and IN predicates are changed to allow a fullselect where a subselect was used in previous releases.
# Limits #	Maximum number of databases is now 65217. Largest simple or segmented table space is 64 gigabytes.
# Reserved words	APPLICATION is no longer a reserved word.

Appendix D. Catalog changes

This appendix provides an overview of the changes to the catalog for Version 7 of DB2 for OS/390 and z/OS:

“New catalog tables”

“Changed catalog tables” on page 122

“New and revised indexes” on page 123

For a complete description of the columns of the new and changed catalog tables, see *DB2 SQL Reference*.

New catalog tables

Table 20 shows new catalog tables.

Table 20. New catalog tables

Catalog table name	Description
SYSIBM.SYSCHECKS2	Contains one row for each table check constraint
SYSIBM.SYSCOLDIST_HIST	Contains the same rows that are added or changed in SYSCOLDIST
SYSIBM.SYSCOLUMNS_HIST	Contains the same rows that are added or changed in SYSCOLUMNS
SYSIBM.SYSINDEXES_HIST	Contains the same rows that are added or changed in SYSINDEXES
SYSIBM.SYSINDEXPART_HIST	Contains the same rows that are added or changed in SYSINDEXPART
SYSIBM.SYSINDEXSTATS_HIST	Contains the same rows that are added or changed in SYSINDEXSTATS
SYSIBM.SYSJARCLASS_SOURCE	Auxiliary table for SYSIBM.SYSCONTENTS
SYSIBM.SYSJARCONTENTS	Contains the Java class source for the installed jar file
SYSIBM.SYSJARDATA	Auxiliary table for SYSIBM.SYSOBJECTS
SYSIBM.SYSJAROBJECTS	Contains binary large object representing the installed jar file
SYSIBM.SYSJAVA_OPTS	Contains build options that are used during INSTALL_JAR
SYSIBM.SYSKEYCOLUSE	Contains a row for every column in a unique constraint (primary or unique key) from the SYSIBM.SYSTABCONST table
SYSIBM.SYSLOBSTATS_HIST	Contains the same rows that are added or changed in SYLOBSTATS
SYSIBM.SYSROUTINES_OPTS	Contains a row for each generated routine that records the build options for the routine
SYSIBM.SYSROUTINES_SRC	Contains source for generated routines
SYSIBM.SYSTABCONST	Contains one row for each unique constraint (primary key or unique key) that is created in DB2 for OS/390 Version 7
SYSIBM.SYSTABLEPART_HIST	Contains the same rows added or changed in SYSTABLEPART
SYSIBM.SYSTABLES_HIST	Contains the same rows that are added or changed in SYSTABLES

Table 20. New catalog tables (continued)

Catalog table name	Description
SYSIBM.SYSTABSTATS_HIST	Contains the same rows that are added or changed in SYSTABSTATS

Changed catalog tables

Many existing catalog tables are changed in Version 7. Table 21 shows a list of the new columns that were added and the existing columns that were revised.

Table 21. Summary of new and revised catalog table columns

	Catalog table name	New column	Revised column
#	LUNAMES		SYSMODENAME
#	SYSCOLUMNS		FOREIGNKEY
	SYSCOPY	COPYPAGESF NPAGESF CPAGESF JOBNAME AUTHID	
	SYSDATABASE		ENCODING_SCHEME
	SYSDATATYPES		ENCODING_SCHEME
	SYSINDEXES	SPACEF REMARKS	
#	SYSINDEXPART	SPACEF DSNUM EXTENTS PSEUDO_DEL_ENTRIES LEAFNEAR LEAFFAR	SQTY SECQTY1
#	SYSPACKAGE	ENCODING_CCSID IMMEDWRITE RELBOUND	VALID
	SYSPACKSTMT	EXPLAINABLE QUERYNO	
	SYSPARMS		ENCODING_SCHEME
#	SYSPLAN	ENCODING_CCSID IMMEDWRITE RELBOUND	VALID
	SYSROUTINES	JAVA_SIGNATURE CLASS JARSHEMA JAR_ID SPECIAL_REGS	
	SYSSEQUENCES		MAXVALUE MINVALUE CYCLE
	SYSSTMT	EXPLAINABLE QUERYNO	
#	SYSTABLEPART	SPACEF DSNUM EXTENTS	SQTY SECQTY1

Table 21. Summary of new and revised catalog table columns (continued)

Catalog table name	New column	Revised column
SYSTABLES	NPAGESF SPACEF AVGROWLEN RELCREATED	ENCODING_SCHEME TABLESTATUS TSNAME
SYSTABLESPACE		ENCODING_SCHEME
SYSTRIGGERS	TRIGNAME	
SYSVIEWDEP	DTYPE	
SYSVIEWS	RELCREATED TYPE	

New and revised indexes

Table 22 shows the new and revised indexes in Version 7.

Table 22. New indexes

Table space DSNDB06. ...	Catalog table SYSIBM. ...	Index	Key column
SYSGRTNS	SYSROUTINES_OPTS	DSNRX01	SCHEMA.ROUTINENAME. BUILDDATE.BUILDTIME
	SYSROUTINES_SRC	DSNRX01	ROUTINENAME
		DSNRX02	SCHEMA.ROUTINENAME. BUILDDATE. SEQNO
SYSHIST	SYSCOLDIST_HIST	DSNHFX01	TBOWNER.TBNAME. NAME.STATSTIME
	SYSCOLUMNS_HIST	DSNHEX01	TBCREATOR.TBNAME. NAME.STATSTIME
		DSNHGX01	IXCREATOR.IXNAME. PARTITION.STATSTIME
	SYSINDEXES_HIST	DSNHXX01	TBCREATOR.TBNAME. NAME.STATSTIME
		DSNHXX02	TBCREATOR.TBNAME. NAME.STATSTIME
	DSNHXX02		CREATOR.NAME
	SYSINDEXSTATS_HIST	DSNHIX01	OWNER.NAME. PARTS.STATSTIME
	SYSLOBSTATS_HIST	DSNHJX01	DBNAME.NAME.STATSTIME
SYSJAVA	SYSTABLEPART_HIST	DSNHGX01	DBNAME.TSNAME. PARTITION.STATSTIME
	SYSTABLES_HIST	DSNHDX01	CREATOR.NAME.STATSTIME
	SYSTABSTATS_HIST	DSNHXX01	OWNER.NAME. PARTITION.STATSTIME
	SYSJARCONTENTS	DSNJCX01	JARSCHEMA.JAR_ID
	SYSJAROBJECTS	DSNJOX01	JARSCHEMA.JAR_ID
SYSJAUXA LOB	SYSJAVAOPTS	DSNJVX01	JARSCHEMA.JAR_ID
	SYSJARDATA	DSNJOX01	JAR_DATA

Table 22. New indexes (continued)

Table space DSNDB06. ...	Catalog table SYSIBM. ...	Index	Key column
SYSJAUXB LOB	SYSJARCLASS_SOURCE	DSNJOX01	CLASS_SOURCE
SYSOBJ	SYSKEYCOLUSE	DSNCUX01	TBCREATOR.TBNAME. CONSTNAME.COLSEQ.
	SYSROUTINES	DSNOFX07	NAME.PARM_COUNT. ROUTINETYPE.SCHEMA. PARM_SIGNATURE. PARM1.PARM2.PARM3. PARM4.PARM5.PARM6.PARM7. PARM8.PARM9.PARM10.PARM11. PARM12.PARM13.PARM14.PARM15. PARM16.PARM17.PARM18.PARM19. PARM20.PARM21.PARM22.PARM23. PARM24.PARM25.PARM26.PARM27. PARM28.PARM29.PARM30
		DSNOFX08	JARSCHEMA. JAR_ID
	SYSTABCONST	DSNCNX01	TBCREATOR.TBNAME.CONSTNAME
	SYSTABCONST	DSNCNX02	IXOWNER IXNAME
	SYSTRIGGERS	DSNOTX03	SCHEMA.TRIGNAME
SYSSTR	SYSCHECKS2	DSNCHX01	TBOWNER.TBNAME.CHECKNAME
SYSVIEWS	SYSVIEWS	DSNVVX01	CREATOR.NAME.SEQNO.TYPE

Appendix E. EXPLAIN table changes

The information in this appendix is Product-sensitive Programming Interface and Associated Guidance Information, as defined in “Notices” on page 141.

This appendix includes the complete definitions for a DB2 PLAN_TABLE. It also provides a description of the PLAN_TABLE columns that are new and changed for Version 7 of DB2 for OS/390 and z/OS.

Before you can use EXPLAIN, you must create a table called PLAN_TABLE to hold the results of EXPLAIN. If a PLAN_TABLE already exists, you can alter it to add the new columns. Figure 17 shows the format of the PLAN_TABLE. Table 23 on page 126 shows the content of each of the new or changed columns for Version 7.

Format of the Version 7 PLAN_TABLE

The Version 7 PLAN_TABLE has two additional columns, giving it a total of 51 columns.

QUERYNO	INTEGER	NOT NULL	PREFETCH	CHAR(1)	NOT NULL WITH DEFAULT
QBLOCKNO	SMALLINT	NOT NULL	COLUMN_FN_EVAL	CHAR(1)	NOT NULL WITH DEFAULT
APPLNAME	CHAR(8)	NOT NULL	MIXOPSEQ	SMALLINT	NOT NULL WITH DEFAULT
PROGNAME	CHAR(8)	NOT NULL	-----28 column format -----		
PLANNO	SMALLINT	NOT NULL	VERSION	VARCHAR(64)	NOT NULL WITH DEFAULT
METHOD	SMALLINT	NOT NULL	COLLID	CHAR(18)	NOT NULL WITH DEFAULT
CREATOR	CHAR(8)	NOT NULL	-----30 column format -----		
TNAME	CHAR(18)	NOT NULL	ACCESS_DEGREE	SMALLINT	
TABNO	SMALLINT	NOT NULL	ACCESS_PGROUP_ID	SMALLINT	
ACCESSTYPE	CHAR(2)	NOT NULL	JOIN_DEGREE	SMALLINT	
MATCHCOLS	SMALLINT	NOT NULL	JOIN_PGROUP_ID	SMALLINT	
ACCESSCREATOR	CHAR(8)	NOT NULL	-----34 column format -----		
ACCESSNAME	CHAR(18)	NOT NULL	SORTC_PGROUP_ID	SMALLINT	
INDEXONLY	CHAR(1)	NOT NULL	SORTN_PGROUP_ID	SMALLINT	
SORTN_UNIQ	CHAR(1)	NOT NULL	PARALLELISM_MODE	CHAR(1)	
SORTN_JOIN	CHAR(1)	NOT NULL	MERGE_JOIN_COLS	SMALLINT	
SORTN_ORDERBY	CHAR(1)	NOT NULL	CORRELATION_NAME	CHAR(18)	
SORTN_GROUPBY	CHAR(1)	NOT NULL	PAGE_RANGE	CHAR(1)	NOT NULL WITH DEFAULT
SORTC_UNIQ	CHAR(1)	NOT NULL	JOIN_TYPE	CHAR(1)	NOT NULL WITH DEFAULT
SORTC_JOIN	CHAR(1)	NOT NULL	GROUP_MEMBER	CHAR(8)	NOT NULL WITH DEFAULT
SORTC_ORDERBY	CHAR(1)	NOT NULL	IBM_SERVICE_DATA	VARCHAR(254)	NOT NULL WITH DEFAULT
SORTC_GROUPBY	CHAR(1)	NOT NULL	-----43 column format -----		
TSLOCKMODE	CHAR(3)	NOT NULL	WHEN_OPTIMIZE	CHAR(1)	NOT NULL WITH DEFAULT
TIMESTAMP	CHAR(16)	NOT NULL	QBLOCK_TYPE	CHAR(6)	NOT NULL WITH DEFAULT
REMARKS	VARCHAR(254)	NOT NULL	BIND_TIME	TIMESTAMP	NOT NULL WITH DEFAULT
-----25 column format -----			-----46 column format -----		
			OPTHINT	CHAR(8)	NOT NULL WITH DEFAULT
			HINT_USED	CHAR(8)	NOT NULL WITH DEFAULT
			PRIMARY_ACCESSTYPE	CHAR(1)	NOT NULL WITH DEFAULT
			-----49 column format-----		
			PARENT_QBLOCKNO	SMALLINT	NOT NULL WITH DEFAULT
			TABLE_TYPE	CHAR(1)	
			-----51 column format-----		

#

Figure 17. Format of PLAN_TABLE

Descriptions of new and changed columns in PLAN_TABLE

DB2 attempts to optimize a query that references a view or nested table expression that is defined with UNION or UNION ALL operators. When possible, DB2 removes unnecessary subselects, avoids materialization of intermediate results, and distributes joins and aggregations of the query across the subselects. Column QBLOCK_TYPE is changed to indicate when a UNION or UNION ALL operation is used. New column PARENT_QBLOCK identifies the QBLOCKNO of the parent query block for subselects in a UNION ALL operation. New column TABLE_TYPE identifies the type of table.

Table 23. Descriptions of new and changed columns in PLAN_TABLE

Column Name	Description
QBLOCK_TYPE	For each query block, an indication of the type of SQL operation performed. For the outermost query, this column identifies the statement type. Possible values: SELECT SELECT INSERT INSERT UPDATE UPDATE DELETE DELETE SELUPD SELECT with FOR UPDATE OF DELCUR DELETE WHERE CURRENT OF CURSOR UPDCUR UPDATE WHERE CURRENT OF CURSOR CORSUB Correlated subquery NCOSUB Noncorrelated subquery TABLEX Table expression UNION UNION UNIONA UNION ALL
# PARENT_QBLOCKNO	A number that indicates the QBLOCKNO of the parent query block.
TABLE_TYPE	The type of new table: F Table function Q Temporary intermediate result table (not materialized) T Table W Work file The value of the column is null if the query uses GROUP BY, ORDER BY, or DISTINCT, which requires an implicit sort.

Changed columns in DSN_STATEMNT_TABLE

When you use the DSN_STATEMNT_TABLE with EXPLAIN to find the estimated cost of your SQL statements, DB2 determines the cost category of each statement. If information that affects the estimate is missing or indeterminate, DB2 sets the COST_CATEGORY column to B. A value of B indicates that DB2 uses default values in the cost estimate. A subselect that contains a HAVING clause is a new condition in Version 7 for which DB2 assigns a statement to cost category B. The REASON column shows the new value HAVING CLAUSE to indicate this condition.

For some types of predicates, DB2 can change a HAVING clause to a WHERE clause. If all of the predicates are changed from a HAVING clause to a WHERE clause, DB2 does not assign the statement is cost category B unless it qualifies for other reasons.

Appendix F. New and changed IFCIDs

The information in this appendix is Product-sensitive Programming Interface and Associated Guidance Information, as defined in “Notices” on page 141.

This appendix briefly describes the new IFCIDs and the changes to the existing IFCIDs for each new function. The new IFCIDs are described in Table 24; the changes to existing IFCIDs are described in Table 25. For a detailed description of the fields in each IFCID record, refer to the mapping macros data set library *prefix.SDSNMACS*.

New IFCIDs

Table 24 lists the new IFCIDs.

Table 24. New IFCIDs

IFCID	Trace	Class	Mapping macro	Description
Improved monitoring of storage usage				
0217	GLOBAL	10	DSNDQW03	Records detailed information about storage usage in the DBM1 address space.
Utility lists with pattern matching and dynamic allocation				
0219	AUDIT	8	DSNDQW03	Records information about LISTDEF data sets.
	PERFORMANCE	10		
Utility lists with pattern matching and dynamic allocation				
0220	AUDIT	8	DSNDQW03	Records information about BSAM data sets for utilities.
	PERFORMANCE	10		
Improved monitoring of storage usage				
0225	STATISTICS	6	DSNDQW03	Records summary information about storage usage in the DBM1 address space.
Support for Windows Kerberos security				
0319	AUDIT	7	DSNDQW04	Provides an audit trail for security processing.

Changed IFCIDs

Table 25 gives an overview of changes to existing IFCIDs. Changes to IFCID 0106, the system parameters record, are not included.

Table 25. Changed IFCIDs

IFCID	Description of changes
Changing subsystem parameters without stopping DB2	
0001	Add a field for the number -SET SYSPARM commands that were executed.
Faster online REORG	
0023, 0024, 0025	Generate the following records for each subtask: <ul style="list-style-type: none">• One IFCID 0023 record at the start of the subtask• One IFCID 0024 record before each logical partition is updated• One IFCID 0025 at the end of the subtask
LOAD partition parallelism	

Table 25. Changed IFCIDs (continued)

IFCID	Description of changes
0023, 0024, 0025	Generate the following records for each subtask: <ul style="list-style-type: none"> • One IFCID 0023 record at the start of the subtask • One IFCID 0024 record before each partition is loaded • One IFCID 0025 at the end of the subtask
UNION everywhere	
0022	Add fields that correspond to new PLAN_TABLE columns.
UNLOAD and MODIFY STATISTICS utilities	
0023, 0024, 0025	Generate these records for UNLOAD, MODIFY STATISTICS, and COPY2COPY.
Windows Kerberos Security	
0312	This record is no longer written.
Miscellaneous changes	
0002	Add a field to record the number of times that pages were added to the LPL.
0003, 0147, 0148	Add new fields that provide more granular information on wait time due to global contention for locks.
0003, 0148	Add new fields that record page P-lock counters at the group buffer pool level.
0148	Add fields to track DDF block fetch information.
0023	Add flags to track the keywords that were used in a utility invocation.
0150	Add the following IFI qualifications: <ul style="list-style-type: none"> • A qualification to request data only for locks with waiters. • A qualification to request data only for lock in which multiple agents have an interest.
0254	Write this record as part of the Statistics trace.
0313	Add fields to record the following information about long-running units of recovery: <ul style="list-style-type: none"> • Whether DB2 reached an installation-defined limit for checkpoints or for log records. • The number of log records that DB2 wrote within a unit of recovery.
0314	Added fields to record information about DBADM authority on databases.

Appendix G. Prerequisites of Version 7 of DB2 for OS/390 and z/OS

This chapter identifies prerequisites and optional programs of DB2 for OS/390 and z/OS[®] Version 7 and for the features that are delivered with it. See “Function-Dependent Requirements for Features of DB2 Universal Database Server for OS/390 and z/OS” on page 136 for information about the requirements for the features of DB2 for OS/390 and z/OS Version 7. This chapter identifies the minimum level of hardware or software that is required; unless otherwise noted, subsequent versions and releases of these products are acceptable.

DB2 for OS/390 and z/OS Version 7 prerequisites

DB2 for OS/390 and z/OS Version 7 has requirements for processors, other programs, and virtual storage.

Hardware requirements

DB2 for OS/390 and z/OS Version 7 operates on any processor that OS/390 Version 2 Release 7 supports and that supports the architectural level set. In this version, DB2 for OS/390 and z/OS intends to use ESA/390 architectural enhancements that were implemented on selected IBM S/390[®] servers.

DB2 for OS/390 and z/OS Version 7 can run only on servers that implement the architectural enhancements and cannot run on any servers that have not implemented them.

The following IBM servers implement the architectural enhancements:

- zSeries 900 e-business enterprise server
- Models of the S/390 Parallel Enterprise Server[™] S/390 Parallel Enterprise Servers, except for Release 1 models
- All models of the S/390 Multiprise and the S/390 Multiprise 3000
- All models of the S/390 Application StarterPak Type 3000
- All PC Server System/390 servers and RS/6000[®] with System/390 Server-on-Board models
- All S/390 Integrated Servers

DB2 for OS/390 and z/OS Version 7 cannot run on the following IBM servers, because they do not implement the architectural enhancements:

- ES/9000[®] Processor Unit 9021, 9121, or 9221
- ES/3090 models
- ES/4381 models
- S/390 Parallel Transaction Server 9672 E or P models
- S/390 Parallel Enterprise Server 9672 Release 1 models

Architectural enhancements provide a variety of improvements of performance and of reliability. For more information about the specific enhancements in ESA/390, see IBM announcement letter 299-234 or the product manuals for ESA/390.

The processor must have enough real storage to satisfy the combined requirements of DB2, OS/390, the appropriate Data Facility Product, appropriate access methods, telecommunications, batch requirements, and other specific user-required applications.

The configuration must include sufficient I/O devices to support the requirements for system output, system residence, and system data sets. Sufficient direct access storage (direct access storage device (DASD) must be available to satisfy the user's information storage requirements . DASD can consist of any direct-access facility that is supported by the system configuration and the programming system.

In addition to listing data communications devices and auxiliary storage, this section identifies function-dependent hardware requirements and virtual storage requirements.

Auxiliary storage

DB2 is independent of tape device type and DASD. You can use any device that is supported by the data facilities component of OS/390 for the DB2 data sets. See Table 26 for a list of device types that are supported for DB2 data sets.

Table 26. Auxiliary storage

Data set type	Device type
Active recovery log data sets	Disk
Archive recovery log data sets	Disk, tape
Image copy data sets	Disk, tape
Bootstrap data set	Disk
User data sets	Disk
DB2 catalog data sets	Disk
Work data sets (for utilities)	Disk, tape

If these data sets are on disk that is shared with other z/OS and OS/390 systems, you should use global resource serialization to prevent concurrent access by more than one z/OS or OS/390 system.

The minimum DASD space requirement, based on installing DB2 using the panel default values, is approximately 600 MB. Users need additional disk space for their data.

If you use dual logging and tape for the log archiving device, you need at least two tape drives.

Data communication devices

Control DB2 operations from:

- The system console
- Authorized IMS/ESA Transaction Manager terminals
- Authorized CICS terminals
- TSO terminals by an authorized user

For information about the data communication devices that are supported by IMS/ESA Transaction Manager, CICS, and TSO, see the documentation for these products.

Function-dependent hardware requirements

Use of the FLOAT IEEE option of the UNLOAD utility requires the basic floating-point extensions facility. The facility is available on the following servers: zSeries, S/390 Parallel Enterprise Servers, Multiprise 3000 Enterprise Server, G5, or G6 processor or above.

The GENERATE_UNIQUE built-in function requires the extended time-of-day clock
facility that is available on the S/390 Parallel Enterprise Server Generation 5, the
S/390 Parallel Enterprise Server Generation 6, or on z/Architecture systems.

Program requirements and optional programs

This section lists licensed programs, or specific elements and features of licensed programs, that are required in the DB2 for OS/390 and z/OS Version 7 environment. You can use subsequent versions or releases of these programs, unless the description for a given program states otherwise. Check the RETAIN[®] Preventative Service Planning (PSP) Facility for the most current information about APARs you must install to run DB2 for OS/390 and z/OS and its optional features.

This section also identifies the requirements that are associated with specific DB2 capabilities, as well as optional programs that you can use with DB2 for OS/390 and z/OS Version 7.

Operating system and support programs

DB2 for OS/390 and z/OS Version 7 requires the function that is provided by the following licensed programs or their equivalents; subsequent versions or releases of any product are acceptable.

- One of the following:
 - z/OS Release 1 (5694-A01)
 - OS/390 Version 2 Release 7 System Services (5647-A01)
- DFSORT, part of the Application Enablement Services optional feature of z/OS and OS/390

Virtual storage requirements

The amount of space needed for the common service area (CSA) below the 16-MB line is less than 40 KB for each DB2 subsystem and 24 KB for each IRLM. High concurrent activity, parallelism, or high contention can require more CSA.

Most of the DB2 common data resides in the extended common service area (ECSA). Most modules, control blocks, and buffers reside in the extended private area. A DB2 subsystem with 200 concurrent users and 2000 open data sets should need less than 2 MB of virtual storage below the 16-MB line.

Function-dependent program requirements

DB2 for OS/390 and z/OS has the following requirements for specific licensed programs, or features of licensed programs:

Application execution: Applications written in high-level programming languages, such as applications or stored procedures written in the C language and using the Open Database Connectivity (ODBC) or CLI interfaces to DB2, require the Application Enablement Services element of OS/390 at run time. Applications or stored procedures written in Java, such as those using the JDBC or SQLJ interfaces to DB2, require Java for OS/390 (5655-A46) at run time. Applications written in Java are only supported on OS/390 Version 2 Release 8 or later.

Migration, Fallback, and Coexistence for DB2 for OS/390 Versions 5 and 6:

Before migrating from DB2 for OS/390 Version 5 or Version 6, you must install APAR PQ34467 to enable support for fallback or coexistence with Version 7.

Extenders:

Audio, Image, Video, and Net Search Extenders: Use of these extenders requires Language Environment, which is part of the Application Enablement Services element of OS/390 and z/OS.

Text Extender: Use of the Text Extender requires the following:

- The IBM Text Search Engine which is part of the e-business Services element of OS/390 Version 2 Release 7 or later
- Language Environment, which is part of the Application Enablement Services element of OS/390.

Specifically, the following functions require OS/390 Version 2 Release 9 or higher and the associated Text Search Engine:

- XML support
- The ability to specify multiple document model files for structured documents
- Support for CCSIDs 13488 (Unicode UCS2), and 1208 (Unicode UTF8), except for NGRAM indexes

DB2 Connect provides the Text Extender client, which optionally can be used with the DB2 for OS/390 Text Extender for administrative tasks.

XML Extender: Use of the XML Extender requires:

- Language Environment, which is part of the Application Enablement Services element of OS/390
- The IBM XML Parser for OS/390, C++ Edition to be available at run time

Release 2 of the IBM XML Toolkit for OS/390 (5655-D44) provides the required parser. Use of either the XMLFile type or the *Getting Started* material requires the UNIX System Services element of OS/390.

Support for Unicode: Use of Unicode data storage and manipulation capabilities requires:

- OS/390 Version 2 Release 8 (5647-A01), or later (with APAR OW44581 applied)
- Support for Unicode in OS/390 Version 2 Release 8, or later releases

OS/390 Version 2 Release 9 or later complements this capability with the Language Environment support in the Application Enablement Services element of OS/390.

Support for DB2 Precompiler Services: To use an SQL statement coprocessor, which implements DB2 precompiler services, your compilers need to be at the following levels:

- The SQL statement coprocessor for COBOL, which is referred to in COBOL documentation as the DB2 Coprocessor requires a minimum level of IBM COBOL for OS/390 & VM (5648-A25) Version 2 Release 2.
- The SQL statement coprocessor for COBOL that supports Unicode values in the UTF-16 format and the EBCDIC CCSID for COBOL character data types requires IBM Enterprise COBOL for z/OS and OS/390 (5655-G53) Version 3 Release 2.
- The SQL statement coprocessor for PL/I, which is referred to in PL/I documentation as the SQL preprocessor, requires a minimum level of IBM Enterprise PL/I for z/OS and OS/390 (5655-H31) Version 3 Release 1.

Windows Kerberos security: DB2 for OS/390 and z/OS Version 7 supports authentication by using Kerberos in conjunction with the following required software:

- Server support: DB2 for OS/390 and z/OS requires the SecureWay Security Server, an optional feature of OS/390 Version 2 Release 10. SecureWay Security Server includes the Network Authentication and Privacy Service based on MIT Kerberos Version 5 and RACF Program Control enhancements.

#

- Client support: DB2 for OS/390 and z/OS requires a client able to perform Windows Kerberos authentication over DRDA, such as DB2 Connect Version 7 for Windows. The client must provide support for single sign-on by using Kerberos security in Windows 2000.

#

MQSeries user-defined functions: DB2 for OS/390 and z/OS MQSeries user-defined functions require a minimum level of IBM MQSeries for OS/390 Version 5 Release 2 (5655-F10).

Optional programs

You can use DB2 for OS/390 and z/OS Version 7 with the specified optional licensed programs to enable the following functions:

DRDA connectivity: DB2 for OS/390 and z/OS Version 7 supports connection to the following relational database products:

- IBM DB2 Connect for Linux for S/390 and zSeries
- IBM DB2 Connect for Linux, UNIX, Windows, OS/2 Version 6 and 7
- IBM DB2 Universal Database for AS/400 Version 4 Release 2
- IBM Operating System/400® (OS/400®) Version 4 Release 1 with DB2 for AS/400 (5769-SS1)
- IBM DB2 Server for VM & VSE Version 7(5697-F42)
- IBM DB2 Server for VM & VSE Version 6 (5648-158)
- IBM DB2 DataJoiner Version 2 Release 1 Modification 1 (5231–200)
- Any other DRDA-compliant relational DBMS

DRDA requires the Communications Server of OS/390. Support over TCP/IP for OS/390 Version 2 Release 7 requires APAR PQ34286.

Web connectivity: The following products provide connectivity to DB2 for OS/390 Version 7 from the Web:

- WebSphere Application Server Version 3 Release 2 (5655-A98), or later releases
- Net.Data for OS/390, a feature of DB2 for OS/390 and z/OS Version 7
- Net Search Extender, a feature of DB2 UDB for OS/390 and z/OS Version 7

Capacity planning: IBM DB2 Estimator for Windows is an element of the DB2 Management Clients Package feature of DB2 for OS/390 and z/OS Version 7. The element works with DB2 data to estimate application feasibility, to model application cost and performance, and to estimate required CPU and I/O capacity.

Transaction management: The following transaction management products work with DB2:

- Information Management System (IMS):
 - IMS Version 7 (5655–B01)
 - IMS/ESA Version 6 (5655-158)
 - IMS/ESA Version 5 (5698-176)
- Customer Information Control System (CICS):
 - CICS Transaction Server for OS/390 Release 1 (5655-147)
 - CICS/ESA Version 4 (5655-018)

Query support: The following programs work with DB2:

- The Query Management Facility (QMF) family includes:
 - QMF for OS/390
 - QMF for Windows
 - QMF High Performance Option

- The DB2 Extenders for Text, Audio, Video, Image, and XML, which are elements of DB2 for OS/390 and z/OS Version 7
- The DB2 Net Search Extender

Application development tools and programming languages: The following application development tools and programming languages work with DB2 for OS/390 and z/OS Version 7:

Assembler	High-Level Assembler, which is part of the Systems Services element of OS/390 (5696–234)
C/C++	Any of the following products: <ul style="list-style-type: none"> • C/C++, with or without Debug Tool, part of the Application Enablement Services optional feature of OS/390 • IBM C/C++ for MVS/ESA Version 3 Release 2 (5655–121) • IBM AD/Cycle[®]C/370 Release 2 (5688–216)
COBOL	Any of the following products: <ul style="list-style-type: none"> • IBM COBOL for OS/390 and VM Version 2 (5648-A25) optionally, with IBM VisualAge COBOL Enterprise Version 2 Release 2 (04L6579) • IBM COBOL for MVS and VM Release 2 (5688-197) • VS COBOL II Release 4 (5668-958, 5688-023, 5688-805)
Fortran	VS Fortran Version 2 (5668-806, 5688–087, 5668–805)
Java	Applications or compiled stored procedures written in Java, such as those using the JDBC or SQLJ interfaces to DB2, require Java for OS/390 at the JDK 1.1.8 level (5655-A46). OS/390 Version 2 Release 8 or later supports these applications or stored procedures. Compiled Java programs and stored procedures also require VisualAge for Java, Enterprise Edition for OS/390 (5655-JAV). Interpreted Java user-defined functions and stored procedures have the following additional requirements: <ul style="list-style-type: none"> • IBM Developer Kit for OS/390, Java 2 Technology Edition at the SDK 1.3.1 level • Additional OS/390 APARs for interpreted Java routine security and WLM environment support <p>Before you can run a Java routine with SECURITY USER or SECURITY DEFINER, you need OS/390 APAR OW47578.</p> <p>To make the WLM address space operate more efficiently when a Java routine is running, install OS/390 APAR OW47978.</p>
PL/I	Any of the following products: <ul style="list-style-type: none"> • IBM Enterprise PL/I for z/OS and OS/390 Version 3 Release 1 (5655–H31) • IBM VisualAge PL/I for OS/390 Version 2 Release 2 (5655-B22) • IBM PL/I for MVS and VM Version 1 Release 1 (5688-235) • OS PL/I Version 2 Release 3 (5668-909, 5668-910, 5668-911)
# #	Use of the FLOAT(IEEE) precompiler option requires IBM Enterprise PL/I for z/OS and OS/390 Version 3 Release 1.
REXX	REXX support requires one of the following products: <ul style="list-style-type: none"> • REXX Version 1 Release 3 (5695–013 or 5695–014) • IBM TSO Extensions for MVS REXX, which is part of OS/390

SQL Procedure Language

To develop stored procedures using the SQL procedure language requires a C language compiler on OS/390

Operational support: The following programs provide operational support for DB2:

- DFSMS features, part of the Systems Management optional feature of OS/390:
 - DFSMSHsm for archiving
 - DFSMSdss for CONCURRENT COPY Utility
- RACF functions that are provided by the Security Server optional feature of OS/390
- IBM Softcopy Reader or Library Readers included on the CD-ROMs for BookManager[®] books
- The IBM Tools for Database Recovery and Replication Management, including the following tools:
 - IBM DB2 DataPropagator for OS/390 Version 7 (5655-E60)
 - IBM IMS DataPropagator Version 2 (5696-705)
 - IBM DB2 Recovery Manager for OS/390 (5697-F56)
 - IBM DB2 Row Archive Manager for OS/390 (5655-E65)
- The IBM Tools for Database Application Environments, including the following tools:
 - IBM DB2 Bind Manager (5655-D38)
 - IBM DB2 Web Query Tool (5655-E71)

Database administration and Systems Management Support: The following tools support database administration and systems management for DB2:

- The DB2 Management Clients Package, which is a feature of DB2 for OS/390 and z/OS Version 7 that includes:
 - DB2 Control Center
 - DB2 Stored Procedure Builder
 - DB2 Installer
 - DB2 for OS/390 and z/OS Visual Explain
 - DB2 Estimator

All editions of DB2 Universal Database for Windows, UNIX, OS/2 and the DB2 Connect products deliver DB2 Control Center and DB2 Stored Procedure Builder. IBM provides a restricted-use copy of DB2 Connect for Windows in the DB2 Management Clients Package feature of DB2 for OS/390 Version 7.

- The IBM Tools for Database Administration, including the following tools:
 - IBM DB2 Administration Tool Version 2 (5655-E70)
 - IBM DB2 Forms Version 3 (5697-G52)
 - IBM DB2 High Performance Unload (5655-E69)
 - IBM DB2 Automation Tool (5655-E72)
 - IBM DB2 Log Analysis Tool (5655-E66)
- The IBM Tools for Database Performance Management, including the following tools:
 - IBM DB2 Performance Monitor for OS/390 Version 7 (5655-E61)
 - IBM DB2 SQL Performance Analyzer for OS/390 (5697-F57)
 - IBM DB2 Query Monitor for OS/390 (5655-E67)

Function-Dependent Requirements for Features of DB2 Universal Database Server for OS/390 and z/OS

DB2 for OS/390 and z/OS Version 7 includes many features, some of which have requirements of their own, above and beyond what DB2 for OS/390 and z/OS Version 7 requires. This section identifies the requirements for using these features with DB2 for OS/390 and z/OS Version 7, but it does not repeat those DB2 for OS/390 and z/OS requirements that apply to the features. Also, some of these features can be used with prior releases of DB2.

Before using these features, refer to the installation information for these features to ensure that you have all required and recommended products.

DB2 Installer requirements

DB2 Installer is an element of the DB2 Management Clients Package, which is a feature of DB2 for OS/390 and z/OS Version 7. DB2 Installer has hardware and program requirements.

Hardware requirements

DB2 Installer requires:

- A workstation that is capable of running Windows or OS/2
- A monitor that is capable of displaying 800-by-600 resolution
- 25 MB of disk space on the target drive and 2 MB of disk space for each subsystem that is defined

Program requirements

DB2 Installer can run in any of the following environments, each of which has its own requirements:

- Microsoft Windows 2000,
- Windows NT Version 4.0
- Windows 95
- OS/2 Warp 4, optionally with either TCP/IP for OS/2 Version 3.0, or OS/2 Warp Connect

These environments require TCP/IP in any of the following circumstances:

- To run jobs from the workstation
- To use the copy-jobs-to-host function from the workstation

If you don't have TCP/IP, you can use DB2 Installer to customize your installation jobs, but you need to use a method outside of DB2 Installer to move jobs from the workstation to OS/390 for execution.

The GENERATE_UNIQUE built-in function requires OS/390 Release 2 Version 8 or
later.

DB2 for OS/390 and z/OS Visual Explain requirements

The Visual Explain element of the Data Management Clients Package has hardware and program requirements.

Hardware requirements

Visual Explain requires:

- A workstation that is capable of running Windows or OS/2
- A high-resolution monitor

- Approximately 12 MB of hard disk space

Program requirements

Visual Explain runs in any of the following operating systems:

- Windows 2000
- Windows NT Version 4.0
- OS/2 Warp Version 4

DB2 Connect Personal Edition Version 6, or later, must be installed on the DB2 Visual Explain workstation.

In addition, DB2 Visual Explain requires one of the following communication protocols:

- TCP/IP, part of the Communications Server element of OS/390
- SNA communications using a product such as Communication Server 5.0, SNA Server Version 4.0, or the integrated SNA support in DB2 Universal Database Personal Edition

Visual Explain includes a browser that lets users view current values of subsystem parameters. To use this browser, your DB2 subsystem must have:

- Stored procedures capability
- The DSNWZP stored procedure enabled

DB2 Estimator requirements

The DB2 Estimator element of the Data Management Clients Package has hardware and program requirements.

Hardware requirements

DB2 Estimator, which is an element of the DB2 Management Clients Package, requires:

- A workstation that is capable of running Windows
- A high-resolution monitor
- Approximately 12 MB of hard disk space

Program requirements

The DB2 Estimator operates in the Windows 2000, Windows NT 4.0, Windows 98, Windows 95 environments.

Net.Data requirements

Net.Data for OS/390 Version 7 has several requirements.

Program requirements

Net.Data requires an HTTP server to be installed on the same server as Net.Data. To configure Net.Data to execute as a servlet requires the following additional products:

- IBM WebSphereApplication Server for OS/390 Release 2 (5655-A98) or later releases
- Java Development Kit (JDK) for OS/390 Version 1.1.8 or later releases

DB2 Warehouse Center Requirements

DB2 Warehouse Center, an element of the DB2 Warehouse Manager, has the following requirements:

Program requirements

DB2 Warehouse Manager for OS/390 uses the warehouse server component of DB2 UDB Enterprise Edition Version 7 for Windows, which supports Microsoft Windows 2000 and Windows NT. A restricted-use copy of DB2 Universal Database Enterprise Edition is provided in the DB2 Warehouse Manager for OS/390 package to satisfy this functional dependency. The administrative client supports Windows, AIX, and Sun Solaris operating environments.

Data Warehouse Center provides agents for OS/390. These agents require:

- UNIX Systems Services element of OS/390
- TCP/IP, part of the Communications Server element of OS/390; OS/390 Version 2 Release 7 requires APAR PQ34286
- DB2 Java stored procedures enablement

Data Warehouse Center can support target data warehouses that are built on the following versions of DB2 Universal Database Server for OS/390 and z/OS:

- DB2 Universal Database Server for OS/390 Version 7 (5675-DB2)
- DB2 Universal Database Server for OS/390 Version 6 (5645-DB2)
- DB2 Server for OS/390 Version 5 (5655-DB2)

QMF requirements

QMF, QMF High Performance Option (HPO), and QMF for Windows have hardware and program requirements.

Hardware requirements

The following QMF features have hardware dependencies:

- QMF for OS/390 requires a display station that is supported by GDDM
- QMF High Performance Option (HPO) requires a display station that is supported by ISPF.
- QMF for Windows requires a workstation that supports:
 - A Windows 32-bit operating system
 - Network connectivity
 - Approximately 10 MB of hard disk space

Program requirements

The following QMF features have program dependencies.

- QMF for Windows requires:
 - A Windows 32-bit operating system
 - Network communication software on each user machine, plus one, or both, of the following programs:
 - An SNA product that provides a CPI-C interface
 - A TCP/IP product that provides a WinSock Version 1.1 interface

Use of QMF for Windows with English Wizard natural language query requires English Wizard Release 3.1, from Linguistic Technologies.

Appendix H. How to use the DB2 library

Titles of books in the library begin with DB2 Universal Database for OS/390 and z/OS Version 7. However, references from one book in the library to another are shortened and do not include the product name, version, and release. Instead, they point directly to the section that holds the information. For a complete list of books in the library, and the sections in each book, see the bibliography at the back of this book.

New Book in Version 7: *An Introduction to DB2 for OS/390* is new in Version 7. The book provides a comprehensive overview of DB2 Universal Database for OS/390 and z/OS . In addition to explaining basic concepts associated with relational database management systems, this book gives new users what they need to know before they begin using the most current version of the product.

The most rewarding task associated with a database management system is asking questions of it and getting answers, the task called *end use*. Other tasks are also necessary—defining the parameters of the system, putting the data in place, and so on. The tasks that are associated with DB2 are grouped into the following major categories (but supplemental information relating to all of the following tasks for new releases of DB2 can be found in this book):

Installation: If you are involved with DB2 only to install the system, *DB2 Installation Guide* might be all you need.

If you will be using data sharing capabilities you also need *DB2 Data Sharing: Planning and Administration*, which describes installation considerations for data sharing.

End use: End users issue SQL statements to retrieve data. They can also insert, update, or delete data, with SQL statements. They might need an introduction to SQL, detailed instructions for using SPUFI, and an alphabetized reference to the types of SQL statements. This information is found in *An Introduction to DB2 for OS/390*, *DB2 Application Programming and SQL Guide*, and *DB2 SQL Reference*.

End users can also issue SQL statements through the Query Management Facility (QMF™) or some other program, and the library for that licensed program might provide all the instruction or reference material they need. For a list of the titles in the QMF library, see the bibliography at the end of this book.

Application Programming: Some users access DB2 without knowing it, using programs that contain SQL statements. DB2 application programmers write those programs. Because they write SQL statements, they need *DB2 Application Programming and SQL Guide*, *DB2 SQL Reference*, and *DB2 ODBC Guide and Reference*, just as end users do.

Application programmers also need instructions on many other topics:

- How to transfer data between DB2 and a host program—written in COBOL, C, or Fortran, for example
- How to prepare to compile a program that embeds SQL statements
- How to process data from two systems simultaneously, say DB2 and IMS™ or DB2 and CICS
- How to write distributed applications across operating systems

- How to write applications that use DB2 ODBC to access DB2 servers
- How to write applications that use Open Database Connectivity (ODBC) to access DB2 servers
- How to write applications in the Java programming language to access DB2 servers

The material needed for writing a host program containing SQL is in *DB2 Application Programming and SQL Guide* and in *DB2 Application Programming Guide and Reference for Java*. The material needed for writing applications that use DB2 ODBC or ODBC to access DB2 servers is in *DB2 ODBC Guide and Reference*. For handling errors, see *DB2 Messages and Codes*.

Information about writing applications across operating systems can be found in *Distributed Relational Database Architecture: Application Programming Guide*.

System and Database Administration: *Administration* covers almost everything else. *DB2 Administration Guide* divides those tasks among the following sections:

- Part 2 (Volume 1) of *DB2 Administration Guide* discusses the decisions that must be made when designing a database and tells how to bring the design into being by creating DB2 objects, loading data, and adjusting to changes.
- Part 3 (Volume 1) of *DB2 Administration Guide* describes ways of controlling access to the DB2 system and to data within DB2, to audit aspects of DB2 usage, and to answer other security and auditing concerns.
- Part 4 (Volume 1) of *DB2 Administration Guide* describes the steps in normal day-to-day operation and discusses the steps one should take to prepare for recovery in the event of some failure.
- Part 5 (Volume 2) of *DB2 Administration Guide* explains how to monitor the performance of the DB2 system and its parts. It also lists things that can be done to make some parts run faster.

In addition, the appendixes in *DB2 Administration Guide* contain valuable information on DB2 sample tables, National Language Support (NLS), writing exit routines, interpreting DB2 trace output, and character conversion for distributed data.

If you are involved with DB2 only to design the database, or plan operational procedures, you need *An Introduction to DB2 for OS/390* and *DB2 Administration Guide*. If you also want to carry out your own plans by creating DB2 objects, granting privileges, running utility jobs, and so on, you also need:

- *DB2 SQL Reference*, which describes the SQL statements you use to create, alter, and drop objects and grant and revoke privileges
- *DB2 Utility Guide and Reference*, which explains how to run utilities
- *DB2 Command Reference*, which explains how to run commands

If you will be using data sharing, you need *DB2 Data Sharing: Planning and Administration*, which describes how to plan for and implement data sharing.

Additional information about system and database administration can be found in *DB2 Messages and Codes*, which lists messages and codes issued by DB2, with explanations and suggested responses.

Diagnosis: Diagnosticians detect and describe errors in the DB2 program. They might also recommend or apply a remedy. The documentation for this task is in *DB2 Diagnosis Guide and Reference* and *DB2 Messages and Codes*.

Notices

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

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

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

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

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

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

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

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

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs

and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
J46A/G4
555 Bailey Avenue
San Jose, CA 95141-1003
U.S.A.

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

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

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

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

This information is for planning purposes only. The information herein is subject to change before the products described become available.

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

COPYRIGHT LICENSE:

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

Programming interface information

This book is intended to help you to use the new commands and options of Version 7 of DB2 for OS/390 and z/OS and write programs that contain the new SQL statements and clauses of Version 7 of DB2 for OS/390 and z/OS. This book primarily documents General-use Programming Interface and Associated Guidance Information provided by IBM DATABASE 2 Universal Database Server for OS/390 and z/OS (DB2 for OS/390 and z/OS).

General-use programming interfaces allow the customer to write programs that obtain the services of DB2 for OS/390 and z/OS.

However, this book also documents Product-sensitive Programming Interface and Associated Guidance Information.

Product-sensitive programming interfaces allow the customer installation to perform tasks such as diagnosing, modifying, monitoring, repairing, tailoring, or tuning of this IBM software product. Use of such interfaces creates dependencies on the detailed design or implementation of the IBM software product. Product-sensitive programming interfaces should be used only for these specialized purposes. Because of their dependencies on detailed design and implementation, it is to be expected that programs written to such interfaces might need to be changed in order to run with new product releases or versions, or as a result of service.

Product-sensitive Programming Interface and Associated Guidance Information is identified where it occurs, by the following marking:

Product-sensitive Programming Interface
Product-sensitive Programming Interface and Associated Guidance Information ...
End of Product-sensitive Programming Interface

Trademarks

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

AD/Cycle	ES/4381
AIX	IBM
APL2	IBMLink
AS/400	IMS
BookManager	IMS/ESA
C/370	Language Environment
CICS	Multiprise
CICS/ESA	MVS/DFP
DataHub	MVS/ESA
DataJoiner	Net.Data
DataPropagator	OS/2
DB2	OS/390
DB2 Connect	Parallel Sysplex
DB2 OLAP Server	QMF
DB2 Universal Database	RACF
DFSMSdfp	RAMAC
DFSMSdss	SAA
DFSMSHsm	System/370

DFSMS/MVS	S/390
DFSORT	S/390 Parallel Enterprise Server
Distributed Relational Database Architecture	SecureWay
DRDA	System/390
Enterprise Storage Server	VisualAge
Enterprise System/3090	VTAM
Enterprise System/9000	WebSphere
ES/3090	Wizard
	z/OS

Domino is a trademark of Lotus Development Corporation in the United States, other countries, or both..

Tivoli and NetView are trademarks of Tivoli Systems Inc. in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

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

UNIX is a registered trademark of The Open Group in the United States and other countries.

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

Glossary

The following terms and abbreviations are defined as they are used in the DB2 library.

A

administrative authority. A set of related privileges that DB2 defines. When you grant one of the administrative authorities to a person's ID, the person has all of the privileges that are associated with that administrative authority.

after trigger. A trigger that is defined with the trigger activation time AFTER.

ASCII. An encoding scheme that is used to represent strings in many environments, typically on PCs and workstations. Contrast with *EBCDIC* and *Unicode*.

auxiliary index. An index on an auxiliary table in which each index entry refers to a LOB.

auxiliary table. A table that stores columns outside the table in which they are defined. Contrast with *base table*.

B

backout. The process of undoing uncommitted changes that an application process made. This might be necessary in the event of a failure on the part of an application process, or as a result of a deadlock situation.

base table. (1) A table that is created by the SQL CREATE TABLE statement and that holds persistent data. Contrast with *result table* and *temporary table*.

(2) A table containing a LOB column definition. The actual LOB column data is not stored with the base table. The base table contains a row identifier for each row and an indicator column for each of its LOB columns. Contrast with *auxiliary table*.

base table space. A table space that contains base tables.

before trigger. A trigger that is defined with the trigger activation time BEFORE.

binary large object (BLOB). A sequence of bytes where the size of the value ranges from 0 bytes to 2 GB-1. Such a string does not have an associated CCSID.

binary string. A sequence of bytes that is not associated with a CCSID. For example, the BLOB data type is a binary string.

BLOB. Binary large object.

block fetch. A capability in which DB2 can retrieve, or fetch, a large set of rows together. Using block fetch can significantly reduce the number of messages that are being sent across the network. Block fetch only applies to cursors that do not update data.

built-in function. A function that DB2 supplies. Contrast with *user-defined function*.

business dimension. A category of data, such as products or time periods, that an organization might want to analyze.

C

cast function. A function that is used to convert instances of a (source) data type into instances of a different (target) data type. In general, a cast function has the name of the target data type. It has one single argument whose type is the source data type; its return type is the target data type.

character conversion. The process of changing characters from one encoding scheme to another.

character large object (CLOB). A sequence of bytes representing single-byte characters or a mixture of single- and double-byte characters where the size of the value can be up to 2 GB-1. In general, character large object values are used whenever a character string might exceed the limits of the VARCHAR type.

character string. A sequence of bytes that represent bit data, single-byte characters, or a mixture of single-byte and multibyte characters.

check constraint. See *table check constraint*.

check integrity. The condition that exists when each row in a table conforms to the check constraints that are defined on that table. Maintaining check integrity requires DB2 to enforce check constraints on operations that add or change data.

class word. A single word that indicates the nature of a data attribute. For example, the class word PROJ indicates that the attribute identifies a project.

CLOB. Character large object.

code page. A set of assignments of characters to code points. In EBCDIC, for example, the character 'A' is assigned code point X'C1', and character 'B' is assigned code point X'C2'. Within a code page, each code point has only one specific meaning.

code point. In CDRA, a unique bit pattern that represents a character in a code page.

column function • double-byte character large object (DBCLOB)

column function. An operation that derives its result by using values from one or more rows. Contrast with *scalar function*.

contracting conversion. A process that occurs when the length of a converted string is smaller than that of the source string. For example, this process occurs when an EBCDIC mixed-data string that contains DBCS characters is converted to ASCII mixed data; the converted string is shorter because of the removal of the shift codes.

cost category. A category into which DB2 places cost estimates for SQL statements at the time the statement is bound. A cost estimate can be placed in either of the following cost categories:

- A: Indicates that DB2 had enough information to make a cost estimate without using default values.
- B: Indicates that some condition exists for which DB2 was forced to use default values for its estimate.

The cost category is externalized in the COST_CATEGORY column of the DSN_STATEMNT_TABLE when a statement is explained.

created temporary table. A table that holds temporary data and is defined with the SQL statement CREATE GLOBAL TEMPORARY TABLE. Information about created temporary tables is stored in the DB2 catalog, so this kind of table is persistent and can be shared across application processes. Contrast with *declared temporary table*. See also *temporary table*.

current SQL ID. An ID that, at a single point in time, holds the privileges that are exercised when certain dynamic SQL statements run. The current SQL ID can be a primary authorization ID or a secondary authorization ID.

cursor sensitivity. The degree to which database updates are visible to the subsequent FETCH statements in a cursor. A cursor can be sensitive to changes that are made with positioned update and delete statements specifying the name of that cursor. A cursor can also be sensitive to changes that are made with searched update or delete statements, or with cursors other than this cursor. These changes can be made by this application process or by another application process.

D

data dictionary. A repository of information about an organization's application programs, databases, logical data models, users, and authorizations. A data dictionary can be manual or automated.

data mart. A small data warehouse that applies to a single department or team. See also *data warehouse*.

data mining. The process of collecting critical business information from a data warehouse, correlating it, and uncovering associations, patterns, and trends.

data space. A range of up to 2 GB of contiguous virtual storage addresses that a program can directly manipulate. Unlike an address space, a data space can hold only data; it does not contain common areas, system data, or programs.

data warehouse. A system that provides critical business information to an organization. The data warehouse system cleanses the data for accuracy and currency, and then presents the data to decision makers so that they can interpret and use it effectively and efficiently.

DBCLOB. Double-byte character large object.

declared temporary table. A table that holds temporary data and is defined with the SQL statement DECLARE GLOBAL TEMPORARY TABLE. Information about declared temporary tables is not stored in the DB2 catalog, so this kind of table is not persistent and can only be used by the application process that issued the DECLARE statement. Contrast with *created temporary table*. See also *temporary table*.

delete hole. The location on which a cursor is positioned when a row in a result table is refetched and the row no longer exists on the base table, because another cursor deleted the row between the time the cursor first included the row in the result table and the time the cursor tried to refetch it.

delete trigger. A trigger that is defined with the triggering SQL operation DELETE.

denormalization. A key step in the task of building a physical relational database design. Denormalization is the intentional duplication of columns in multiple tables, and the consequence is increased data redundancy. Denormalization is sometimes necessary to minimize performance problems. Contrast with *normalization*.

deterministic function. A user-defined function whose result is dependent on the values of the input arguments. That is, successive invocations with the same input values produce the same answer. Sometimes referred to as a *not-variant* function. Contrast this with an *not-deterministic function* (sometimes called a *variant function*), which might not always produce the same result for the same inputs.

distinct type. A user-defined data type that is internally represented as an existing type (its source type), but is considered to be a separate and incompatible type for semantic purposes.

domain. The set of valid values for an attribute.

double-byte character large object (DBCLOB). A sequence of bytes representing double-byte characters

where the size of the values can be up to 2 GB. In general, double-byte character large object values are used whenever a double-byte character string might exceed the limits of the VARGRAPHIC type.

double-byte character set (DBCS). A set of characters, which are used by national languages such as Japanese and Chinese, that have more symbols than can be represented by a single byte. Each character is 2 bytes in length. Contrast with *single-byte character set* and *multibyte character set*.

E

EA-enabled table space. A table space or index space that is enabled for extended addressability and that contains individual partitions (or pieces, for LOB table spaces) that are greater than 4 GB.

EBCDIC. Extended binary coded decimal interchange code. An encoding scheme that is used to represent character data in the OS/390, MVS, VM, VSE, and OS/400® environments. Contrast with *ASCII* and *Unicode*.

e-business. The transformation of key business processes through the use of Internet technologies.

encoding scheme. A set of rules to represent character data (ASCII, EBCDIC, or Unicode).

entity. A significant object of interest to an organization.

enumerated list. A set of DB2 objects that are defined with a LISTDEF utility control statement in which pattern-matching characters (*, %, _ or ?) are not used.

exception table. A table that holds rows that violate referential constraints or table check constraints that the CHECK DATA utility finds.

expanding conversion. A process that occurs when the length of a converted string is greater than that of the source string. For example, this process occurs when an ASCII mixed-data string that contains DBCS characters is converted to an EBCDIC mixed-data string; the converted string is longer because of the addition of shift codes.

Extensible Markup Language (XML). A text-based tag language that organizations use for document processing and for publishing information on the Web.

external function. A function for which the body is written in a programming language that takes scalar argument values and produces a scalar result for each invocation. Contrast with *sourced function*, *built-in function*, and *SQL function*.

external procedure. An application program written by a user that can be invoked with the SQL CALL statement written in a programming language. Contrast with *SQL procedure*.

F

federated database. The combination of a DB2 Universal Database server (in UNIX®, Windows®, and OS/2® environments) and multiple data sources to which the server sends queries. In a federated database system, a client application can use a single SQL statement to join data that is distributed across multiple database management systems and can view the data as if it were local.

fetch orientation. The specification of the desired placement of the cursor as part of a FETCH statement (for example, BEFORE, AFTER, NEXT, PRIOR, CURRENT, FIRST, LAST, ABSOLUTE, and RELATIVE).

fullselect. A subselect, a values-clause, or a number of both that are combined by set operators. *Fullselect* specifies a result table. If UNION is not used, the result of the fullselect is the result of the specified subselect.

function. A mapping, embodied as a program (the function body), invocable by means of zero or more input values (arguments), to a single value (the result). See also *column function* and *scalar function*.

Functions can be user-defined, built-in, or generated by DB2. (See *built-in function*, *cast function*, *external function*, *sourced function*, *SQL function*, and *user-defined function*.)

function definer. The authorization ID of the owner of the schema of the function that is specified in the CREATE FUNCTION statement.

function implementer. The authorization ID of the owner of the function program and function package.

function package. A package that results from binding the DBRM for a function program.

function package owner. The authorization ID of the user who binds the function program's DBRM into a function package.

function resolution. The process, internal to the DBMS, by which a function invocation is bound to a particular function instance. This process uses the function name, the data types of the arguments, and a list of the applicable schema names (called the *SQL path*) to make the selection. This process is sometimes called *function selection*.

function selection. See *function resolution*.

function signature. The logical concatenation of a fully qualified function name with the data types of all of its parameters.

G

group buffer pool duplexing. The ability to write data to two instances of a group buffer pool structure: a *primary group buffer pool* and a *secondary group buffer pool*. OS/390 publications refer to these instances as the "old" (for primary) and "new" (for secondary) structures.

I

identity column. A column that provides a way for DB2 to automatically generate a numeric value for each row. The generated values are unique if cycling is not used. Identity columns are defined with the AS IDENTITY clause. Uniqueness of values can be ensured by defining a single-column unique index using the identity column. A table can have no more than one identity column.

indicator column. A 4-byte value that is stored in a base table in place of a LOB column.

inheritance. The passing of class resources or attributes from a parent class downstream in the class hierarchy to a child class.

inoperative package. A package that cannot be used because one or more user-defined functions or procedures that the package depends on were dropped. Such a package must be explicitly rebound. Contrast with *invalid package*.

insensitive cursor. A cursor that is not sensitive to inserts, updates, or deletes that are made to the underlying rows of a result table after the result table has materialized.

insert trigger. A trigger that is defined with the triggering SQL operation INSERT.

internationalization. The support for an encoding scheme that is able to represent the code points of characters from many different geographies and languages. To support all geographies, the Unicode standard requires more than 1 byte to represent a single character. See also *Unicode*.

invalid package. A package that depends on an object (other than a user-defined function) that is dropped. Such a package is implicitly rebound on invocation. Contrast with *inoperative package*.

invariant character set. (1) A character set, such as the syntactic character set, whose code point assignments do not change from code page to code page. (2) A minimum set of characters that is available as part of all character sets.

J

Java[®] Archive (JAR). A file format that is used for aggregating many files into a single file.

L

large object (LOB). A sequence of bytes representing bit data, single-byte characters, double-byte characters, or a mixture of single- and double-byte characters. A LOB can be up to 2 GB–1 byte in length. See also *BLOB*, *CLOB*, and *DBCLOB*.

list. A type of object, which DB2 utilities can process, that identifies multiple table spaces, multiple index spaces, or both. A list is defined with the LISTDEF utility control statement.

LOB. Large object.

LOB locator. A mechanism that allows an application program to manipulate a large object value in the database system. A LOB locator is a fullword integer value that represents a single LOB value. An application program retrieves a LOB locator into a host variable and can then apply SQL operations to the associated LOB value using the locator.

LOB lock. A lock on a LOB value.

LOB table space. A table space that contains all the data for a particular LOB column in the related base table.

locale. The definition of a subset of a user's environment that combines characters that are defined for a specific language and country, and a CCSID.

logical data modeling. The process of documenting the comprehensive business information requirements in an accurate and consistent format. Data modeling is the first task of designing a database.

M

mass delete. The deletion of all rows of a table.

materialize. (1) The process of putting rows from a view or nested table expression into a work file for additional processing by a query.

(2) The placement of a LOB value into contiguous storage. Because LOB values can be very large, DB2 avoids materializing LOB data until doing so becomes absolutely necessary.

MBCS. Multibyte character set. UTF-8 is an example of an MBCS. Characters in UTF-8 can range from 1 to 4 bytes in DB2.

mixed data string. A character string that can contain both single-byte and double-byte characters.

modeling database. A DB2 database that you create on your workstation that you use to model a DB2 for OS/390 subsystem, which can then be evaluated by the Index Advisor.

multibyte character set (MBCS). A character set that represents single characters with more than a single byte. Contrast with *single-byte character set* and *double-byte character set*. See also *Unicode*.

multidimensional analysis. The process of assessing and evaluating an enterprise on more than one level.

N

nonscrollable cursor. A cursor that can be moved only in a forward direction. Nonscrollable cursors are sometimes called forward-only cursors or serial cursors.

normalization. A key step in the task of building a logical relational database design. Normalization helps you avoid redundancies and inconsistencies in your data. An entity is normalized if it meets a set of constraints for a particular normal form (first normal form, second normal form, and so on). Contrast with *denormalization*.

not-deterministic function. A user-defined function whose result is not solely dependent on the values of the input arguments. That is, successive invocations with the same argument values can produce a different answer. This type of function is sometimes called a *variant function*. Contrast this with a *deterministic function* (sometimes called a *not-variant function*), which always produces the same result for the same inputs.

not-variant function. See *deterministic function*.

O

overloaded function. A function name for which multiple function instances exist.

P

parallel complex. A cluster of machines that work together to handle multiple transactions and applications.

partitioning index. An index that determines how rows are physically ordered in a partitioned table space. Partitioning indexes are also clustering indexes.

path. See *SQL path*.

postponed abort UR. A unit of recovery that was inflight or in-abort, was interrupted by system failure or cancellation, and did not complete backout during restart.

primary group buffer pool. For a duplexed group buffer pool, the structure used to maintain the coherency of cached data. This structure is used for page registration and cross-invalidation. The OS/390 equivalent is *old* structure. Compare with *secondary group buffer pool*.

principal. An entity that can communicate securely with another entity. In Kerberos, principals are represented as entries in the Kerberos registry database and include users, servers, computers, and others.

R

RACF. Resource Access Control Facility, which is a component of the SecureWay Security Server for OS/390.

record. The storage representation of a row or other data.

registry database. A database of security information about principals, groups, organizations, accounts, and security policies.

REORG pending (REORP). A condition that restricts SQL access and most utility access to an object that must be reorganized.

REORP. REORG pending.

repeating group. A situation in which an entity includes multiple attributes that are inherently the same. The presence of a repeating group violates the requirement of first normal form. In an entity that satisfies the requirement of first normal form, each attribute is independent and unique in its meaning and its name. See also *normalization*.

restart pending (RESTP). A restrictive state of a page set or partition that indicates that restart (backout) work needs to be performed on the object. All access to the page set or partition is denied except for access by the:

- RECOVER POSTPONED command
- Automatic online backout (which DB2 invokes after restart if the system parameter LBACKOUT=AUTO)

RESTP. Restart pending.

result set. The set of rows that a stored procedure returns to a client application.

result table. The set of rows that are specified by a SELECT statement.

ROWID. Row identifier.

row identifier (ROWID). A value that uniquely identifies a row. This value is stored with the row and never changes.

row trigger. A trigger that is defined with the trigger granularity FOR EACH ROW.

row-value-expression • structure

row-value-expression. A comma-separated list of value expressions enclosed in parentheses.

S

savepoint. A named entity that represents the state of data and schemas at a particular point in time within a unit of work. SQL statements exist to set a savepoint, release a savepoint, and restore data and schemas to the state that the savepoint represents. The restoration of data and schemas to a savepoint is usually referred to as *rolling back to a savepoint*.

scalar function. An SQL operation that produces a single value from another value and is expressed as a function name, followed by a list of arguments that are enclosed in parentheses. Contrast with *column function*.

schema. A logical grouping for user-defined functions, distinct types, triggers, and stored procedures. When an object of one of these types is created, it is assigned to one schema, which is determined by the name of the object. For example, the following statement creates a distinct type T in schema C:

```
CREATE DISTINCT TYPE C.T ...
```

scrollability. The ability to use a cursor to fetch in either a forward or backward direction. The FETCH statement supports multiple fetch orientations to indicate the new position of the cursor. See also *fetch orientation*.

scrollable cursor. A cursor that can be moved in both a forward and a backward direction.

secondary group buffer pool. For a duplexed group buffer pool, the structure that is used to back up changed pages that are written to the primary group buffer pool. No page registration or cross-invalidation occurs using the secondary group buffer pool. The OS/390 equivalent is *new* structure.

segment. A group of pages that holds rows of a single table. See also *segmented table space*.

sensitive cursor. A cursor that is sensitive to changes made to the database after the result table has materialized.

serial cursor. A cursor that can be moved only in a forward direction.

server-side programming. A method for adding DB2 data into dynamic Web pages. Three common types of server-side programs are Common Gateway Interface (CGI) programs, Web server API programs, and Java servlets.

SGML. Standard Generalized Markup Language.

single-byte character set (SBCS). A set of characters in which each character is represented by a single byte. Contrast with *double-byte character set* or *multibyte character set*.

sourced function. A function that is implemented by another built-in or user-defined function that is already known to the database manager. This function can be a scalar function or a column (aggregating) function; it returns a single value from a set of values (for example, MAX or AVG). Contrast with *built-in function*, *external function*, and *SQL function*.

source type. An existing type that is used to internally represent a distinct type.

specific function name. A particular user-defined function that is known to the database manager by its specific name. Many specific user-defined functions can have the same function name. When a user-defined function is defined to the database, every function is assigned a specific name that is unique within its schema. Either the user can provide this name, or a default name is used.

SQL function. A user-defined function in which the CREATE FUNCTION statement contains the source code. The source code is a single SQL expression that evaluates to a single value. The SQL user-defined function can return only one parameter.

SQL path. An ordered list of schema names that are used in the resolution of unqualified references to user-defined functions, distinct types, and stored procedures. In dynamic SQL, the current path is found in the CURRENT PATH special register. In static SQL, it is defined in the PATH bind option.

SQL procedure. A user-written program that can be invoked with the SQL CALL statement. Contrast with *external procedure*.

SQL statement coprocessor. An alternative to the DB2 precompiler that lets the user process SQL statements at compile time. The user invokes an SQL statement coprocessor by specifying a compiler option.

statement trigger. A trigger that is defined with the trigger granularity FOR EACH STATEMENT.

strong typing. A process that guarantees that only user-defined functions and operations that are defined on a distinct type can be applied to that type. For example, you cannot directly compare two currency types, such as Canadian dollars and U.S. dollars. But you can provide a user-defined function to convert one currency to the other and then do the comparison.

structure. A construct that uses MVS to map and manage storage on a coupling facility. See *cache structure*, *list structure*, or *lock structure*.

subject table. The table for which a trigger is created. When the defined triggering event occurs on this table, the trigger is activated.

surrogate pair. A coded representation for a single character that consists of a sequence of two Unicode values, where the first value of the pair is a high-surrogate in the range U+D800 through U+DBFF, and the second value is a low-surrogate in the range U+DC00 through U+DFFF. Surrogate pairs provide an extension mechanism for encoding 917 476 characters without requiring the use of 32-bit characters.

syntactic character set. A set of 81 graphic characters that are registered in the IBM registry as character set 00640. This set was originally recommended to the programming language community to be used for syntactic purposes toward maximizing portability and interchangeability across systems and country boundaries. It is contained in most of the primary registered character sets, with a few exceptions. See also *invariant character set*.

T

table function. A function that receives a set of arguments and returns a table to the SQL statement that references the function. A table function can be referenced only in the FROM clause of a subselect.

table locator. A mechanism that allows access to trigger transition tables in the FROM clause of SELECT statements, the subselect of INSERT statements, or from within user-defined functions. A table locator is a fullword integer value that represents a transition table.

table space set. A set of table spaces and partitions that should be recovered together for one of these reasons:

- Each of them contains a table that is a parent or descendent of a table in one of the others.
- The set contains a base table and associated auxiliary tables.

A table space set can contain both types of relationships.

TB. Terabyte (1 099 511 627 776 bytes).

template. A DB2 utilities output data set descriptor that is used for dynamic allocation. A template is defined by the TEMPLATE utility control statement.

temporary table. A table that holds temporary data; for example, temporary tables are useful for holding or sorting intermediate results from queries that contain a large number of rows. The two kinds of temporary table, which are created by different SQL statements, are the created temporary table and the declared temporary table. Contrast with *result table*. See also *created temporary table* and *declared temporary table*.

transition table. A temporary table that contains all the affected rows of the subject table in their state before or after the triggering event occurs. Triggered SQL statements in the trigger definition can reference the table of changed rows in the old state or the new state.

transition variable. A variable that contains a column value of the affected row of the subject table in its state before or after the triggering event occurs. Triggered SQL statements in the trigger definition can reference the set of old values or the set of new values.

trigger. A set of SQL statements that are stored in a DB2 database and executed when a certain event occurs in a DB2 table.

trigger activation. The process that occurs when the trigger event that is defined in a trigger definition is executed. Trigger activation consists of the evaluation of the triggered action condition and conditional execution of the triggered SQL statements.

trigger activation time. An indication in the trigger definition of whether the trigger should be activated before or after the triggered event.

trigger body. The set of SQL statements that is executed when a trigger is activated and its triggered action condition evaluates to true.

trigger cascading. The process that occurs when the triggered action of a trigger causes the activation of another trigger.

triggered action. The SQL logic that is performed when a trigger is activated. The triggered action consists of an optional triggered action condition and a set of triggered SQL statements that are executed only if the condition evaluates to true.

triggered action condition. An optional part of the triggered action. This Boolean condition appears as a WHEN clause and specifies a condition that DB2 evaluates to determine if the triggered SQL statements should be executed.

triggered SQL statements. The set of SQL statements that is executed when a trigger is activated and its triggered action condition evaluates to true. Triggered SQL statements are also called the *trigger body*.

trigger granularity. A characteristic of a trigger, which determines whether the trigger is activated:

- Only once for the triggering SQL statement
- Once for each row that the SQL statement modifies

triggering event. The specified operation in a trigger definition that causes the activation of that trigger. The triggering event is comprised of a triggering operation (INSERT, UPDATE, or DELETE) and a subject table on which the operation is performed.

triggering SQL operation • z/OS

triggering SQL operation. The SQL operation that causes a trigger to be activated when performed on the subject table.

trigger package. A package that is created when a CREATE TRIGGER statement is executed. The package is executed when the trigger is activated.

typed parameter marker. A parameter marker that is specified along with its target data type. It has the general form:

CAST(? AS data-type)

U

UCS-2. Universal Character Set, coded in 2 octets, which means that characters are represented in 16-bits per character.

UDF. User-defined function.

UDT. User-defined data type. In DB2 for OS/390 and z/OS, the term *distinct type* is used instead of user-defined data type. See *distinct type*.

Unicode. A standard that parallels the ISO-10646 standard. Several implementations of the Unicode standard exist, all of which have the ability to represent a large percentage of the characters contained in the many scripts that are used throughout the world.

uniform resource locator (URL). A Web address, which offers a way of naming and locating specific items on the Web.

union. An SQL operation that combines the results of two select statements. Unions are often used to merge lists of values that are obtained from several tables.

untyped parameter marker. A parameter marker that is specified without its target data type. It has the form of a single question mark (?).

updatability. The ability of a cursor to perform positioned updates and deletes. The updatability of a cursor can be influenced by the SELECT statement and the cursor sensitivity option that is specified on the DECLARE CURSOR statement.

update hole. The location on which a cursor is positioned when a row in a result table is fetched again and the new values no longer satisfy the search condition, because another cursor updated the row between the time the cursor first included the row in the result table and the time the cursor tried to refetch it.

update trigger. A trigger that is defined with the triggering SQL operation UPDATE.

URL. Uniform resource locator.

user-defined data type (UDT). See *distinct type*.

user-defined function (UDF). A function that is defined to DB2 by using the CREATE FUNCTION statement and that can be referenced thereafter in SQL statements. A user-defined function can be an *external function*, a *sourced function*, or an *SQL function*. Contrast with *built-in function*.

user view. In logical data modeling, a model or representation of critical information that the business requires.

UTF-8. Unicode Transformation Format, 8-bit encoding form, which is designed for ease of use with existing ASCII-based systems. The CCSID value for data in UTF-8 format is 1208. DB2 for OS/390 and z/OS supports UTF-8 in mixed data fields.

UTF-16. Unicode Transformation Format, 16-bit encoding form, which is designed to provide code values for over a million characters and a superset of UCS-2. The CCSID value for data in UTF-16 format is 1200. DB2 for OS/390 and z/OS supports UTF-16 in graphic data fields.

V

variant function. See *not-deterministic function*.

Z

z/OS. An operating system for the eServer product line that supports 64-bit real storage.

Bibliography

DB2 Universal Database Server for OS/390 and z/OS Version 7 product libraries:

DB2 for OS/390 and z/OS

- *An Introduction to DB2 for OS/390*, SC26-9937
- *DB2 Administration Guide*, SC26-9931
- *DB2 Application Programming and SQL Guide*, SC26-9933
- *DB2 Application Programming Guide and Reference for Java*, SC26-9932
- *DB2 Command Reference*, SC26-9934
- *DB2 Data Sharing: Planning and Administration*, SC26-9935
- *DB2 Data Sharing Quick Reference Card*, SX26-3846
- *DB2 Diagnosis Guide and Reference*, LY37-3740
- *DB2 Diagnostic Quick Reference Card*, LY37-3741
- *DB2 Image, Audio, and Video Extenders Administration and Programming*, SC26-9947
- *DB2 Installation Guide*, GC26-9936
- *DB2 Licensed Program Specifications*, GC26-9938
- *DB2 Master Index*, SC26-9939
- *DB2 Messages and Codes*, GC26-9940
- *DB2 ODBC Guide and Reference*, SC26-9941
- *DB2 Reference for Remote DRDA Requesters and Servers*, SC26-9942
- *DB2 Reference Summary*, SX26-3847
- *DB2 Release Planning Guide*, SC26-9943
- *DB2 SQL Reference*, SC26-9944
- *DB2 Text Extender Administration and Programming*, SC26-9948
- *DB2 Utility Guide and Reference*, SC26-9945
- *DB2 What's New?* GC26-9946
- *DB2 XML Extender for OS/390 and z/OS Administration and Programming*, SC27-9949
- *DB2 Program Directory*, GI10-8182

DB2 Administration Tool

- *DB2 Administration Tool for OS/390 and z/OS User's Guide*, SC26-9847

DB2 Buffer Pool Tool

- *DB2 Buffer Pool Tool for OS/390 and z/OS User's Guide and Reference*, SC26-9306

DB2 DataPropagator™

- *DB2 UDB Replication Guide and Reference*, SC26-9920

Net.Data®

The following books are available at this Web site: www.ibm.com/software/net.data/library.html

- *Net.Data Library: Administration and Programming Guide for OS/390 and z/OS*
- *Net.Data Library: Language Environment Interface Reference*
- *Net.Data Library: Messages and Codes*
- *Net.Data Library: Reference*

DB2 PM for OS/390

- *DB2 PM for OS/390 Batch User's Guide*, SC27-0857
- *DB2 PM for OS/390 Command Reference*, SC27-0855
- *DB2 PM for OS/390 Data Collector Application Programming Interface Guide*, SC27-0861
- *DB2 PM for OS/390 General Information*, GC27-0852
- *DB2 PM for OS/390 Installation and Customization*, SC27-0860
- *DB2 PM for OS/390 Messages*, SC27-0856
- *DB2 PM for OS/390 Online Monitor User's Guide*, SC27-0858
- *DB2 PM for OS/390 Report Reference Volume 1*, SC27-0853
- *DB2 PM for OS/390 Report Reference Volume 2*, SC27-0854
- *DB2 PM for OS/390 Using the Workstation Online Monitor*, SC27-0859
- *DB2 PM for OS/390 Program Directory*, GI10-8223

Query Management Facility (QMF™)

- *Query Management Facility: Developing QMF Applications*, SC26-9579
- *Query Management Facility: Getting Started with QMF on Windows*, SC26-9582
- *Query Management Facility: High Performance Option User's Guide for OS/390 and z/OS*, SC26-9581

- *Query Management Facility: Installing and Managing QMF on OS/390 and z/OS, GC26-9575*
- *Query Management Facility: Installing and Managing QMF on Windows, GC26-9583*
- *Query Management Facility: Introducing QMF, GC26-9576*
- *Query Management Facility: Messages and Codes, GC26-9580*
- *Query Management Facility: Reference, SC26-9577*
- *Query Management Facility: Using QMF, SC26-9578*

Ada/370

- *IBM Ada/370 Language Reference, SC09-1297*
- *IBM Ada/370 Programmer's Guide, SC09-1414*
- *IBM Ada/370 SQL Module Processor for DB2 Database Manager User's Guide, SC09-1450*

APL2®

- *APL2 Programming Guide, SH21-1072*
- *APL2 Programming: Language Reference, SH21-1061*
- *APL2 Programming: Using Structured Query Language (SQL), SH21-1057*

AS/400®

The following books are available at this Web site:
www.as400.ibm.com/infocenter

- *DB2 Universal Database for AS/400 Database Programming*
- *DB2 Universal Database for AS/400 Performance and Query Optimization*
- *DB2 Universal Database for AS/400 Distributed Data Management*
- *DB2 Universal Database for AS/400 Distributed Data Programming*
- *DB2 Universal Database for AS/400 SQL Programming Concepts*
- *DB2 Universal Database for AS/400 SQL Programming with Host Languages*
- *DB2 Universal Database for AS/400 SQL Reference*

BASIC

- *IBM BASIC/MVS Language Reference, GC26-4026*
- *IBM BASIC/MVS Programming Guide, SC26-4027*

BookManager® READ/MVS

- *BookManager READ/MVS V1R3: Installation Planning & Customization, SC38-2035*

SAA® AD/Cycle® C/370™

- *IBM SAA AD/Cycle C/370 Programming Guide, SC09-1841*
- *IBM SAA AD/Cycle C/370 Programming Guide for Language Environment/370, SC09-1840*
- *IBM SAA AD/Cycle C/370 User's Guide, SC09-1763*
- *SAA CPI C Reference, SC09-1308*

Character Data Representation Architecture

- *Character Data Representation Architecture Overview, GC09-2207*
- *Character Data Representation Architecture Reference and Registry, SC09-2190*

CICS/ESA

- *CICS/ESA Application Programming Guide, SC33-1169*
- *CICS External Interfaces Guide, SC33-1944*
- *CICS for MVS/ESA Application Programming Reference, SC33-1170*
- *CICS for MVS/ESA CICS-RACF Security Guide, SC33-1185*
- *CICS for MVS/ESA CICS-Supplied Transactions, SC33-1168*
- *CICS for MVS/ESA Customization Guide, SC33-1165*
- *CICS for MVS/ESA Data Areas, LY33-6083*
- *CICS for MVS/ESA Installation Guide, SC33-1163*
- *CICS for MVS/ESA Intercommunication Guide, SC33-1181*
- *CICS for MVS/ESA Messages and Codes, GC33-1177*
- *CICS for MVS/ESA Operations and Utilities Guide, SC33-1167*
- *CICS/ESA Performance Guide, SC33-1183*
- *CICS/ESA Problem Determination Guide, SC33-1176*
- *CICS for MVS/ESA Resource Definition Guide, SC33-1166*
- *CICS for MVS/ESA System Definition Guide, SC33-1164*
- *CICS for MVS/ESA System Programming Reference, GC33-1171*

CICS Transaction Server for OS/390

- *CICS Application Programming Guide, SC33-1687*
- *CICS External Interfaces Guide, SC33-1703*
- *CICS DB2 Guide, SC33-1939*
- *CICS Resource Definition Guide, SC33-1684*

IBM C/C++ for MVS/ESA™

- *IBM C/C++ for MVS/ESA Library Reference, SC09-1995*
- *IBM C/C++ for MVS/ESA Programming Guide, SC09-1994*

IBM COBOL

- *IBM COBOL Language Reference, SC26-4769*
- *IBM COBOL for MVS & VM Programming Guide, SC26-9049*

Conversion Guide

- *IMS-DB and DB2 Migration and Coexistence Guide, GH21-1083*

Cooperative Development Environment

- *CoOperative Development Environment/370: Debug Tool, SC09-1623*

DataPropagator NonRelational

- *DataPropagator NonRelational MVS/ESA Administration Guide, SH19-5036*
- *DataPropagator NonRelational MVS/ESA Reference, SH19-5039*

Data Facility Data Set Services

- *Data Facility Data Set Services: User's Guide and Reference, SC26-4388*

Database Design

- *DB2 Design and Development Guide by Gabrielle Wiorkowski and David Kull, Addison Wesley, ISBN 0-20158-049-7*
- *Handbook of Relational Database Design by C. Fleming and B. Von Halle, Addison Wesley, ISBN 0-20111-434-8*

DataHub®

- *IBM DataHub General Information, GC26-4874*

Data Refresher

- *Data Refresher Relational Extract Manager for MVS GI10-9927*

DB2 Connect®

- *DB2 Connect Enterprise Edition for OS/2 and Windows: Quick Beginnings, GC09-2953*
- *DB2 Connect Enterprise Edition for UNIX: Quick Beginnings, GC09-2952*
- *DB2 Connect Personal Edition Quick Beginnings, GC09-2967*
- *DB2 Connect User's Guide, SC09-2954*

DB2 Red Books

- *DB2 UDB Server for OS/390 Version 6 Technical Update, SG24-6108-00*

DB2 Server for VSE & VM

- *DB2 Server for VM: DBS Utility, SC09-2394*
- *DB2 Server for VSE: DBS Utility, SC09-2395*

DB2 Universal Database for UNIX, Windows, OS/2

- *DB2 UDB Administration Guide: Planning, SC09-2946*
- *DB2 UDB Administration Guide: Implementation, SC09-2944*
- *DB2 UDB Administration Guide: Performance, SC09-2945*
- *DB2 UDB Administrative API Reference, SC09-2947*
- *DB2 UDB Application Development Guide, Volume 3, SC09-2948*
- *DB2 UDB Application Development Guide, SC09-2949*
- *DB2 UDB CLI Guide and Reference, SC09-2950*
- *DB2 UDB Command Reference, SC09-2951*
- *DB2 UDB SQL Getting Started, SC09-2973*
- *DB2 UDB SQL Reference Volume 1, SC09-2974*
- *DB2 UDB SQL Reference Volume 2, SC09-2975*

Device Support Facilities

- *Device Support Facilities User's Guide and Reference, GC35-0033*

DFSMS

These books provide information about a variety of components of DFSMS, including DFSMS/MVS[®], DFSMSdfp[™], DFSMSdss[™], DFSMSshsm[™], and MVS/DFP[™].

- *DFSMS/MVS: Access Method Services for the Integrated Catalog, SC26-4906*
- *DFSMS/MVS: Access Method Services for VSAM Catalogs, SC26-4905*
- *DFSMS/MVS: Administration Reference for DFSMSdss, SC26-4929*
- *DFSMS/MVS: DFSMSshsm Managing Your Own Data, SH21-1077*
- *DFSMS/MVS: Diagnosis Reference for DFSMSdfp, LY27-9606*
- *DFSMS/MVS Storage Management Library: Implementing System-Managed Storage, SC26-3123*
- *DFSMS/MVS: Macro Instructions for Data Sets, SC26-4913*
- *DFSMS/MVS: Managing Catalogs, SC26-4914*
- *DFSMS/MVS: Program Management, SC26-4916*

- *DFSMS/MVS: Storage Administration Reference for DFSMSdfp*, SC26-4920
- *DFSMS/MVS: Using Advanced Services*, SC26-4921
- *DFSMS/MVS: Utilities*, SC26-4926
- *OS/390 DFSMS: Using Data Sets*, SC26-4749

DFSORT™

- *DFSORT Application Programming: Guide*, SC33-4035

Distributed Relational Database Architecture™

- *Data Stream and OPA Reference*, SC31-6806
- *IBM SQL Reference*, SC26-8416
- *Open Group Technical Standard*

The Open Group presently makes the following DRDA® books available through its Web site at: www.opengroup.org

- *DRDA Version 2 Vol. 1: Distributed Relational Database Architecture (DRDA)*
- *DRDA Version 2 Vol. 2: Formatted Data Object Content Architecture*
- *DRDA Version 2 Vol. 3: Distributed Data Management Architecture*

Domain Name System

- *DNS and BIND, Third Edition*, Paul Albitz and Cricket Liu, O'Reilly, ISBN 1-56592-512-2

Education

- *IBM Dictionary of Computing*, McGraw-Hill, ISBN 0-07031-489-6
- *1999 IBM All-in-One Education and Training Catalog*, GR23-8105

Enterprise System/9000® and Enterprise System/3090™

- *Enterprise System/9000 and Enterprise System/3090 Processor Resource/System Manager Planning Guide*, GA22-7123

High Level Assembler

- *High Level Assembler for MVS and VM and VSE Language Reference*, SC26-4940
- *High Level Assembler for MVS and VM and VSE Programmer's Guide*, SC26-4941

Parallel Sysplex® Library

- *OS/390 Parallel Sysplex Application Migration*, GC28-1863
- *System/390 MVS Sysplex Hardware and Software Migration*, GC28-1862
- *OS/390 Parallel Sysplex Overview: An Introduction to Data Sharing and Parallelism*, GC28-1860

- *OS/390 Parallel Sysplex Systems Management*, GC28-1861
- *OS/390 Parallel Sysplex Test Report*, GC28-1963
- *System/390 9672/9674 System Overview*, GA22-7148

ICSF/MVS

- *ICSF/MVS General Information*, GC23-0093

IMS

- *IMS Batch Terminal Simulator General Information*, GH20-5522
- *IMS Administration Guide: System*, SC26-9420
- *IMS Administration Guide: Transaction Manager*, SC26-9421
- *IMS Application Programming: Database Manager*, SC26-9422
- *IMS Application Programming: Design Guide*, SC26-9423
- *IMS Application Programming: Transaction Manager*, SC26-9425
- *IMS Command Reference*, SC26-9436
- *IMS Customization Guide*, SC26-9427
- *IMS Install Volume 1: Installation and Verification*, GC26-9429
- *IMS Install Volume 2: System Definition and Tailoring*, GC26-9430
- *IMS Messages and Codes*, GC27-1120
- *IMS Utilities Reference: System*, SC26-9441

ISPF

- *ISPF V4 Dialog Developer's Guide and Reference*, SC34-4486
- *ISPF V4 Messages and Codes*, SC34-4450
- *ISPF V4 Planning and Customizing*, SC34-4443
- *ISPF V4 User's Guide*, SC34-4484

Language Environment®

- *Debug Tool User's Guide and Reference*, SC09-2137

MQSeries

- *MQSeries Application Messaging Interface*, SC34-5604
- *MQSeries for OS/390 Concepts and Planning Guide*, GC34-5650
- *MQSeries for OS/390 System Setup Guide*, SC34-5651

National Language Support

- *IBM National Language Support Reference Manual Volume 2*, SE09-8002

NetView®

- *NetView Installation and Administration Guide, SC31-8043*
- *NetView User's Guide, SC31-8056*

Microsoft® ODBC

- *Microsoft ODBC 3.0 Software Development Kit and Programmer's Reference, Microsoft Press, ISBN 1-57231-516-4*

OS/390

- *OS/390 C/C++ Programming Guide, SC09-2362*
- *OS/390 C/C++ Run-Time Library Reference, SC28-1663*
- *OS/390 C/C++ User's Guide, SC09-2361*
- *OS/390 eNetwork Communications Server: IP Configuration, SC31-8513*
- *OS/390 Hardware Configuration Definition Planning, GC28-1750*
- *OS/390 Information Roadmap, GC28-1727*
- *OS/390 Introduction and Release Guide, GC28-1725*
- *OS/390 JES2 Initialization and Tuning Guide, SC28-1791*
- *OS/390 JES3 Initialization and Tuning Guide, SC28-1802*
- *OS/390 Language Environment for OS/390 & VM Concepts Guide, GC28-1945*
- *OS/390 Language Environment for OS/390 & VM Customization, SC28-1941*
- *OS/390 Language Environment for OS/390 & VM Debugging Guide, SC28-1942*
- *OS/390 Language Environment for OS/390 & VM Programming Guide, SC28-1939*
- *OS/390 Language Environment for OS/390 & VM Programming Reference, SC28-1940*
- *OS/390 MVS Diagnosis: Procedures, LY28-1082*
- *OS/390 MVS Diagnosis: Reference, SY28-1084*
- *OS/390 MVS Diagnosis: Tools and Service Aids, LY28-1085*
- *OS/390 MVS Initialization and Tuning Guide, SC28-1751*
- *OS/390 MVS Initialization and Tuning Reference, SC28-1752*
- *OS/390 MVS Installation Exits, SC28-1753*
- *OS/390 MVS JCL Reference, GC28-1757*
- *OS/390 MVS JCL User's Guide, GC28-1758*
- *OS/390 MVS Planning: Global Resource Serialization, GC28-1759*
- *OS/390 MVS Planning: Operations, GC28-1760*
- *OS/390 MVS Planning: Workload Management, GC28-1761*
- *OS/390 MVS Programming: Assembler Services Guide, GC28-1762*
- *OS/390 MVS Programming: Assembler Services Reference, GC28-1910*
- *OS/390 MVS Programming: Authorized Assembler Services Guide, GC28-1763*
- *OS/390 MVS Programming: Authorized Assembler Services Reference, Volumes 1-4, GC28-1764, GC28-1765, GC28-1766, GC28-1767*
- *OS/390 MVS Programming: Callable Services for High-Level Languages, GC28-1768*
- *OS/390 MVS Programming: Extended Addressability Guide, GC28-1769*
- *OS/390 MVS Programming: Sysplex Services Guide, GC28-1771*
- *OS/390 MVS Programming: Sysplex Services Reference, GC28-1772*
- *OS/390 MVS Programming: Workload Management Services, GC28-1773*
- *OS/390 MVS Routing and Descriptor Codes, GC28-1778*
- *OS/390 MVS Setting Up a Sysplex, GC28-1779*
- *OS/390 MVS System Codes, GC28-1780*
- *OS/390 MVS System Commands, GC28-1781*
- *OS/390 MVS System Messages Volume 1, GC28-1784*
- *OS/390 MVS System Messages Volume 2, GC28-1785*
- *OS/390 MVS System Messages Volume 3, GC28-1786*
- *OS/390 MVS System Messages Volume 4, GC28-1787*
- *OS/390 MVS System Messages Volume 5, GC28-1788*
- *OS/390 MVS Using the Subsystem Interface, SC28-1789*
- *OS/390 SecureWay Security Server Network Authentication and Privacy Service Administration, SC24-5896*
- *OS/390 Security Server External Security Interface (RACROUTE) Macro Reference, GC28-1922*
- *OS/390 Security Server (RACF) Auditor's Guide, SC28-1916*
- *OS/390 Security Server (RACF) Command Language Reference, SC28-1919*
- *OS/390 Security Server (RACF) General User's Guide, SC28-1917*
- *OS/390 Security Server (RACF) Introduction, GC28-1912*
- *OS/390 Security Server (RACF) Macros and Interfaces, SK2T-6700 (OS/390 Collection Kit), SK27-2180 (OS/390 Security Server Information Package)*
- *OS/390 Security Server (RACF) Security Administrator's Guide, SC28-1915*
- *OS/390 Security Server (RACF) System Programmer's Guide, SC28-1913*
- *OS/390 SMP/E Reference, SC28-1806*

- *OS/390 SMP/E User's Guide, SC28-1740*
- *OS/390 Support for Unicode: Using Conversion Services, SC33-7050*
- *OS/390 RMF User's Guide, SC28-1949*
- *OS/390 TSO/E CLISTS, SC28-1973*
- *OS/390 TSO/E Command Reference, SC28-1969*
- *OS/390 TSO/E Customization, SC28-1965*
- *OS/390 TSO/E Messages, GC28-1978*
- *OS/390 TSO/E Programming Guide, SC28-1970*
- *OS/390 TSO/E Programming Services, SC28-1971*
- *OS/390 TSO/E REXX Reference, SC28-1975*
- *OS/390 TSO/E User's Guide, SC28-1968*
- *OS/390 DCE Administration Guide, SC28-1584*
- *OS/390 DCE Introduction, GC28-1581*
- *OS/390 DCE Messages and Codes, SC28-1591*
- *OS/390 UNIX System Services Command Reference, SC28-1892*
- *OS/390 UNIX System Services Messages and Codes, SC28-1908*
- *OS/390 UNIX System Services Planning, SC28-1890*
- *OS/390 UNIX System Services User's Guide, SC28-1891*
- *OS/390 UNIX System Services Programming: Assembler Callable Services Reference, SC28-1899*
- *OS/390 V2R10.0 IBM CS IP Configuration Reference, SC31-8726*
- *Program Directory for OS/390 V2 R8/R9/R10 support for Unicode, GI10-9760*

IBM Enterprise PL/I for z/OS and OS/390

- *IBM Enterprise PL/I for z/OS and OS/390 Language Reference, SC26-9476*
- *IBM Enterprise PL/I for z/OS and OS/390 Programming Guide, SC26-9473*

OS PL/I

- *OS PL/I Programming Language Reference, SC26-4308*
- *OS PL/I Programming Guide, SC26-4307*

Prolog

- *IBM SAA AD/Cycle Prolog/MVS & VM Programmer's Guide, SH19-6892*

RAMAC[®] and Enterprise Storage Server[™]

- *IBM RAMAC Virtual Array, SG24-4951*
- *RAMAC Virtual Array: Implementing Peer-to-Peer Remote Copy, SG24-5338*
- *Enterprise Storage Server Introduction and Planning, GC26-7294*

Remote Recovery Data Facility

- *Remote Recovery Data Facility Program Description and Operations, LY37-3710*

Storage Management

- *DFSMS/MVS Storage Management Library: Implementing System-Managed Storage, SC26-3123*
- *MVS/ESA Storage Management Library: Leading a Storage Administration Group, SC26-3126*
- *MVS/ESA Storage Management Library: Managing Data, SC26-3124*
- *MVS/ESA Storage Management Library: Managing Storage Groups, SC26-3125*
- *MVS Storage Management Library: Storage Management Subsystem Migration Planning Guide, SC26-4659*

System/370[™] and System/390[®]

- *ESA/370 Principles of Operation, SA22-7200*
- *ESA/390 Principles of Operation, SA22-7201*
- *System/390 MVS Sysplex Hardware and Software Migration, GC28-1210*

System Network Architecture (SNA)

- *SNA Formats, GA27-3136*
- *SNA LU 6.2 Peer Protocols Reference, SC31-6808*
- *SNA Transaction Programmer's Reference Manual for LU Type 6.2, GC30-3084*
- *SNA/Management Services Alert Implementation Guide, GC31-6809*

TCP/IP

- *IBM TCP/IP for MVS: Customization & Administration Guide, SC31-7134*
- *IBM TCP/IP for MVS: Diagnosis Guide, LY43-0105*
- *IBM TCP/IP for MVS: Messages and Codes, SC31-7132*
- *IBM TCP/IP for MVS: Planning and Migration Guide, SC31-7189*

VS COBOL II

- *VS COBOL II Application Programming Guide for MVS and CMS, SC26-4045*
- *VS COBOL II Application Programming: Language Reference, GC26-4047*
- *VS COBOL II Installation and Customization for MVS, SC26-4048*

VS Fortran

- *VS Fortran Version 2: Language and Library Reference, SC26-4221*

- *VS Fortran Version 2: Programming Guide for CMS and MVS, SC26-4222*

VTAM®

- *Planning for NetView, NCP, and VTAM, SC31-8063*
- *VTAM for MVS/ESA Diagnosis, LY43-0069*
- *VTAM for MVS/ESA Messages and Codes, SC31-6546*
- *VTAM for MVS/ESA Network Implementation Guide, SC31-6548*
- *VTAM for MVS/ESA Operation, SC31-6549*
- *VTAM for MVS/ESA Programming, SC31-6550*
- *VTAM for MVS/ESA Programming for LU 6.2, SC31-6551*
- *VTAM for MVS/ESA Resource Definition Reference, SC31-6552*

Index

A

- APIs, ODBC 38
- application servers
 - Net.Data 27
 - WebSphere 27
- applications
 - enhancements for e-business 25
 - Net Search Extender 60
 - Web 60
- Automatic Restart Manager (ARM) 20
- auxiliary storage 130
- availability enhancements 13

B

- bookmarks, CLI 48

C

- CANCEL THREAD command
 - use without rollback processing 20
- catalog tables
 - columns, new and changed 122
 - indexes, new and changed 123
 - tables, new 121
- CHECK DATA utility
 - description 100
- CHECK INDEX utility
 - description 100
- CHECK LOB utility
 - description 101
- commands
 - CANCEL THREAD 20
 - changes in Version 7 97
 - SET SYSPARM 19
- COMMIT and ROLLBACK in stored procedures 45
- CONNECT statement 37
- connection pooling 48
- connectivity improvements 45
- constraints 9
- COPY utility
 - description 101
- COPYTOCOPY utility 15
 - description 99
- coupling facility storage
 - structure size changes 21
- cursor
 - scrollable 31

D

- DASD requirement 130
- data communications devices 130
- data definitions, generating using Control Center 53
- Data Management Tools 64
- data set templates 54
- data sets and device types 130

- data sharing
 - enhancements 20
 - group attachment enhancements 20
 - improvements in failure recovery of group buffer pool 22
 - notification of incomplete units of recovery 22
 - provision of more robust failure recovery 22
 - restart light 20
 - structure size changes, persistent 21
- database access using DB2 Connect 27
- DAYOFWEEK_ISO 38
- DB2 Administration Tool 53
- DB2 Connect 27
- DB2 Estimator 59
- DB2 family compatibility, SQL enhancements 30
- DB2 Installer 56
- DB2 Management Clients Package 52
- DB2 Stored Procedure Builder 55
- DB2 UDB Control Center
 - managing DB2 for OS/390 and z/OS from a workstation 52
- DB2 Visual Explain 57
- DB2 Warehouse Manager 62
- DB2 XML Extender for OS/390 and z/OS
 - description 29
 - graphical support 30
 - using SQL to perform searches 30
- DBADM authority
 - CREATE ALIAS command 9
 - CREATE VIEW command 9
- DELETE or UPDATE with self-referencing subselect 35
- DISPLAY DDF
 - DETAIL keyword 22
- distributed applications
 - coordinating updates 28
 - global transaction support 47
 - performance improvement 35, 36
- DSN_STATEMNT_TABLE
 - changed columns 126
- dynamic allocation of data sets 7

E

- e-business
 - DB2 XML Extender for OS/390 and z/OS 29
 - Net Search Extender 60
 - Net.Data 61
 - scalability 13
 - system parameters, changing online 19
- encrypted change password 47
- encrypted password 46
- encrypted userid and password 46
- EXEC SQL utility control statement 99
- EXPLAIN table changes
 - DSN_STATEMNT_TABLE 125
 - PLAN_TABLE 125

F

- fallback
 - frozen objects
 - associated with a fallback to Version 5 74
 - associated with a fallback to Version 6 81
 - preparation
 - Version 5 74
 - Version 6 81
 - release incompatibilities
 - associated with a fallback to Version 5 76
 - associated with a fallback to Version 6 82
- fast implicit close 36
- features of DB2 for OS/390 and z/OS 49
- FETCH FIRST *n* ROWS ONLY clause 35
- frozen objects
 - associated with a fallback to Version 5 74
 - associated with a fallback to Version 6 81
- functions
 - column 117
 - new in Version 7 117
 - scalar 117
 - table 117
- functions, ODBC 38

G

- global transaction support for distributed applications 47

H

- hardware requirements
 - auxiliary storage 129
 - data communications devices 130
 - function-dependent 130

I

- identity column values with wrapping 34
- IFCID (instrumentation facility component identifier)
 - new and changed 127
 - user-defined function 127
- IN-list index access
 - parallelism 19
- Information Center 55
- insert operations, preformatting for improved performance 22
- installation panel changes 94
- ISO
 - DAYOFWEEK_ISO 38
 - WEEK_ISO 38

J

- Java
 - Java applets and Javascript with Net.Data 61
 - reports with Java query capability 49
 - stored procedures 55
- Java stored procedure
 - support for interpreted 39

- Java user-defined function
 - support for 39
- JDBC and SQLJ
 - enhancements 39
- join transformation
 - subqueries 17

K

- Kerberos security 46

L

- LISTDEF utility control statement 8, 99
- LOAD RESUME, online 17
- LOAD utility
 - description 102
 - enhancements 40
- log read request, retry 20

M

- MERGECOPY utility
 - description 102
- migration considerations
 - Version 5 to Version 7 69
 - Version 6 to Version 7 78
- MODIFY RECOVERY utility
 - description 103
- MODIFY STATISTICS utility
 - description 99
- monitoring
 - server-elapsed time for remote requests 47
- MQSeries user-defined functions 48
- multisite updates 28
- multitier environment 25

N

- Net Search Extender 60
- Net.Data
 - use for secure Web applications 61
- NOBACKOUT option of CANCEL THREAD
 - command 20
- notices, legal 141

O

- object collections with the Control Center 54
- object lists 54
- ODBC enhancements 38
- optical storage 130
- optional features of DB2 for OS/390 and z/OS 49
- OPTIONS utility control statement 99
- ORDER BY clause
 - sort changes 18

P

- parallel LOAD with multiple inputs 15

- parallelism
 - IN-list index access support 19
- pattern-matching characters in utility lists 7
- performance considerations
 - scrollable cursor 33
- performance, estimating 59
- PLAN_TABLE
 - columns, new and changed 126
 - format 125
- Precompiler Services, component of DB2 for OS/390 and z/OS 40
- program requirements
 - function-dependent 131
 - operating systems 131
 - optional 133

Q

- QMF (Query Management Facility) 49
- QMF for Windows 50
- QMF High Performance Option (HPO) 50
- QUIESCE utility
 - description 103

R

- REBUILD INDEX utility
 - description 103
- RECOVER utility
 - description 103
- release coexistence 93
- release dependency markers
 - associated with a fallback to Version 5 74
 - associated with a fallback to Version 6 81
- release incompatibilities
 - migration from Version 5 83
 - migration from Version 6 89
- remote request response times 47
- REORG INDEX utility
 - description 104
- REORG TABLESPACE utility
 - description 105
- REORG utility, online 17
- REPORT utility
 - description 105
- requirements for DB2 for OS/390 and z/OS
 - hardware 129
 - program 131
 - virtual storage 131
- restart light
 - use in data sharing 20
- RUNSTATS utility
 - description 105

S

- sample jobs, changes 96
- scalability improvements 13
- scrollable cursor 31
 - performance considerations 33

- security
 - encrypted change password 47
 - encrypted password 46
 - encrypted userid and password 46
 - Web applications 61
 - Windows Kerberos 46
 - workstation clients 46
- SELECT statement 37
 - ORDER BY 37
- self-referencing subselect on UPDATE and DELETE 35
- server-elapsed time monitoring 47
- SQL (Structured Query Language)
 - CONNECT statement 37
 - enhancements for DB2 family compatibility 30
 - processing queries 17
 - SELECT statement 37
 - support for COMMIT and ROLLBACK in stored procedures 45
 - use with DB2 XML Extender for OS/390 and z/OS 30
- SQL procedures
 - IN and OUT parameters 37
- SQL scalar function 37
- SQL statement coprocessor 40
- SQL statements
 - changes in Version 7 107
- sqlint32 38
- statistics history, comprehensive 7
- storage
 - requirements
 - real 129
 - virtual 131
- stored procedure support for COMMIT and ROLLBACK 45
- structure size changes, persistent 21
- subquery
 - join transformation 17
- system parameters, changing online 19

T

- tables
 - UNION and UNION ALL operators, using 30
- TEMPLATE control statement 8
- TEMPLATE utility control statement 99
- templates, data set 54
- three-tier architecture 25
- transaction manager support in DB2 28
- transaction pooling 48

U

- Unicode support 42
- UNION and UNION ALL operators 30
- UNLOAD utility 14
 - description 100
- UPDATE or DELETE with self-referencing subselect 35
- utilities
 - changes in Version 7 99

utilities (*continued*)

- CHECK DATA 100
- CHECK INDEX 100
- CHECK LOB 101
- Control Center, using with 53
- COPY 101
- COPYTOCOPY 15, 99
- dynamic allocation of data sets 7
- LOAD 15, 102
- LOAD RESUME, online 17
- MERGECOPY 102
- MODIFY RECOVERY 103
- MODIFY STATISTICS 99
- procedures using the Control Center 54
- QUIESCE 103
- REBUILD INDEX 103
- RECOVER 103
- REORG INDEX 104
- REORG TABLESPACE 105
- REORG, online 17
- REPORT 105
- RUNSTATS 105
- support for DB2 13
- UNLOAD 14, 100
 - using pattern-matching characters 7
- utility control statement
 - EXEC SQL 99
 - LISTDEF 99
 - OPTIONS 99
 - TEMPLATE 99

W

- warehouse management 62
- Web
 - accessing books 55
 - application servers 27
 - Net.Data 27
 - WebSphere 27
 - applications 60
 - Net Search Extender 60
 - publishing reports using QMF 49
- WEEK_ISO 38
- Windows Kerberos security 46
- wrapping of identity column values 34

X

- XML documents
 - searching 29
 - storing 29
 - transmitting 29

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

DB2 Universal Database for OS/390 and z/OS
Release Planning Guide
Version 7

Publication No. SC26-9943-03

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

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Overall satisfaction	<input type="checkbox"/>				

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

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Accurate	<input type="checkbox"/>				
Complete	<input type="checkbox"/>				
Easy to find	<input type="checkbox"/>				
Easy to understand	<input type="checkbox"/>				
Well organized	<input type="checkbox"/>				
Applicable to your tasks	<input type="checkbox"/>				

Please tell us how we can improve this book:

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

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

Name

Address

Company or Organization

Phone No.



Fold and Tape

Please do not staple

Fold and Tape



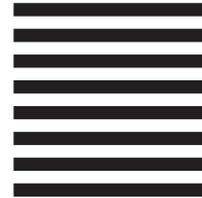
NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

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

POSTAGE WILL BE PAID BY ADDRESSEE

International Business Machines Corporation
H150/090
555 Bailey Avenue
San Jose, CA 95141-9989
U. S. A.



Fold and Tape

Please do not staple

Fold and Tape



Program Number: 5675-DB2

Printed in U.S.A.

SC26-9943-03

