

Db2 13 for z/OS

What's New?



Notes

Before using this information and the product it supports, be sure to read the general information under "Notices" at the end of this information.

Subsequent editions of this PDF will not be delivered in IBM Publications Center. Always download the latest edition from [IBM Documentation](#).

2022-06-15 edition

This edition applies to Db2[®] 13 for z/OS[®] (product number 5698-DB2[®]), Db2 13 for z/OS Value Unit Edition (product number 5698-DBV), 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.

Specific changes are indicated by a vertical bar to the left of a change. A vertical bar to the left of a figure caption indicates that the figure has changed. Editorial changes that have no technical significance are not noted.

© **Copyright International Business Machines Corporation 2022.**

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

Contents

About this information.....	V
Terminology and citations.....	vi
How to send your comments about Db2 for z/OS documentation.....	vi
Part 1. What's new in Db2 13.....	1
Chapter 1. Overview of what's new in Db2 13.....	3
Chapter 2. Db2 13 function levels.....	15
Function level 501 (Db2 13 installation or migration - May 2022).....	15
Function level 500 (for migrating to Db2 13 - May 2022).....	19
Function level 100 (for migrating to Db2 13 - May 2022).....	24
Part 2. Changes to plan for in Db2 13.....	33
Chapter 3. Incompatible changes in Db2 13.....	35
Chapter 4. Storage changes in Db2 13.....	37
Chapter 5. Command changes in Db2 13.....	39
Chapter 6. SQL changes in Db2 13.....	41
Chapter 7. Utility changes in Db2 13.....	45
Chapter 8. Catalog changes in Db2 13.....	47
Chapter 9. IFCID changes.....	51
Chapter 10. Subsystem parameter changes in Db2 13.....	55
Chapter 11. The extended 10-byte RBA and LRSN in Db2 13.....	59
Chapter 12. Function that Db2 13 no longer supports.....	63
Chapter 13. Deprecated function in Db2 13.....	65
Part 3. Adopting new capabilities in Db2 13 continuous delivery.....	69
Chapter 14. Function levels and related levels in Db2 13.....	71
Chapter 15. Determining the Db2 code level, catalog level, and function level.....	75
Chapter 16. Testing Db2 function level activation.....	79
Chapter 17. Identifying applications that are incompatible with catalog updates.....	81
Chapter 18. Activating Db2 13 function levels.....	83
Chapter 19. Activating Db2 function levels by using z/OSMF.....	87

Job DSNTIJOZ: bring Db2 subsystem parameter changes online.....	88
Chapter 20. Updating Db2 initialization parameters for function level activation.....	89
Chapter 21. Controlling Db2 application compatibility.....	91
Chapter 22. Responding to problems after function level activation.....	93
Chapter 23. How Db2 function levels are documented.....	95
Information resources for Db2 for z/OS and related products.....	97
Notices.....	99
Programming interface information.....	100
Trademarks.....	100
Terms and conditions for product documentation.....	100
Privacy policy considerations.....	101
Glossary.....	103
Index.....	105

About this information

This information provides an executive overview of new function in Db2 13 for z/OS. The topics in this information provide a framework for describing new function in Db2 for z/OS. New functions are categorized according to user benefits such as information on demand, availability, and performance.

In addition, this information summarizes changes that were introduced in this version for Db2 commands, Db2 utilities, SQL statements, the Db2 catalog, Db2 performance monitoring, and instrumentation facility component identifiers (IFCIDs).

Throughout this information, "Db2" means "Db2 13 for z/OS". References to other Db2 products use complete names or specific abbreviations.

Important: To find the most up to date content for Db2 13 for z/OS, always use [IBM® Documentation](#) or download the latest PDF file from [PDF format manuals for Db2 13 for z/OS \(Db2 for z/OS in IBM Documentation\)](#).

Most documentation topics for Db2 13 for z/OS assume that the highest available function level is activated and that your applications are running with the highest available application compatibility level, with the following exceptions:

- The following documentation sections describe the Db2 13 migration process and how to activate new capabilities in function levels:
 - [Migrating to Db2 13 \(Db2 Installation and Migration\)](#)
 - [Part 1, “What's new in Db2 13,” on page 1](#)
 - [Part 3, “Adopting new capabilities in Db2 13 continuous delivery,” on page 69](#)
- [FL 500 A](#) label like this one usually marks documentation changed for function level 500 or higher, with a link to the description of the function level that introduces the change in Db2 13. For more information, see [Chapter 23, “How Db2 function levels are documented,” on page 95](#).

The availability of new function in Db2 13 depends on the type of enhancement, the activated function level, and the application compatibility levels of the applications. For a list of all available function levels in Db2 13, see [Chapter 2, “Db2 13 function levels,” on page 15](#).

Function level 100

Db2 starts at function level 100 (V13R1M100) during migration to Db2 13, and fallback and coexistence with Db2 12 in data sharing remain possible. Many new capabilities in Db2 13 remain unavailable. For more information, see [“Function level 100 \(for migrating to Db2 13 - May 2022\)” on page 24](#).

Function level 500

Activating function level 500 (V13R1M500) prevents coexistence with and fallback to Db2 12. Function level 500 is also the first opportunity for applications to use many of the new capabilities in Db2 13. However, new capabilities that depend on Db2 13 catalog changes remain unavailable. For more information, see [“Function level 500 \(for migrating to Db2 13 - May 2022\)” on page 19](#).

Function level 501

Function level 501 (V13R1M501) is the first opportunity after migration to Db2 13 for applications to use new features and capabilities that depend on catalog changes in Db2 13. For more information, see [“Function level 501 \(Db2 13 installation or migration - May 2022\)” on page 15](#).

Some virtual storage and optimization enhancements take effect in function level 100. Optimization enhancements become available after full prepare of the SQL statements, depending on the statement type:

- For static SQL statements, after bind or rebind of the package.
- For non-stabilized dynamic SQL statements, immediately, unless the statement is in the dynamic statement cache.

- For stabilized dynamic SQL statements, after invalidation, free, or changed application compatibility level.

Who should read this information

This information is written primarily for people who are evaluating and planning for Db2 for z/OS.

Terminology and citations

When referring to a Db2 product other than Db2 for z/OS, this information 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.

IBM OMEGAMON® for Db2 Performance Expert on z/OS

Refers to any of the following products:

- IBM IBM OMEGAMON for Db2 Performance Expert on z/OS
- IBM Db2 Performance Monitor on z/OS
- IBM Db2 Performance Expert for Multiplatforms and Workgroups
- IBM Db2 Buffer Pool Analyzer for z/OS

C, C++, and C language

Represent the C or C++ programming language.

CICS®

Represents CICS Transaction Server for z/OS.

IMS

Represents the IMS Database Manager or IMS Transaction Manager.

MVS™

Represents the MVS element of the z/OS operating system, which is equivalent to the Base Control Program (BCP) component of the z/OS operating system.

RACF®

Represents the functions that are provided by the RACF component of the z/OS Security Server.

How to send your comments about Db2 for z/OS documentation

Your feedback helps IBM to provide quality documentation.

Send any comments about Db2 for z/OS and related product documentation by email to db2zinfo@us.ibm.com.

To help us respond to your comment, include the following information in your email:

- The product name and version
- The address (URL) of the page, for comments about online documentation
- The book name and publication date, for comments about PDF manuals
- The topic or section title
- The specific text that you are commenting about and your comment

Related concepts

[About this information \(Db2 for z/OS in IBM Documentation\)](#)

Related reference

[PDF format manuals for Db2 13 for z/OS \(Db2 for z/OS in IBM Documentation\)](#)

Part 1. What's new in Db2 13

Db2 13 for z/OS brings leading-edge innovation to reinforce Db2 for z/OS as a foundation for enterprise computing within the hybrid cloud world.

Db2 13 for z/OS delivers significant advances in all critical enterprise database success factors: availability, scalability, performance, security, and ease of use. The value of Db2 13 is also maximized by synergy with surrounding tools and technology. The latest advances in IBM Z hardware, new and improved development tooling, and add-on capabilities such as AI infusion and IBM Db2 Analytics Accelerator query acceleration, complement the new capabilities in Db2 13 to provide the most complete, intelligent and intuitive database environment yet.

Continuous delivery in Db2 13

Db2 13 evolves the *continuous delivery* of new capabilities and enhancements in a single service stream as soon as they are ready, which was introduced in Db2 12. The result is that you can benefit from new capabilities and enhancements without waiting for an entire new release. *Function levels* enable you to control the timing of the activation and adoption of new features, with the option to continue to apply corrective and preventative service without adopting new feature function.

New for Db2 13, function level 501 is available immediately at GA, and it activates any new capabilities that require catalog changes in Db2 13. For more information, and a list of all available function levels in Db2 13, see [Chapter 2, “Db2 13 function levels,” on page 15](#).

Tip: For the most current information, view documentation for Db2 continuous delivery and function levels in [IBM Documentation](#), or download the latest PDF edition from [PDF format manuals for Db2 13 for z/OS \(Db2 for z/OS in IBM Documentation\)](#).

Db2 for z/OS News from the Lab blog: See the [Db2 for z/OS News from the Lab blog](#) for the latest news about new capabilities and enhancements in Db2 for z/OS continuous delivery, from the IBM experts who design, build, test, and support Db2

[Subscribe today!](#)

Related information

[IBM Db2 13 for z/OS and More \(IBM Redbooks\)](#)

Chapter 1. Overview of what's new in Db2 13

Db2 13 introduces new capabilities and enhancements for simplified migration, SQL enhancements, applications management, IBM Z hardware synergy, availability and scalability, performance, and more!

Many of the new capabilities in Db2 13 take effect when you activate function level 500 or higher in Db2 13. The function level for each new capability is identified here, and you can also find the same descriptions organized by function level in [Chapter 2, “Db2 13 function levels,”](#) on page 15.

Contents

[Simplified migration](#)

[SQL enhancements and applications management](#)

[IBM Z hardware synergy](#)

[Availability](#)

[Scalability](#)

[Performance](#)

[Insight, instrumentation, and serviceability](#)

[Db2 Utilities](#)

Simplified migration to Db2 13

Availability of Db2 13 new function

Migrations to Db2 13 for z/OS use a single phase. As in migrations to Db2 12, you use function levels to control the availability of most new function in Db2 13. For a complete list of available function levels, see [Chapter 2, “Db2 13 function levels,”](#) on page 15.

At general availability, Db2 13 includes the following function levels:

Function level 100

Db2 starts at function level 100 (V13R1M100) during migration to Db2 13, and fallback and coexistence with Db2 12 in data sharing remain possible. Many new capabilities in Db2 13 remain unavailable. For more information, see [“Function level 100 \(for migrating to Db2 13 - May 2022\)”](#) on page 24.

Function level 500

Activating function level 500 (V13R1M500) prevents coexistence with and fallback to Db2 12. Function level 500 is also the first opportunity for applications to use many of the new capabilities in Db2 13. However, new capabilities that depend on Db2 13 catalog changes remain unavailable. For more information, see [“Function level 500 \(for migrating to Db2 13 - May 2022\)”](#) on page 19.

Function level 501

Function level 501 (V13R1M501) is the first opportunity after migration to Db2 13 for applications to use new features and capabilities that depend on catalog changes in Db2 13. For more information, see [“Function level 501 \(Db2 13 installation or migration - May 2022\)”](#) on page 15.

For more information, see [Activating Db2 13 function levels 500 and 501 \(Db2 Installation and Migration\)](#).

Simplified catalog migration

When migrating to Db2 13, you use the CATMAINT utility invoked by job DSNTIJTC to control the timing of the migration as in earlier releases, but it does not make any structural changes to the Db2 catalog. After the V13R1M100 CATMAINT completes, Db2 is at catalog level and function level

V13R1M100. This is a change from Db2 12, which started with catalog level V12R1M500 at function level 100.

The first changes to catalog objects are delayed until you tailor the Db2 catalog for activation of function level 501.

For more information, see [Migrating your Db2 subsystem to Db2 13 \(Db2 Installation and Migration\)](#).

Subsystem parameter simplification

Function level 100 introduces changes to the default values for various subsystem parameters to match current best practices. It also removes a number of obsolete subsystem parameters. For a list of these changes, see [Chapter 10, “Subsystem parameter changes in Db2 13,”](#) on page 55.

SQL enhancements and applications management in Db2 13

SQL Data Insights

Function level 500 delivers SQL Data Insights, an integrated solution that brings deep learning AI capabilities into Db2. SQL Data Insights uses unsupervised neural networks to generate a specialized vector-embedding model called database embedding, which can be referenced through SQL queries called "cognitive intelligence" queries.

The SQL Data Insights user interface is an optional feature available at no additional charge with Db2 13, which provides the user interface for training models and exploring data insights. Db2 provides the in-database infrastructure for training and model table (vector table) management. Db2 also provides three new built-in cognitive functions to speed up query execution.

For more information, see [Running AI queries with SQL Data Insights](#).

Increased flexibility for package ownership

Starting at function level 500, you can specify the type of owner for a plan, package, or service, or the type of package owner for an SQL PL routine. The owner can be a role or an authorization ID. The default owner is a role in a trusted context that is defined with the role as object owner and qualifier attributes, otherwise the default owner is an authorization ID.

For more information, see the `PACKAGE OWNER` clause of [CREATE PROCEDURE \(SQL - native\) \(Db2 SQL\)](#) and the `OWNERTYPE` option of the [OWNER bind option \(Db2 Commands\)](#).

Column names longer than 30 bytes

Function level 100 extends the maximum length of a column name from 30 bytes of EBCDIC, up to 128 bytes with limited support for using the longer column names. The longer column names can be used when the `TABLE_COL_NAME_EXPANSION` subsystem parameter setting is ON. Although you can now define a column with a name up to 128 bytes, column names with a length greater than 30 bytes of EBCDIC might be truncated on a character boundary. Column names returned in an SQLDA contain 30 bytes at most. APIs that do not use the SQLDA to obtain a column name might return complete column names.

For more information, see [Column names longer than 30 bytes \(Db2 SQL\)](#) and [TABLE_COL_NAME_EXPANSION in macro DSN6SPRM \(Db2 Installation and Migration\)](#).

REBIND simplification

Function level 100 introduces reduced storage usage during BIND/REBIND for queries that involve many tables. This enhancement optimizes Db2 13 and improves its performance without changing how you configure, monitor, and use Db2.

IBM Z hardware synergy in Db2 13

Db2 13 introduces the following new capabilities that take advantage of the synergy between IBM Z hardware and Db2 for z/OS.

Expanded SORTL usage with learning from execution (IBM z15™)

Function level 100 introduces expanded SORTL usage based on machine learning on the amount of storage and the number of records being sorted, when run on IBM z15 or later processors. This

enhancement optimizes Db2 13 and improves its performance without changing how you configure, monitor, and use Db2.

Db2 support for z/OS continuous compliance

Customers are looking for solutions that provide evidence that they can trust the security of z/OS systems. z/OS 2.5 introduces new SMF type 1154 records that provide evidence of security compliance. Participating components and products can collect and write compliance data to their associated SMF 1154 subtype records. Function level 100 adds the capability to collect evidence on Db2 subsystems' compliance by writing SMF 1154 subtype 81 records. For more information, see [Db2 evidence for z/OS continuous compliance \(Managing Security\)](#) and [What is new in z/OS \(V2R4 - V2R5\)](#).

IBM z16 group buffer pool (GBP) residency time

Starting in function level 100, two new statistics are added to relevant group buffer pool statistics storage areas:

- The average time a data area resides in a storage class before it is reclaimed.
- The average time a directory entry resides in a storage class before it is reclaimed.

These new statistics are supported only if Db2 is installed on z/OS 2.4 or later with necessary PTFs applied. Db2 and the coupling facility must be running on IBM z16 or later processors, and the coupling facility must be at CFLEVEL 25 or higher. For more information, see [CFLEVEL and operating system level coexistence \(z/OS MVS Setting Up a Sysplex\)](#).

You can access these metrics with the IFCID record trace and the -DISPLAY GROUPBUFFERPOOL command. For more information, see "DSNB820I: Average residency times" in [DSNB750I \(Db2 Messages\)](#).

Availability in Db2 13

Db2 13 introduces the following new capabilities that improve the online availability of your data and database applications on Db2 for z/OS.

DBAT availability improvements

Function level 100 introduces changes to Db2 13 DBAT termination processing to support the following objectives:

- Reduction of the overall frequency and number of DBAT terminations.
- Reduction of the number of concurrent DBAT terminations that are caused by a short-term increase in DBAT usage.

This enhancement optimizes Db2 13 and improves its performance without changing how you configure, monitor, and use Db2.

Online conversion of tables from growth-based (PBG) to range-based (PBR) partitions

Function level 500 introduces the capability to convert the partitioning scheme of a table with growth-based partitions (in a PBG table space) to use range-based partitions (in a PBR table space). The conversion can be completed as an online change with minimal impact to your applications.

PBG and PBR universal table spaces (UTS) are the strategic table space types for tables in Db2 for z/OS. PBG table spaces are the default UTS type, and they are well-suited for small to medium-sized tables. However, if an existing table in a PBG table space grows too large, performance degradation or data and index management issues might arise. Consider converting from PBG to PBR when that occurs.

To complete the conversion, you issue an ALTER TABLE statement with the new ALTER PARTITIONING TO PARTITION BY RANGE clause and run the REORG TABLESPACE utility to materialize the pending change. The table space for the table is converted to PBR with relative page numbers (RPN).

For more information, see [Converting tables from growth-based to range-based partitions \(Db2 Administration Guide\)](#) and "ALTER PARTITIONING TO PARTITION BY RANGE" in [ALTER TABLE \(Db2 SQL\)](#).

Increased control for applications over how long to wait for a lock timeout

Function level 500 introduces the CURRENT LOCK TIMEOUT special register and the SET CURRENT LOCK TIMEOUT SQL statement to allow the lock timeout value to be set at the application level. So, you can set a lock timeout interval that suits the needs of a specific application, or even an individual SQL statement. Doing so minimizes application lock contention and simplifies portability of applications to Db2, without the need to assign the application to a separate Db2 subsystem.

The value of the CURRENT LOCK TIMEOUT special register overrides the value of the IRLMRWT subsystem parameter. It applies to certain processes related to locking, like the claim or drain of an object and cached dynamic statement quiescing.

For more information, see [CURRENT LOCK TIMEOUT \(Db2 SQL\)](#) and [SET CURRENT LOCK TIMEOUT \(Db2 SQL\)](#).

You can limit use of CURRENT LOCK TIMEOUT by setting the new SPREG_LOCK_TIMEOUT_MAX subsystem parameter. For more information, see [LOCK TIMEOUT MAX \(SPREG_LOCK_TIMEOUT_MAX subsystem parameter\) \(Db2 Installation and Migration\)](#).

You can also use Db2 profile tables to specify an assignment for the CURRENT LOCK TIMEOUT special register, for both remote and local threads. See [Setting special registers by using profile tables \(Db2 Administration Guide\)](#).

Allow applications to specify a deadlock resolution priority

Function level 501 introduces the SYSIBMADM.DEADLOCK_RESOLUTION_PRIORITY built-in global variable to allow an application to specify a priority to use when resolving a deadlock situation with other threads. When an application sets and uses this built-in global variable (by using a SET *assignment-statement* SQL statement), the Db2 subsystem uses that value as a relative weighting factor to resolve deadlock situations with other threads.

For more information, see [DEADLOCK_RESOLUTION_PRIORITY \(Db2 SQL\)](#).

You can also use Db2 profile tables to specify values for the new SYSIBMADM.DEADLOCK_RESOLUTION_PRIORITY built-in global variable for both remote and local applications. See [Setting built-in global variables by using profile tables \(Db2 Administration Guide\)](#).

Profile table enhancements for application environment settings

Db2 13 introduces the capability to use system profiles for local applications in certain situations. Previously, the initial values for special registers and system built-in global variables can be specified in the Db2 profile tables, but they are used only for initialization with distributed threads. The new Db2 profile table support for local applications requires Db2 to be started with the DDF subsystem parameter set to AUTO or COMMAND. See [DDF STARTUP OPTION field \(DDF subsystem parameter\) \(Db2 Installation and Migration\)](#).

You can use Db2 profiles tables for both local and remote applications in the following situations:

Db2 function level	New Db2 profile table support
Function level 500 or higher	<ul style="list-style-type: none">You can specify assignments to the new CURRENT LOCK TIMEOUT special register. For more information, see CURRENT LOCK TIMEOUT (Db2 SQL) and Setting special registers by using profile tables (Db2 Administration Guide).You can specify a new RELEASE_PACKAGE keyword with a COMMIT attribute to change the release bind option for a package. See Overriding the RELEASE(DEALLOCATE) option for packages by using profile tables (Db2 Performance).
Function level 501 or higher	<ul style="list-style-type: none">You can specify values for the new SYSIBMADM.DEADLOCK_RESOLUTION_PRIORITY built-in global variable. See DEADLOCK_RESOLUTION_PRIORITY (Db2 SQL) and Setting built-in global variables by using profile tables (Db2 Administration Guide).

Ability to delete an active log data set from the BSDS while Db2 is running

Function level 500 introduces the new REMOVELOG option for the -SET LOG command to support online removal of an active log data set from the BSDS, eliminating the need to stop Db2 to accomplish the task by using the offline utility DSNJU003. The -SET LOG REMOVELOG command deletes the specified log from the BSDS if it is not in use or mark the log REMOVAL PENDING if it is in use.

To provide monitoring of the current active log status for log data sets with REMOVAL PENDING status, function level 500 also introduces the DETAIL option for the -DISPLAY LOG command. It shows information regarding REMOVAL PENDING status for local active log data sets. The output from the utility DSNJU004 also shows the REMOVAL PENDING status where applicable.

For more information, see [Deleting an active log data set from the BSDS with the -SET LOG command \(Db2 Administration Guide\)](#).

Relative page numbers for new PBR table spaces

Starting in function level 100, the default value of the PAGESET_PAGENUM subsystem parameter is changed to RELATIVE. The PAGESET_PAGENUM subsystem parameter specifies the default value that Db2 uses when you omit the PAGENUM option in CREATE TABLESPACE or CREATE TABLE statement for a partition-by-range (PBR) table space. That is, it specifies whether Db2 creates the table space and associated partitioned indexes to use relative page numbers (RPN) or absolute page numbers (APN) across partitions. RPN is the strategic direction for PBR table spaces in Db2. If you accept the new default and create all new PBR table spaces with relative page numbers, you can avoid costly future conversions. Converting from absolute to relative pages numbers always requires a REORG of the entire table space.

See [PAGE SET PAGE NUMBERING field \(PAGESET_PAGENUM subsystem parameter\) \(Db2 Installation and Migration\)](#).

Improved concurrency for altering tables for DATA CAPTURE

Function level 500 introduces a concurrency improvement for ALTER TABLE statements that change the DATA CAPTURE attribute of tables. With this enhancement, Db2 no longer waits for other statements that depend on the altered table to commit. As a result, the DATA CAPTURE alteration can now succeed even when concurrent statements are running continually against the table.

Earlier Db2 releases quiesce the following objects that depend on the altered table as part of the DATA CAPTURE alteration:

- Static packages
- Cached dynamic SQL statements

Because the DATA CAPTURE alteration waited for applications that depended on the altered table to commit, continuous concurrent activity on the table might cause the ALTER TABLE statements to fail.

The new DATA CAPTURE attribute now takes effect immediately when the processing completes, even before the ALTER statement commits. As a result, concurrent statements on the same Db2 member might write out different log formats in the same transaction. For more information, see [Altering a table to capture changed data \(Db2 Administration Guide\)](#).

CREATE TABLESPACE uses MAXPARTITIONS 254 by default

At application compatibility level V13R1M500 or higher, CREATE TABLESPACE statements use MAXPARTITIONS 254 by default.

When MAXPARTITIONS 256 is explicitly specified, the default DSSIZE varies from 4 G to 32 G depending on the page size. However, starting with application compatibility level V12R1M504, when MAXPARTITIONS is not explicitly specified, Db2 12 use MAXPARTITIONS 256 by default, but the default DSSIZE is always 4 G regardless of the page size.

This apparent inconsistency avoided a risk of failure for existing statements, where the default data set size might be greater than 4 G depending on the page size. The statements might fail with SQLCODE -904 with reason code 00D70008 if the data sets for the table space are not associated with a DFSMS data class that is specified with extended format and extended addressability.

With MAXPARTITIONS 254 as the default, the result is now consistent regardless of whether MAXPARTITIONS is explicitly specified. The calculated default DSSIZE is always 4 G.

See the MAXPARTITIONS and DSSIZE descriptions in [CREATE TABLESPACE \(Db2 SQL\)](#).

Scalability in Db2 13

Db2 13 introduces the following capabilities to improve the scalability of your data and database applications on Db2 for z/OS.

Reduced ECSA storage for IFI buffers

Db2 13 reduces the use of ECSA storage for IFI buffers from a maximum of 50 MB to a fixed 8 MB.

Function level 100 reduces the use of ECSA storage for IFI buffers to a maximum of 25 MB. Then, after function level 500 is first activated, it is further reduced to 8 MB. The storage behavior that is introduced in function level 500 continues even if you later activate function level 100*.

To compensate for the reduction in ECSA storage, you must set aside an extra 50 MB for HVCOMMON and 25 MB for private storage. You can reduce the ECSA storage after function level 500 is activated and Db2 starts using the new storage pools. When Db2 uses the new storage pools, the use of ECSA for the retrieval of IFI records noticeably decreases. You can monitor use of the new storage pools by starting the statistics trace to collect IFCID 0225. Then, you can check the SHARED / COMMON storage summary report in the formatted IDCID 225 SMF trace record.

For more information about ECSA storage requirements, see [Calculating the storage requirement for the extended common service area \(Db2 Installation and Migration\)](#).

Reduced ECSA storage use for distributed data facility (DDF) processing

Function level 100 reduces the amount of ECSA storage that is used for processing DDF threads to be equivalent to processing local threads. The previous recommendation was an extra 2 KB per DDF thread. For more information, see [Calculating the storage requirement for the extended common service area \(Db2 Installation and Migration\)](#).

Reduced agent local below-the-bar (BTB) storage

Starting in function level 100, Db2 supports a greater number of concurrent threads, by using above-the-bar (ATB) agent-local storage for statement text and attribute strings for dynamic SQL statements. In earlier releases, Db2 kept a copy of dynamic SQL statement text and attribute strings in agent local below-the-bar (BTB) storage while the statement is being prepared and executed.

For any specific thread, multiple dynamic SQL statements can be executing depending on the nesting level. The maximum length of an SQL statement is 2 MB, but much more storage can be allocated and the consumption of BTB storage could prevent the number of threads from scaling.

This enhancement optimizes Db2 13 and improves its performance without changing how you configure, monitor, and use Db2.

Improved storage monitoring and contraction

Function level 100 introduces the following enhancements to provide storage constraint relief:

- When below-the-bar Db2 storage consumption exceeds 64-percent threshold, Db2 automatically begins contraction of private storage pools.
- When extended common service area (ECSA) storage consumption exceeds the 85-percent threshold, Db2 automatically begins contraction of storage pools that are allocated in the ECSA.

In both cases, the storage contraction stops after storage consumption drops below the threshold.

This enhancement optimizes Db2 13 and improves its performance without changing how you configure, monitor, and use Db2.

Improved Db2 installation and migration process for customizing the amount of private storage for the IRLM lock control structure

In the MAX STORAGE FOR LOCKS field on installation panel DSNTIPJ, you can specify the maximum amount of private storage above the 2 GB bar for the IRLM lock control structure. In earlier Db2

releases, you can specify a value of only up to 102400 megabytes. Starting in Db2 13, you can specify a value of up to 16384 petabytes.

For more information, see [MAX STORAGE FOR LOCKS field \(Db2 Installation and Migration\)](#) and [MAX LOCK STORAGE UNIT field \(Db2 Installation and Migration\)](#).

Real-time statistics scalability

As data volumes become larger, the widths of some columns in the real-time statistics tables are not large enough to accommodate larger values. In addition, during high volume processing, lock escalation might occur on the real-time statistics history table spaces. Lock escalation can negatively affect concurrency and performance.

Starting after function level 501 is activated, the following changes to the real-time statistics tables and table spaces are introduced to provide greater capacity and concurrency:

- In real-time statistics tables SYSIBM.SYSTABLESPACESTATS and SYSIBM.SYSINDEXSPACESTATS, and their associated history tables SYSIBM.SYSTABSPACESTATS_H and SYSIBM.SYSIXSPACESTATS_H, some column data types are BIGINT instead of INTEGER, or INTEGER instead of SMALLINT.
- Lock escalation is disabled on the following table spaces: DSNDB06.SYSTSTSS and DSNDB06.SYSTSISS for the RTS tables; and DSNDB06.SYSTSTSH and DSNDB06.SYSTSISH for the RTS history tables

For more information, see [SYSTABLESPACESTATS catalog table \(Db2 SQL\)](#) and [SYSINDEXSPACESTATS catalog table \(Db2 SQL\)](#).

This enhancement optimizes Db2 13 and improves its performance without changing how you configure, monitor, and use Db2.

More efficient cleanup for above-the-bar storage

Function level 100 introduces improvements to how Db2 manages and frees above-the-bar storage, especially to reduce the disruptive impact of issuing excessive IARV64 REQUEST(DISCARDDATA) service requests.

Db2 13 no longer issues the IARV64 REQUEST(DICARDDATA) request during thread deallocation or at certain intervals of COMMIT, and enhanced storage management is no longer controlled by the REALSTORAGE_MANAGEMENT subsystem parameter, which is also removed. In Db2 13, the storage is returned to the memory object. A system-level timer drives contraction for the memory object to release unused frames back to z/OS. Also, Db2 13 periodically checks the available free frames before the LPAR starts to page (by using the z/OS calculations for available free frames and LO threshold). If this value becomes lower than 5 times the z/OS calculated OK threshold, the memory object contraction is triggered.

SPT01 and SYSLGRNX table spaces are converted to DSSIZE 256 GB

Starting in function level 500, the first time that the REORG TABLESPACE utility runs for the following directory objects, it converts the DSSIZE to 256 GB.

- DSNDB01.SPT01 to resolve issues that are related to the removal of the SPT01_INLINE_LENGTH subsystem parameter by APAR PH24358 in Db2 12.
- DSNDB01.SYSLGRNX in anticipation of future growth in this table for increasing workloads and conversions of non-UTS table space to UTS.

The conversion is automatic and does not require any special utility syntax. For more information, see [“Function level 500 \(for migrating to Db2 13 - May 2022\)” on page 19](#).

Enhanced dynamic allocation processing

In function level 100 and z/OS 2.5 or later, dynamic allocation processing supports system work blocks (SWBs) for data sets that are in 64-bit storage. This new dynamic allocation function helps reduce below-the-bar storage usage for address spaces that allocate large numbers of data sets.

To enable this feature, complete one of the following actions:

- Update the ALLOCxx parmlib member to set the SYSTEM SWBSTORAGE value to ATB. The default value is SWA, which indicates that SWBs reside in 24-bit storage or 31-bit storage. ATB indicates that SWBs are allowed to reside in 64-bit storage.
- Issue system command SETALLOC SYSTEM,SWBSTORAGE=ATB.

It is best to update the ALLOCxx parmlib member because the change remains effective across IPLs. If the SETALLOC command is used to enable SYSTEM SWBSTORAGE, you must restart Db2 for the change to take effect.

Previously, the CLIST calculation for data set storage size used 5 KB per open data set. With the new dynamic allocation function, the storage that is required per open data set is reduced to 4 KB. You must adjust the calculation for data set storage size. For more information, see [Calculating data set control block storage \(Db2 Installation and Migration\)](#).

Performance in Db2 13

Sort optimization

Function level 100 introduces enhanced sort optimizations, which were previously introduced for ORDER BY and GROUP BY processing. It applies them to improve the performance of certain operations, such as the following processing enhancements:

- Machine-generated code support for DECFLOAT processing.
- Support for the following enhancements for GROUPING SET, multiple DISTINCT, and PERCENTILE processing:
 - Machine-generated code support.
 - Sort processing can use its own work file.
 - A check for ordered data in the first iteration of a sort.
 - Larger sort trees can be used.
- SUBSTR support for the LISTAGG built-in function, if the start position and length for SUBSTR is a constant.
- Support for avoiding rereading of a single work file, if the sort work file is used, and if IBM Watson[®] Machine Learning for z/OS is enabled.
- Support for reducing the length of long VARCHAR keys, if the last key in an ORDER BY list is a VARCHAR over 100 bytes, and if IBM Watson Machine Learning for z/OS is enabled.

This enhancement optimizes Db2 13 and improves its performance without changing how you configure, monitor, and use Db2.

Improved locking for INSERT to partition-by-growth (PBG) table spaces

Function level 100 introduces retry logic for INSERT operations. An extra attempt is made to obtain a partition lock on a PBG table space after a failed first attempt, thereby increasing the success rate of INSERT operations.

Before this enhancement, only a single attempt was made to obtain a lock on the target partition. If the attempt failed, the target partition was skipped, and the next partition was evaluated. This process would continue until the INSERT operation either successfully obtained a partition lock or it finished searching all existing partitions without obtaining a partition lock.

In most cases, the duration of partition lock contention is short; however, because the INSERT operation did not make another attempt to obtain a lock on a partition after the first failed attempt, the INSERT operation terminated unnecessarily. In many cases, making an extra attempt to obtain a partition lock results in the successful completion of an INSERT operation that otherwise would fail.

This enhancement optimizes Db2 13 and improves its performance without changing how you configure, monitor, and use Db2.

Index look-aside optimization

Function level 100 introduces index look-aside optimizations, to improve performance for insert, update, and delete operations. This enhancement optimizes Db2 13 and improves its performance without changing how you configure, monitor, and use Db2.

Improved performance when using external security

Function level 100 introduces the following enhancements to improve performance for Db2 environments that use external security:

- Db2 caches plan authorization checks that the access control authorization exit (ACAE) routine uses. Previously, successful authorization checks on the EXECUTE privilege for plans were not cached if those checks were completed by the ACAE routine. This enhancement provides consistent behavior in plan authorization cache behavior regardless of whether security is managed with Db2 facilities or with the ACAE.

To enable plan authorization caching when the ACAE routine is being used, the AUTHEXIT_CACHEREFRESH subsystem parameter must be set to ALL and the z/OS release must be 2.5 or later. Db2 caches the results of authorization checks on the EXECUTE privilege for plans if a profile in the RACF class MDSNPN permits access to the plan. Db2 does not cache the results if access is allowed due to administrative authority, such as the DATAACCESS or SYSADM authorities.

For more information, see [Caching of EXECUTE on plans, packages, and routines \(Managing Security\)](#).

- Db2 is enhanced to cache more authorization IDs per plan. For more information, see [Caching authorization IDs for plans \(Managing Security\)](#).
- The AUTHCACH subsystem parameter is removed to simplify plan authorization cache management. Use the CACHESIZE bind option on the BIND PLAN subcommand to specify the size of the authorization cache for that plan. The default value is 4K.
- If the AUTHEXIT_CACHEREFRESH subsystem parameter is set to ALL, the global authentication cache takes the timestamp into consideration for user IDs that were authenticated by using credentials other than multi-factor authentication (MFA). For more information, see [Global authentication cache \(Managing Security\)](#).
- When you specify a key label for data set encryption, the specified key label cannot refer to an archived key for decryption operations only. Key labels can be specified by using the ENCRYPTION_KEYLABEL subsystem parameter or any of the following SQL statements:
 - ALTER STOGROUP
 - ALTER TABLE
 - CREATE STOGROUP
 - CREATE TABLE

If the specified key label refers to a decryption-only archived key, the key label specification fails and returns an error message. For more information on decryption-only archived keys, see [ICSF: Limit archived keys to decrypt operations only](#).

Fast index traversal (FTB) support for larger index keys

Function level 500 extends FTB support to unique indexes with a key size for the ordering columns up to 128 bytes and nonunique indexes with a key size up to 120 bytes. For more information, see [Fast index traversal \(Db2 Performance\)](#).

Db2 Utilities in Db2 13

Db2 13 introduces the following new capabilities and enhancements to Db2 Utilities Suite for z/OS:

Collection of real-time and historical information about utility execution

To improve utility management, function level 501 introduces the ability to collect real-time and historical information about utility execution. After you activate utility history collection by setting the UTILITY_HISTORY subsystem parameter to UTILITY, information about utilities is added to the SYSIBM.SYSUTILITIES catalog table. One row is inserted into the SYSUTILITIES table at the start

of each utility execution. Then, information in the row is updated as the utility progresses, and final information is updated in the row when the utility execution finishes.

For more information, see [Monitoring utility history \(Db2 Utilities\)](#), [UTILITY HISTORY \(UTILITY_HISTORY subsystem parameter\) \(Db2 Installation and Migration\)](#), and [SYSUTILITIES catalog table \(Db2 SQL\)](#).

Page sampling for inline statistics

Beginning in function level 500, the REORG TABLESPACE and LOAD utilities can now use page sampling when they gather inline statistics. Page sampling has the potential to reduce both CPU time and elapsed time. In earlier Db2 releases, the RUNSTATS utility can use page sampling, but inline statistics that are gathered by other utilities can use row sampling only. To use page sampling for inline statistics with REORG TABLESPACE or LOAD, specify the TABLESAMPLE SYSTEM option or ensure that the STATPGSAMP subsystem parameter is set to SYSTEM (the default) or YES. In function level 500, STATPGSAMP is extended to apply to inline statistics. For more information, see the TABLESAMPLE SYSTEM option description in [Syntax and options of the LOAD control statement \(Db2 Utilities\)](#) and [Syntax and options of the REORG TABLESPACE control statement \(Db2 Utilities\)](#).

Enhanced space-level recovery with the RECOVER utility

Starting in function level 100, the RECOVER utility supports space-level recovery (where DSNUM ALL is specified or is the default), even if the image copies were created at the partition or piece level for the following objects:

- Universal table spaces (UTS).
- Index spaces or indexes for a UTS.
- XML UTS with a base table that resides in a UTS.
- Auxiliary index spaces or indexes for an XML UTS with a base table that resides in a UTS.
- LOB table spaces with a base table that resides in a UTS.
- Auxiliary index spaces or indexes for LOB table spaces with a base table that resides in a UTS.

Space-level recovery of these supported object types is processed as partition-level or piece-level recoveries in the RESTORE phase. Db2 catalog and directory objects that meet the criteria above are supported by this enhancement. When the list contains a mixture of supported and unsupported object types, recovery behavior for the unsupported object types is the same as Db2 12 or earlier releases.

When the RECOVER utility is invoked (with DSNUM ALL specified or as the default) in Db2 12 or earlier releases, for space-level recovery of table spaces, index spaces, or indexes, an error message is issued if the image copies were created at the partition or piece level. The DSNU512I (DATASET LEVEL RECOVERY IS REQUIRED) error message indicates that recovery cannot be done at the space level. Recovery must instead be requested at the partition or piece level. Objects with these errors are not recovered and the RECOVER ends with RC8 indicating errors were encountered.

For more information, see [Recovering a data set or partition \(Db2 Utilities\)](#).

REPAIR utility WRITELOG for decompression dictionaries

Function level 100 introduces the capability to write a decompression dictionary log record up to the maximum log record size supported by Db2. This capability can be used after you run an application or utility that builds a new dictionary without writing the old one to the log. So, it is useful for replication products that would otherwise require a refresh of the replication target. After the decompression dictionary is successfully written, the REPAIR utility issues message DSNU3335I with the location of the log record. Applications can then use this information to insert a SYSIBM.SYSCOPY record.

For more information, see the option descriptions for TYPE X'4002 and SUBTYPE 'X'000A' in [Syntax and options of the REPAIR control statement \(Db2 Utilities\)](#), and [DSNU3335I \(Db2 Messages\)](#).

Change REORG INDEX SHRLEVEL REFERENCE or CHANGE so the NOSYSUT1 keyword is the default

Starting at function level 500, the NOSYSUT1 keyword is the default for the REORG INDEX utility when specified with the SHRLEVEL REFERENCE or CHANGE keywords. So, the utility avoids use of a work data set, which improves performance and allows REORG INDEX to use parallel subtasks to unload

and to build index keys. The default value of the REORG_INDEX_NOSYSUT1 subsystem parameter is also changed from NO to YES, and YES is now the only option. So, this subsystem parameter no longer influences the behavior of REORG INDEX.

For more information, see [Syntax and options of the REORG INDEX control statement \(Db2 Utilities\)](#) and [REORG TS NOPAD DEFAULT \(REORG_TS_NOPAD_DEFAULT subsystem parameter\) \(Db2 Installation and Migration\)](#).

Insight, instrumentation, and serviceability in Db2 13

Improved default statistics collection granularity

Starting in function level 100, that default value of the STATIME_MAIN subsystem parameter is changed to 10 seconds. The STATIME_MAIN subsystem parameter specifies the time interval in seconds, for collection of interval-driven statistics that are not collected at the interval that is specified by the STATIME subsystem parameter. With the default statistics interval set at 60 seconds in earlier releases, Db2 database administrators and system programmers cannot identify true workload peaks by using Db2 statistics for subsystem level performance tuning and planning. Similarly, a slowdown of 5 - 15 seconds is difficult to diagnose with statistics collected at a 60-second interval.

See [MAIN STATS TIME field \(STATIME_MAIN subsystem parameter\) \(Db2 Installation and Migration\)](#)

Index split instrumentation enhancements

Function level 500 introduces IFCID 0396 to provide detailed information about index splits. When data is inserted into a base table, the corresponding indexes are modified accordingly. As a result, performance of SQL insert operations can be impacted during the *index split* process, where synchronous I/O is required under data sharing and the group buffer pool GBP is dependent on the related index page sets.

The existing IFCID 0359 records already contain information index split events. However, the information that is recorded is not detailed enough to identify the cause of performance issues. IFCID 0359 is also disabled by default, and it can miss capturing some abnormal index split situations.

IFCID 0396 is always enabled by default under statistics trace class 3 and performance trace class 6. An IFCID 396 record is generated when an index split is considered an abnormal split process, such as when the total elapsed time is greater than 1 second. The generated IFCID 0396 record contains information such as the DBID, PSID, member ID, URID, page number, and more. The information is helpful for both customers and IBM Support to identify the root cause of INSERT performance issues.

For more information, see the IFCID 0396 descriptions in [Chapter 9, “IFCID changes,” on page 51](#) and in the [Trace field descriptions \(Db2 Performance\)](#).

Starting after the catalog level V13R1M501 update, the following RTS columns in the SYSIBM.SYSINDEXSPACESTATS catalog table are populated. They record and aggregate general index split information since last table reorganization, index rebuild, or load replace:

Column name	Data type	Description
REORGTOTALSPLITS	INTEGER	The number of index splits since last reorganization or rebuild.
REORGSPLITTIME	BIGINT	Aggregated elapsed time for all index splits since last reorganization or rebuild.
REORGEXCSPLITS	INTEGER	The number of abnormal index splits (such as elapsed times greater than 1 second) since last reorganization or rebuild.

Db2 starts populating these RTS columns as soon as the catalog level V13R1M501 update completes, even before function level 501 is activated.

Accounting information on the longest wait times for common suspension types

When Db2 transactions take a long time, it is important to determine:

- Where the transaction time is spent.
- Whether the problem is many short suspensions or a few long suspensions.
- Which resources the suspensions are for.

Before this enhancement, detailed performance traces were required to find this information. This enhancement simplifies the diagnosis process by providing information in Db2 accounting records for the longest wait time for a number of common suspension types. The following suspension types are included:

- IRLM lock suspensions
- Db2 internal latch suspensions
- Waits for Db2 synchronous or asynchronous I/O
- Waits for Db2 service tasks
- Waits for page latches

For more information, see [Chapter 9, “IFCID changes,”](#) on page 51 and the DSNWMSGs file.

Partition range support in IFCID 306 for users of replication applications

Function level 100 introduces the capability for applications that collect IFCID 306 trace records for the log read process to request filtering on a range of partitions. For more information, see the new WQLSFLG flag and WQLSDBPP mapping in [Qualification fields for READS requests \(Db2 Performance\)](#).

EDITPROC support in IFCID 306 for users of nonproxy mode replication applications

Function level 100 introduces the support for any user of IFCID 306 in non-proxy mode to use EDITPROC support as an on-request function. For more information, see the new values X'04', X'05', X'06', and X'07' for the 1-byte WQALLOPT field in [Qualification fields for READS requests \(Db2 Performance\)](#).

Related concepts

[Changes to plan for in Db2 13](#)

Use this information when you are planning migration to Db2 13 and for planning to adopt new capabilities that Db2 13 introduces.

Related reference

[Db2 13 function levels](#)

New Db2 capabilities and enhancements are continuously delivered in a single maintenance stream as the code becomes ready. You can activate the new capabilities in a data sharing group or Db2 subsystem after a function level is delivered. A *function level* corresponds to a single PTF that enables the activation of a specific set of enhancements that shipped in previous prerequisite or co-requisite PTFs. The activation of a function level results in the activation of all lower function levels.

Chapter 2. Db2 13 function levels

New Db2 capabilities and enhancements are continuously delivered in a single maintenance stream as the code becomes ready. You can activate the new capabilities in a data sharing group or Db2 subsystem after a function level is delivered. A *function level* corresponds to a single PTF that enables the activation of a specific set of enhancements that shipped in previous prerequisite or co-requisite PTFs. The activation of a function level results in the activation of all lower function levels.

Db2 for z/OS News from the Lab blog: See the [Db2 for z/OS News from the Lab blog](#) for the latest news about new capabilities and enhancements in Db2 for z/OS continuous delivery, from the IBM experts who design, build, test, and support Db2

[Subscribe today!](#)

About function levels in Db2 13

A *function level* enables a particular set of new Db2 capabilities and enhancements that were previously delivered in the single continuous stream of Db2 code. It includes code that supports new capabilities, defect fixes, and preventive service items. Before you can use the new capabilities of a function level, you must activate the function level, or a higher function level. Activation of a function level implies activation of the capabilities that are introduced by all lower function levels.

For more information about function levels, and how to activate them in Db2 13, see [Part 3, “Adopting new capabilities in Db2 13 continuous delivery,”](#) on page 69.

Available function levels

The following function levels are available in Db2 13. They are listed in release order, beginning with the highest available function level.

Function level 501 (Db2 13 installation or migration - May 2022)

Function level 501 (V13R1M501) is the first opportunity after migration to Db2 13 for applications to use new features and capabilities that depend on catalog changes in Db2 13.

Contents

[“New capabilities in function level 501”](#) on page 15

[“How to activate function level 501”](#) on page 17

Enabling APAR:	None
Full identifier:	V13R1M501
Catalog level required:	V13R1M501
Product identifier (PRDID):	DSN13015
Incompatible changes:	None

New capabilities in function level 501

Function level 501 activates the following new capabilities in Db2 13.

Allow applications to specify a deadlock resolution priority

Function level 501 introduces the SYSIBMADM.DEADLOCK_RESOLUTION_PRIORITY built-in global variable to allow an application to specify a priority to use when resolving a deadlock situation with other threads. When an application sets and uses this built-in global variable (by using a SET

assignment-statement SQL statement), the Db2 subsystem uses that value as a relative weighting factor to resolve deadlock situations with other threads.

For more information, see [DEADLOCK_RESOLUTION_PRIORITY \(Db2 SQL\)](#).

You can also use Db2 profile tables to specify values for the new SYSIBMADM.DEADLOCK_RESOLUTION_PRIORITY built-in global variable for both remote and local applications. See [Setting built-in global variables by using profile tables \(Db2 Administration Guide\)](#).

Profile table enhancements for application environment settings

Db2 13 introduces the capability to use system profiles for local applications in certain situations. Previously, the initial values for special registers and system built-in global variables can be specified in the Db2 profile tables, but they are used only for initialization with distributed threads. The new Db2 profile table support for local applications requires Db2 to be started with the DDF subsystem parameter set to AUTO or COMMAND. See [DDF STARTUP OPTION field \(DDF subsystem parameter\) \(Db2 Installation and Migration\)](#).

Starting in function level 501, Db2 profile tables can be used for both local and remote applications in the following situations:

- You can specify values for the new SYSIBMADM.DEADLOCK_RESOLUTION_PRIORITY built-in global variable. See [DEADLOCK_RESOLUTION_PRIORITY \(Db2 SQL\)](#) and [Setting built-in global variables by using profile tables \(Db2 Administration Guide\)](#).

Real-time statistics scalability

As data volumes become larger, the widths of some columns in the real-time statistics tables are not large enough to accommodate larger values. In addition, during high volume processing, lock escalation might occur on the real-time statistics history table spaces. Lock escalation can negatively affect concurrency and performance.

Starting after function level 501 is activated, the following changes to the real-time statistics tables and table spaces are introduced to provide greater capacity and concurrency:

- In real-time statistics tables SYSIBM.SYSTABLESPACESTATS and SYSIBM.SYSINDEXSPACESTATS, and their associated history tables SYSIBM.SYSTABSPACESTATS_H and SYSIBM.SYSIXSPACESTATS_H, some column data types are BIGINT instead of INTEGER, or INTEGER instead of SMALLINT.
- Lock escalation is disabled on the following table spaces: DSNDB06.SYSTSTSS and DSNDB06.SYSTSISS for the RTS tables; and DSNDB06.SYSTSTSH and DSNDB06.SYSTSISH for the RTS history tables

For more information, see [SYSTABLESPACESTATS catalog table \(Db2 SQL\)](#) and [SYSINDEXSPACESTATS catalog table \(Db2 SQL\)](#).

This enhancement optimizes Db2 13 and improves its performance without changing how you configure, monitor, and use Db2.

Collection of real-time and historical information about utility execution

To improve utility management, function level 501 introduces the ability to collect real-time and historical information about utility execution. After you activate utility history collection by setting the UTILITY_HISTORY subsystem parameter to UTILITY, information about utilities is added to the SYSIBM.SYSUTILITIES catalog table. One row is inserted into the SYSUTILITIES table at the start of each utility execution. Then, information in the row is updated as the utility progresses, and final information is updated in the row when the utility execution finishes.

For more information, see [Monitoring utility history \(Db2 Utilities\)](#), [UTILITY HISTORY \(UTILITY_HISTORY subsystem parameter\) \(Db2 Installation and Migration\)](#), and [SYSUTILITIES catalog table \(Db2 SQL\)](#).

Real-time statistics support for index splits

Catalog level V13R1M501 introduces three new real-time statistics (RTS) table columns in the Db2 catalog to provide detailed information about index splits.

The following RTS table fields are added in the SYSIBM.SYSINDEXSPACESTATS catalog table to record and aggregate general index split information since the last table reorganization, index rebuild or load replace:

Column name	Data type	Description
REORGTOTALSPLITS	INTEGER	The number of index splits since last reorganization or rebuild.
REORGSPLITTIME	BIGINT	Aggregated-elapsed time for all index splits since last reorganization or rebuild.
REORGEXSPLITS	INTEGER	The number of abnormal index splits (such as elapsed times greater than 1 second) since last reorganization or rebuild.

Db2 starts populating these RTS columns as soon as the catalog level V13R1M501 update completes, even before function level 501 is activated.

V13R1M501 application compatibility

Most new SQL capabilities become available only to applications that use the equivalent application compatibility (APPLCOMPAT) level or higher. For a list, see [Chapter 6, “SQL changes in Db2 13,” on page 41](#).

For more information about application compatibility levels, see [Chapter 21, “Controlling Db2 application compatibility,” on page 91](#).

How to activate function level 501

In new Db2 13 installations, function level 501 is already activated. Use this procedure if you are migrating from Db2 12.

Before you begin

Before you activate function level 500 or higher, complete the following prerequisite tasks:

1. In Db2 12, identify and resolve incompatible changes and activate function level 510. You can run the pre-migration job DSNTIJPE in Db2 12 to identify the incompatible changes. For more information, see [Verify Db2 13 premigration activities and activate function level 510 in Db2 12 \(Db2 Installation and Migration\)](#).

2. Verify that every member was restarted with the fallback SPE applied in Db2 12.

Important: Inactive members that never started in Db2 12 with the fallback SPE applied cannot start after function level 500 is activated in Db2 13.

3. Migrate the Db2 subsystem or data sharing group to Db2 13, as described in [Migrating your Db2 subsystem to Db2 13 \(Db2 Installation and Migration\)](#) or [Migrating an existing data sharing group to Db2 13 \(Db2 Installation and Migration\)](#).

4. Verify that you no longer need to fall back to Db2 12.

Important: After function level 500 is activated in Db2 13, coexistence and fallback to Db2 12 are no longer possible. You can activate function level 100* to disable new capabilities in Db2 13, but function level 100* does not support coexistence or fallback .

5. In data sharing, ensure that the group has no active Db2 12 members. See [Migrating subsequent members of a group to Db2 13 \(Db2 Installation and Migration\)](#).

Procedure

To activate function level 501, complete the following steps:

1. Tailor the CATMAINT and function level activation jobs by running the installation CLIST:

- a. In panel DSNTIPA1, specify `INSTALL TYPE ===> ACTIVATE`. Then, specify the name of the output member from the previous function level activation (or migration) in the `INPUT MEMBER` field, and specify a new member name in the `OUTPUT MEMBER` field.
 - b. In panel DSNTIP00, specify the current function level and `TARGET FUNCTION LEVEL ===> V13R1M501`. The CLIST uses this value when it tailors the `ACTIVATE` command in the DSNTIJAF job and the CATMAINT utility control statement in the DSNTIJTC job.
 - c. Proceed through the remaining CLIST panels, and wait for the CLIST to tailor the jobs for the activation process. The output data set contains the tailored jobs for the activation process.
2. If the current function level is V13R1M100, activate function level 500 by running the generated DSNTIJA0 job, or by issuing the following `ACTIVATE` command:

```
-ACTIVATE FUNCTION LEVEL (V13R1M500)
```

3. Ensure that no incompatible applications can interfere with the catalog update. For more information, see [Chapter 17, “Identifying applications that are incompatible with catalog updates,”](#) on page 81.
4. Update the catalog and verify the changes for function level 501 by completing the following steps:
 - a. Run the tailored DSNTIJTC job, or run the CATMAINT utility with `LEVEL V13R1M501`, to update the catalog to the appropriate catalog level. If multiple catalog updates are required, the CATMAINT job processes each update in sequential order. If a later update in the sequence fails, the previous successful updates do not roll back, and the catalog level remains at the highest level reached. If that occurs, you can correct the reason for the failure and resubmit the same CATMAINT job.
 - b. Run the generated DSNTIJX2 job to run the CHECK INDEX utility for Db2 catalog and directory indexes for new objects that are created in Db2 13.
5. Optionally, check that Db2 is ready for function level activation by issuing the following `ACTIVATE` command with the TEST option:

```
-ACTIVATE FUNCTION LEVEL (V13R1M501) TEST
```

Db2 issues message DSNU757I to indicate the results. For more information, see [Chapter 16, “Testing Db2 function level activation,”](#) on page 79.

6. Run the tailored DSNTIJAF job, or issue the following `ACTIVATE` command:

```
-ACTIVATE FUNCTION LEVEL (V13R1M501)
```

7. If you are ready for applications to use new capabilities in this function level, rebind them at the corresponding application compatibility level. For more information, see [Chapter 21, “Controlling Db2 application compatibility,”](#) on page 91.

Optionally, when you are ready for all applications to use the new capabilities of the target function level, you can run the following jobs:

- a. Run DSNTIJUZ to modify the subsystem parameter module with the APPLCOMPAT value that was specified on panel DSNTIP00.
- b. Run DSNTIJOZ job to issue SET SYSPARM command to bring the APPLCOMPAT subsystem parameter changes online.
- c. Run DSNTIJUA job to modify the Db2 data-only application defaults module with the SQLLEVEL value that was specified on panel DSNITP00.

Related tasks

[Activating Db2 13 function levels 500 and 501 \(Db2 Installation and Migration\)](#)

[Activating Db2 13 function levels](#)

You control the activation and adoption of new features in Db2 13 by activating function levels and specifying the application compatibility level. You can also continue to apply corrective and preventative service without adopting new feature function.

Related reference

[-ACTIVATE \(Db2\) \(Db2 Commands\)](#)

Function level 500 (for migrating to Db2 13 - May 2022)

Activating function level 500 (V13R1M500) prevents coexistence with and fallback to Db2 12. Function level 500 is also the first opportunity for applications to use many of the new capabilities in Db2 13. However, new capabilities that depend on Db2 13 catalog changes remain unavailable.

Contents

[“New capabilities in function level 500” on page 19](#)

[“How to activate function level 500” on page 23](#)

Enabling APAR:	None
Full identifier:	V13R1M500
Catalog level required:	V13R1M100
Product identifier (PRDID):	DSN13015
Incompatible changes:	None

New capabilities in function level 500

Function level 500 activates the following new capabilities in Db2 13.

Increased flexibility for package ownership

Starting at function level 500, you can specify the type of owner for a plan, package, or service, or the type of package owner for an SQL PL routine. The owner can be a role or an authorization ID. The default owner is a role in a trusted context that is defined with the role as object owner and qualifier attributes, otherwise the default owner is an authorization ID.

For more information, see the `PACKAGE OWNER` clause of [CREATE PROCEDURE \(SQL - native\) \(Db2 SQL\)](#) and the `OWNERTYPE` option of the [OWNER bind option \(Db2 Commands\)](#).

Page sampling for inline statistics

Beginning in function level 500, the REORG TABLESPACE and LOAD utilities can now use page sampling when they gather inline statistics. Page sampling has the potential to reduce both CPU time and elapsed time. In earlier Db2 releases, the RUNSTATS utility can use page sampling, but inline statistics that are gathered by other utilities can use row sampling only. To use page sampling for inline statistics with REORG TABLESPACE or LOAD, specify the TABLESAMPLE SYSTEM option or ensure that the STATPGSAMP subsystem parameter is set to SYSTEM (the default) or YES. In function level 500, STATPGSAMP is extended to apply to inline statistics. For more information, see the TABLESAMPLE SYSTEM option description in [Syntax and options of the LOAD control statement \(Db2 Utilities\)](#) and [Syntax and options of the REORG TABLESPACE control statement \(Db2 Utilities\)](#).

SQL Data Insights

Function level 500 delivers SQL Data Insights, an integrated solution that brings deep learning AI capabilities into Db2. SQL Data Insights uses unsupervised neural networks to generate a specialized vector-embedding model called database embedding, which can be referenced through SQL queries called "cognitive intelligence" queries.

The SQL Data Insights user interface is an optional feature available at no additional charge with Db2 13, which provides the user interface for training models and exploring data insights. Db2 provides the in-database infrastructure for training and model table (vector table) management. Db2 also provides three new built-in cognitive functions to speed up query execution.

For more information, see [Running AI queries with SQL Data Insights](#).

Reduced ECSA storage for IFI buffers

Db2 13 reduces the use of ECSA storage for IFI buffers from a maximum of 50 MB to a fixed 8 MB.

Function level 100 reduces the use of ECSA storage for IFI buffers to a maximum of 25 MB. Then, after function level 500 is first activated, it is further reduced to 8 MB. The storage behavior that is introduced in function level 500 continues even if you later activate function level 100*.

To compensate for the reduction in ECSA storage, you must set aside an extra 50 MB for HVCOMMON and 25 MB for private storage. You can reduce the ECSA storage after function level 500 is activated and Db2 starts using the new storage pools. When Db2 uses the new storage pools, the use of ECSA for the retrieval of IFI records noticeably decreases. You can monitor use of the new storage pools by starting the statistics trace to collect IFCID 0225. Then, you can check the SHARED / COMMON storage summary report in the formatted IDCID 225 SMF trace record.

For more information about ECSA storage requirements, see [Calculating the storage requirement for the extended common service area \(Db2 Installation and Migration\)](#).

Online conversion of tables from growth-based (PBG) to range-based (PBR) partitions

Function level 500 introduces the capability to convert the partitioning scheme of a table with growth-based partitions (in a PBG table space) to use range-based partitions (in a PBR table space). The conversion can be completed as an online change with minimal impact to your applications.

PBG and PBR universal table spaces (UTS) are the strategic table space types for tables in Db2 for z/OS. PBG table spaces are the default UTS type, and they are well-suited for small to medium-sized tables. However, if an existing table in a PBG table space grows too large, performance degradation or data and index management issues might arise. Consider converting from PBG to PBR when that occurs.

To complete the conversion, you issue an ALTER TABLE statement with the new ALTER PARTITIONING TO PARTITION BY RANGE clause and run the REORG TABLESPACE utility to materialize the pending change. The table space for the table is converted to PBR with relative page numbers (RPN).

For more information, see [Converting tables from growth-based to range-based partitions \(Db2 Administration Guide\)](#) and "ALTER PARTITIONING TO PARTITION BY RANGE" in [ALTER TABLE \(Db2 SQL\)](#).

Fast index traversal (FTB) support for larger index keys

Function level 500 extends FTB support to unique indexes with a key size for the ordering columns up to 128 bytes and nonunique indexes with a key size up to 120 bytes. For more information, see [Fast index traversal \(Db2 Performance\)](#).

Increased control for applications over how long to wait for a lock timeout

Function level 500 introduces the CURRENT LOCK TIMEOUT special register and the SET CURRENT LOCK TIMEOUT SQL statement to allow the lock timeout value to be set at the application level. So, you can set a lock timeout interval that suits the needs of a specific application, or even an individual SQL statement. Doing so minimizes application lock contention and simplifies portability of applications to Db2, without the need to assign the application to a separate Db2 subsystem.

The value of the CURRENT LOCK TIMEOUT special register overrides the value of the IRLMRWT subsystem parameter. It applies to certain processes related to locking, like the claim or drain of an object and cached dynamic statement quiescing.

For more information, see [CURRENT LOCK TIMEOUT \(Db2 SQL\)](#) and [SET CURRENT LOCK TIMEOUT \(Db2 SQL\)](#).

You can limit use of CURRENT LOCK TIMEOUT by setting the new SPREG_LOCK_TIMEOUT_MAX subsystem parameter. For more information, see [LOCK TIMEOUT MAX \(SPREG_LOCK_TIMEOUT_MAX subsystem parameter\) \(Db2 Installation and Migration\)](#).

You can also use Db2 profile tables to specify an assignment for the CURRENT LOCK TIMEOUT special register, for both remote and local threads. See [Setting special registers by using profile tables \(Db2 Administration Guide\)](#).

Profile table enhancements for application environment settings

Db2 13 introduces the capability to use system profiles for local applications in certain situations. Previously, the initial values for special registers and system built-in global variables can be specified in the Db2 profile tables, but they are used only for initialization with distributed threads. The new Db2 profile table support for local applications requires Db2 to be started with the DDF subsystem parameter set to AUTO or COMMAND. See [DDF STARTUP OPTION field \(DDF subsystem parameter\) \(Db2 Installation and Migration\)](#).

Starting in function level 500, Db2 profile tables can now be used for both local and remote applications in the following situations:

- You can specify assignments to the new CURRENT LOCK TIMEOUT special register. For more information, see [CURRENT LOCK TIMEOUT \(Db2 SQL\)](#) and [Setting special registers by using profile tables \(Db2 Administration Guide\)](#).
- You can specify a new RELEASE_PACKAGE keyword with a COMMIT attribute to change the release bind option for a package. See [Overriding the RELEASE\(DEALLOCATE\) option for packages by using profile tables \(Db2 Performance\)](#).

Ability to delete an active log data set from the BSDS while Db2 is running

Function level 500 introduces the new REMOVELOG option for the -SET LOG command to support online removal of an active log data set from the BSDS, eliminating the need to stop Db2 to accomplish the task by using the offline utility DSNJU003. The -SET LOG REMOVELOG command deletes the specified log from the BSDS if it is not in use or mark the log REMOVAL PENDING if it is in use.

To provide monitoring of the current active log status for log data sets with REMOVAL PENDING status, function level 500 also introduces the DETAIL option for the -DISPLAY LOG command. It shows information regarding REMOVAL PENDING status for local active log data sets. The output from the utility DSNJU004 also shows the REMOVAL PENDING status where applicable.

For more information, see [Deleting an active log data set from the BSDS with the -SET LOG command \(Db2 Administration Guide\)](#).

Index split instrumentation enhancements

Function level 500 introduces IFCID 0396 to provide detailed information about index splits. When data is inserted into a base table, the corresponding indexes are modified accordingly. As a result, performance of SQL insert operations can be impacted during the *index split* process, where synchronous I/O is required under data sharing and the group buffer pool GBP is dependent on the related index page sets.

The existing IFCID 0359 records already contain information index split events. However, the information that is recorded is not detailed enough to identify the cause of performance issues. IFCID 0359 is also disabled by default, and it can miss capturing some abnormal index split situations.

IFCID 0396 is always enabled by default under statistics trace class 3 and performance trace class 6. An IFCID 396 record is generated when an index split is considered an abnormal split process, such as when the total elapsed time is greater than 1 second. The generated IFCID 0396 record contains information such as the DBID, PSID, member ID, URID, page number, and more. The information is helpful for both customers and IBM Support to identify the root cause of INSERT performance issues.

For more information, see the IFCID 0396 descriptions in [Chapter 9, “IFCID changes,” on page 51](#) and in the [Trace field descriptions \(Db2 Performance\)](#).

Starting after the catalog level V13R1M501 update, the following RTS columns in the SYSIBM.SYSINDEXSPACESTATS catalog table are populated. They record and aggregate general index split information since last table reorganization, index rebuild, or load replace:

Column name	Data type	Description
REORGTOTALSPLITS	INTEGER	The number of index splits since last reorganization or rebuild.

Column name	Data type	Description
REORGSPPLITTIME	BIGINT	Aggregated elapsed time for all index splits since last reorganization or rebuild.
REORGEXCSPLITS	INTEGER	The number of abnormal index splits (such as elapsed times greater than 1 second) since last reorganization or rebuild.

Db2 starts populating these RTS columns as soon as the catalog level V13R1M501 update completes, even before function level 501 is activated.

SPT01 and SYSLGRNX table spaces are converted to DSSIZE 256 GB

Starting in function level 500, the first time that the REORG TABLESPACE utility runs for the following directory objects, it converts the DSSIZE to 256 GB.

- DSNCDB01.SPT01 to resolve issues that are related to the removal of the SPT01_INLINE_LENGTH subsystem parameter by APAR PH24358 in Db2 12.
- DSNCDB01.SYSLGRNX in anticipation of future growth in this table for increasing workloads and conversions of non-UTS table space to UTS.

The conversion is automatic and does not require any special utility syntax. It updates the following Db2 catalog table values for each table space:

- The DSSIZE columns in SYSIBM.SYSTABLESPACE and SYSIBM.SYSTABLEPART are updated to 256G.
- A SYSCOPY record is inserted for the table space, with the following values to indicate that REORG changed the DSSIZE: ICTYPE = 'A', STYPE = 'D', TTYPE = '64G'.

If function level 100* is activated, already converted table spaces continue to use the larger DSSIZE, but the REORG utility does not convert unconverted table spaces.

Recovery to a point-in-time (PIT) before REORG converted the DSSIZE reverts the DSSIZE to 64GB. As always, if any one of the catalog or directory objects are recovered to a prior PIT, it is best to recover all catalog and directory objects to the same PIT.

Improved concurrency for altering tables for DATA CAPTURE

Function level 500 introduces a concurrency improvement for ALTER TABLE statements that change the DATA CAPTURE attribute of tables. With this enhancement, Db2 no longer waits for other statements that depend on the altered table to commit. As a result, the DATA CAPTURE alteration can now succeed even when concurrent statements are running continually against the table.

Earlier Db2 releases quiesce the following objects that depend on the altered table as part of the DATA CAPTURE alteration:

- Static packages
- Cached dynamic SQL statements

Because the DATA CAPTURE alteration waited for applications that depended on the altered table to commit, continuous concurrent activity on the table might cause the ALTER TABLE statements to fail.

The new DATA CAPTURE attribute now takes effect immediately when the processing completes, even before the ALTER statement commits. As a result, concurrent statements on the same Db2 member might write out different log formats in the same transaction. For more information, see [Altering a table to capture changed data \(Db2 Administration Guide\)](#).

Change REORG INDEX SHRLEVEL REFERENCE or CHANGE so the NOSYSUT1 keyword is the default

Starting at function level 500, the NOSYSUT1 keyword is the default for the REORG INDEX utility when specified with the SHRLEVEL REFERENCE or CHANGE keywords. So, the utility avoids use of a work data set, which improves performance and allows REORG INDEX to use parallel subtasks to unload and to build index keys. The default value of the REORG_INDEX_NOSYSUT1 subsystem parameter is also changed from NO to YES, and YES is now the only option. So, this subsystem parameter no longer influences the behavior of REORG INDEX.

For more information, see [Syntax and options of the REORG INDEX control statement \(Db2 Utilities\)](#) and [REORG TS NOPAD DEFAULT \(REORG_TS_NOPAD_DEFAULT subsystem parameter\) \(Db2 Installation and Migration\)](#).

CREATE TABLESPACE uses MAXPARTITIONS 254 by default

At application compatibility level V13R1M500 or higher, CREATE TABLESPACE statements use MAXPARTITIONS 254 by default.

When MAXPARTITIONS 256 is explicitly specified, the default DSSIZE varies from 4 G to 32 G depending on the page size. However, starting with application compatibility level V12R1M504, when MAXPARTITIONS is not explicitly specified, Db2 12 use MAXPARTITIONS 256 by default, but the default DSSIZE is always 4 G regardless of the page size.

This apparent inconsistency avoided a risk of failure for existing statements, where the default data set size might be greater than 4 G depending on the page size. The statements might fail with SQLCODE -904 with reason code 00D70008 if the data sets for the table space are not associated with a DFSMS data class that is specified with extended format and extended addressability.

With MAXPARTITIONS 254 as the default, the result is now consistent regardless of whether MAXPARTITIONS is explicitly specified. The calculated default DSSIZE is always 4 G.

See the MAXPARTITIONS and DSSIZE descriptions in [CREATE TABLESPACE \(Db2 SQL\)](#).

V13R1M500 application compatibility

Most new SQL capabilities become available only to applications that use the equivalent application compatibility (APPLCOMPAT) level or higher. For a list, see [Chapter 6, “SQL changes in Db2 13,” on page 41](#).

For more information about application compatibility levels, see [Chapter 21, “Controlling Db2 application compatibility,” on page 91](#).

How to activate function level 500

Before you begin

Before you activate function level 500 or higher, complete the following prerequisite tasks:

1. In Db2 12, identify and resolve incompatible changes and activate function level 510. You can run the pre-migration job DSNTIJPE in Db2 12 to identify the incompatible changes. For more information, see [Verify Db2 13 premigration activities and activate function level 510 in Db2 12 \(Db2 Installation and Migration\)](#).

2. Verify that every member was restarted with the fallback SPE applied in Db2 12.

Important: Inactive members that never started in Db2 12 with the fallback SPE applied cannot start after function level 500 is activated in Db2 13.

3. Migrate the Db2 subsystem or data sharing group to Db2 13, as described in [Migrating your Db2 subsystem to Db2 13 \(Db2 Installation and Migration\)](#) or [Migrating an existing data sharing group to Db2 13 \(Db2 Installation and Migration\)](#).

4. Verify that you no longer need to fall back to Db2 12.

Important: After function level 500 is activated in Db2 13, coexistence and fallback to Db2 12 are no longer possible. You can activate function level 100* to disable new capabilities in Db2 13, but function level 100* does not support coexistence or fallback .

5. In data sharing, ensure that the group has no active Db2 12 members. See [Migrating subsequent members of a group to Db2 13 \(Db2 Installation and Migration\)](#).

Procedure

To activate function level 500, complete the following steps:

1. Tailor the function level activation job by running the installation CLIST:

- a. In panel DSNTIPA1, specify `INSTALL TYPE ===> ACTIVATE`. Then, specify the name of the output member from the previous function level activation (or migration) in the `INPUT MEMBER` field, and specify a new member name in the `OUTPUT MEMBER` field.
 - b. In panel DSNTIP00, specify `TARGET FUNCTION LEVEL ===> V13R1M500`. The CLIST uses this value when it tailors the `ACTIVATE` command in the `DSNTIJAF` job and the `CATMAINT` utility control statement in the `DSNTIJTC` job. (Function level 500 uses catalog level 100, and the tailored `DSNTIJTC` job is not used.)
 - c. Proceed through the remaining CLIST panels, and wait for the CLIST to tailor the jobs for the activation process. The output data set contains the tailored jobs for the activation process.
2. Optionally, check that Db2 is ready for function level activation by issuing the following `ACTIVATE` command with the `TEST` option:

```
-ACTIVATE FUNCTION LEVEL (V13R1M500) TEST
```

Db2 issues message `DSNU757I` to indicate the results. For more information, see [Chapter 16, “Testing Db2 function level activation,”](#) on page 79.

3. Run the tailored `DSNTIJAF` job, or issue the following `ACTIVATE` command:

```
-ACTIVATE FUNCTION LEVEL (V13R1M500)
```

4. If you are ready for applications to use new capabilities in this function level, rebind them at the corresponding application compatibility level. For more information, see [Chapter 21, “Controlling Db2 application compatibility,”](#) on page 91.

Optionally, when you are ready for all applications to use the new capabilities of the target function level, you can run the following jobs:

- a. Run `DSNTIJUZ` to modify the subsystem parameter module with the `APPLCOMPAT` value that was specified on panel `DSNTIP00`.
- b. Run `DSNTIJOZ` job to issue `SET SYSPARM` command to bring the `APPLCOMPAT` subsystem parameter changes online.
- c. Run `DSNTIJUA` job to modify the Db2 data-only application defaults module with the `SQLLEVEL` value that was specified on panel `DSNITP00`.

What to do next

To activate new capabilities with catalog dependencies in Db2 13, activate function level 501 or higher. See [“Function level 501 \(Db2 13 installation or migration - May 2022\)”](#) on page 15.

Related tasks

[Activating Db2 13 function levels 500 and 501 \(Db2 Installation and Migration\)](#)

Related reference

[-ACTIVATE \(Db2\) \(Db2 Commands\)](#)

Function level 100 (for migrating to Db2 13 - May 2022)

Db2 starts at function level 100 (`V13R1M100`) during migration to Db2 13, and fallback and coexistence with Db2 12 in data sharing remain possible. Many new capabilities in Db2 13 remain unavailable.

Contents

[“New capabilities in function level 100”](#) on page 25

[“How to migrate to Db2 13 function level 100”](#) on page 30

Enabling APAR:	None
Full identifier:	V13R1M100
Catalog level required:	V13R1M100

Product identifier (PRDID):	DSN13010
Incompatible changes:	See Chapter 3, “Incompatible changes in Db2 13,” on page 35.

New capabilities in function level 100

In function level 100 (V13R1M100), Db2 runs on Db2 13 code, and virtual storage and many optimization enhancements in Db2 13 become available. However, most new application and SQL capabilities remain unavailable until you activate the function level that introduces them.

The following capabilities and enhancements in Db2 13 become available in function level 100.

Index look-aside optimization

Function level 100 introduces index look-aside optimizations, to improve performance for insert, update, and delete operations. This enhancement optimizes Db2 13 and improves its performance without changing how you configure, monitor, and use Db2.

Sort optimization

Function level 100 introduces enhanced sort optimizations, which were previously introduced for ORDER BY and GROUP BY processing. It applies them to improve the performance of certain operations, such as the following processing enhancements:

- Machine-generated code support for DECFLOAT processing.
- Support for the following enhancements for GROUPING SET, multiple DISTINCT, and PERCENTILE processing:
 - Machine-generated code support.
 - Sort processing can use its own work file.
 - A check for ordered data in the first iteration of a sort.
 - Larger sort trees can be used.
- SUBSTR support for the LISTAGG built-in function, if the start position and length for SUBSTR is a constant.
- Support for avoiding rereading of a single work file, if the sort work file is used, and if IBM Watson Machine Learning for z/OS is enabled.
- Support for reducing the length of long VARCHAR keys, if the last key in an ORDER BY list is a VARCHAR over 100 bytes, and if IBM Watson Machine Learning for z/OS is enabled.

This enhancement optimizes Db2 13 and improves its performance without changing how you configure, monitor, and use Db2.

Expanded SORTL usage with learning from execution (IBM z15)

Function level 100 introduces expanded SORTL usage based on machine learning on the amount of storage and the number of records being sorted, when run on IBM z15 or later processors. This enhancement optimizes Db2 13 and improves its performance without changing how you configure, monitor, and use Db2.

Improved locking for INSERT to partition-by-growth (PBG) table spaces

Function level 100 introduces retry logic for INSERT operations. An extra attempt is made to obtain a partition lock on a PBG table space after a failed first attempt, thereby increasing the success rate of INSERT operations.

Before this enhancement, only a single attempt was made to obtain a lock on the target partition. If the attempt failed, the target partition was skipped, and the next partition was evaluated. This process would continue until the INSERT operation either successfully obtained a partition lock or it finished searching all existing partitions without obtaining a partition lock.

In most cases, the duration of partition lock contention is short; however, because the INSERT operation did not make another attempt to obtain a lock on a partition after the first failed attempt,

the INSERT operation terminated unnecessarily. In many cases, making an extra attempt to obtain a partition lock results in the successful completion of an INSERT operation that otherwise would fail.

This enhancement optimizes Db2 13 and improves its performance without changing how you configure, monitor, and use Db2.

Reduced ECSA storage for IFI buffers

Db2 13 reduces the use of ECSA storage for IFI buffers from a maximum of 50 MB to a fixed 8 MB.

Function level 100 reduces the use of ECSA storage for IFI buffers to a maximum of 25 MB. Then, after function level 500 is first activated, it is further reduced to 8 MB. The storage behavior that is introduced in function level 500 continues even if you later activate function level 100*.

To compensate for the reduction in ECSA storage, you must set aside an extra 50 MB for HVCOMMON and 25 MB for private storage. You can reduce the ECSA storage after function level 500 is activated and Db2 starts using the new storage pools. When Db2 uses the new storage pools, the use of ECSA for the retrieval of IFI records noticeably decreases. You can monitor use of the new storage pools by starting the statistics trace to collect IFCID 0225. Then, you can check the SHARED / COMMON storage summary report in the formatted IDCID 225 SMF trace record.

For more information about ECSA storage requirements, see [Calculating the storage requirement for the extended common service area \(Db2 Installation and Migration\)](#).

Reduced agent local below-the-bar (BTB) storage

Starting in function level 100, Db2 supports a greater number of concurrent threads, by using above-the-bar (ATB) agent-local storage for statement text and attribute strings for dynamic SQL statements. In earlier releases, Db2 kept a copy of dynamic SQL statement text and attribute strings in agent local below-the-bar (BTB) storage while the statement is being prepared and executed.

For any specific thread, multiple dynamic SQL statements can be executing depending on the nesting level. The maximum length of an SQL statement is 2 MB, but much more storage can be allocated and the consumption of BTB storage could prevent the number of threads from scaling.

This enhancement optimizes Db2 13 and improves its performance without changing how you configure, monitor, and use Db2.

DBAT availability improvements

Function level 100 introduces changes to Db2 13 DBAT termination processing to support the following objectives:

- Reduction of the overall frequency and number of DBAT terminations.
- Reduction of the number of concurrent DBAT terminations that are caused by a short-term increase in DBAT usage.

This enhancement optimizes Db2 13 and improves its performance without changing how you configure, monitor, and use Db2.

Improved performance when using external security

Function level 100 introduces the following enhancements to improve performance for Db2 environments that use external security:

- Db2 caches plan authorization checks that the access control authorization exit (ACAE) routine uses. Previously, successful authorization checks on the EXECUTE privilege for plans were not cached if those checks were completed by the ACAE routine. This enhancement provides consistent behavior in plan authorization cache behavior regardless of whether security is managed with Db2 facilities or with the ACAE.

To enable plan authorization caching when the ACAE routine is being used, the AUTHEXIT_CACHEREFRESH subsystem parameter must be set to ALL and the z/OS release must be 2.5 or later. Db2 caches the results of authorization checks on the EXECUTE privilege for plans if a profile in the RACF class MDSNPN permits access to the plan. Db2 does not cache the results if access is allowed due to administrative authority, such as the DATAACCESS or SYSADM authorities.

For more information, see [Caching of EXECUTE on plans, packages, and routines \(Managing Security\)](#).

- Db2 is enhanced to cache more authorization IDs per plan. For more information, see [Caching authorization IDs for plans \(Managing Security\)](#).
- The AUTHCACH subsystem parameter is removed to simplify plan authorization cache management. Use the CACHESIZE bind option on the BIND PLAN subcommand to specify the size of the authorization cache for that plan. The default value is 4K.
- If the AUTHEXIT_CACHEREFRESH subsystem parameter is set to ALL, the global authentication cache takes the timestamp into consideration for user IDs that were authenticated by using credentials other than multi-factor authentication (MFA). For more information, see [Global authentication cache \(Managing Security\)](#).
- When you specify a key label for data set encryption, the specified key label cannot refer to an archived key for decryption operations only. Key labels can be specified by using the ENCRYPTION_KEYLABEL subsystem parameter or any of the following SQL statements:
 - ALTER STOGROUP
 - ALTER TABLE
 - CREATE STOGROUP
 - CREATE TABLE

If the specified key label refers to a decryption-only archived key, the key label specification fails and returns an error message. For more information on decryption-only archived keys, see [ICSF: Limit archived keys to decrypt operations only](#).

Reduced ECSA storage use for distributed data facility (DDF) processing

Function level 100 reduces the amount of ECSA storage that is used for processing DDF threads to be equivalent to processing local threads. The previous recommendation was an extra 2 KB per DDF thread. For more information, see [Calculating the storage requirement for the extended common service area \(Db2 Installation and Migration\)](#).

Improved storage monitoring and contraction

Function level 100 introduces the following enhancements to provide storage constraint relief:

- When below-the-bar Db2 storage consumption exceeds 64-percent threshold, Db2 automatically begins contraction of private storage pools.
- When extended common service area (ECSA) storage consumption exceeds the 85-percent threshold, Db2 automatically begins contraction of storage pools that are allocated in the ECSA.

In both cases, the storage contraction stops after storage consumption drops below the threshold.

This enhancement optimizes Db2 13 and improves its performance without changing how you configure, monitor, and use Db2.

Improved Db2 installation and migration process for customizing the amount of private storage for the IRLM lock control structure

In the MAX STORAGE FOR LOCKS field on installation panel DSNTIPJ, you can specify the maximum amount of private storage above the 2 GB bar for the IRLM lock control structure. In earlier Db2 releases, you can specify a value of only up to 102400 megabytes. Starting in Db2 13, you can specify a value of up to 16384 petabytes.

For more information, see [MAX STORAGE FOR LOCKS field \(Db2 Installation and Migration\)](#) and [MAX LOCK STORAGE UNIT field \(Db2 Installation and Migration\)](#).

REBIND simplification

Function level 100 introduces reduced storage usage during BIND/REBIND for queries that involve many tables. This enhancement optimizes Db2 13 and improves its performance without changing how you configure, monitor, and use Db2.

Partition range support in IFCID 306 for users of replication applications

Function level 100 introduces the capability for applications that collect IFCID 306 trace records for the log read process to request filtering on a range of partitions. For more information, see the new WQLSFLG flag and WQLSDBPP mapping in [Qualification fields for READS requests \(Db2 Performance\)](#).

EDITPROC support in IFCID 306 for users of nonproxy mode replication applications

Function level 100 introduces the support for any user of IFCID 306 in non-proxy mode to use EDITPROC support as an on-request function. For more information, see the new values X'04', X'05', X'06', and X'07' for the 1-byte WQALLOPT field in [Qualification fields for READS requests \(Db2 Performance\)](#).

Relative page numbers for new PBR table spaces

Starting in function level 100, the default value of the PAGESET_PAGENUM subsystem parameter is changed to RELATIVE. The PAGESET_PAGENUM subsystem parameter specifies the default value that Db2 uses when you omit the PAGENUM option in CREATE TABLESPACE or CREATE TABLE statement for a partition-by-range (PBR) table space. That is, it specifies whether Db2 creates the table space and associated partitioned indexes to use relative page numbers (RPN) or absolute page numbers (APN) across partitions. RPN is the strategic direction for PBR table spaces in Db2. If you accept the new default and create all new PBR table spaces with relative page numbers, you can avoid costly future conversions. Converting from absolute to relative pages numbers always requires a REORG of the entire table space.

See PAGE SET PAGE NUMBERING field (PAGESET_PAGENUM subsystem parameter) ([Db2 Installation and Migration](#)).

Improved default statistics collection granularity

Starting in function level 100, that default value of the STATIME_MAIN subsystem parameter is changed to 10 seconds. The STATIME_MAIN subsystem parameter specifies the time interval in seconds, for collection of interval-driven statistics that are not collected at the interval that is specified by the STATIME subsystem parameter. With the default statistics interval set at 60 seconds in earlier releases, Db2 database administrators and system programmers cannot identify true workload peaks by using Db2 statistics for subsystem level performance tuning and planning. Similarly, a slowdown of 5 - 15 seconds is difficult to diagnose with statistics collected at a 60-second interval.

See MAIN STATS TIME field (STATIME_MAIN subsystem parameter) ([Db2 Installation and Migration](#))

REPAIR utility WRITELOG for decompression dictionaries

Function level 100 introduces the capability to write a decompression dictionary log record up to the maximum log record size supported by Db2. This capability can be used after you run an application or utility that builds a new dictionary without writing the old one to the log. So, it is useful for replication products that would otherwise require a refresh of the replication target. After the decompression dictionary is successfully written, the REPAIR utility issues message DSNU3335I with the location of the log record. Applications can then use this information to insert a SYSIBM.SYSCOPY record.

For more information, see the option descriptions for TYPE X'4002 and SUBTYPE 'X'000A' in [Syntax and options of the REPAIR control statement \(Db2 Utilities\)](#), and [DSNU3335I \(Db2 Messages\)](#).

Enhanced space-level recovery with the RECOVER utility

Starting in function level 100, the RECOVER utility supports space-level recovery (where DSNUM ALL is specified or is the default), even if the image copies were created at the partition or piece level for the following objects:

- Universal table spaces (UTS).
- Index spaces or indexes for a UTS.
- XML UTS with a base table that resides in a UTS.
- Auxiliary index spaces or indexes for an XML UTS with a base table that resides in a UTS.
- LOB table spaces with a base table that resides in a UTS.

- Auxiliary index spaces or indexes for LOB table spaces with a base table that resides in a UTS.

Space-level recovery of these supported object types is processed as partition-level or piece-level recoveries in the RESTORE phase. Db2 catalog and directory objects that meet the criteria above are supported by this enhancement. When the list contains a mixture of supported and unsupported object types, recovery behavior for the unsupported object types is the same as Db2 12 or earlier releases.

When the RECOVER utility is invoked (with DSNUM ALL specified or as the default) in Db2 12 or earlier releases, for space-level recovery of table spaces, index spaces, or indexes, an error message is issued if the image copies were created at the partition or piece level. The DSNU512I (DATASET LEVEL RECOVERY IS REQUIRED) error message indicates that recovery cannot be done at the space level. Recovery must instead be requested at the partition or piece level. Objects with these errors are not recovered and the RECOVER ends with RC8 indicating errors were encountered.

For more information, see [Recovering a data set or partition \(Db2 Utilities\)](#).

Column names longer than 30 bytes

Function level 100 extends the maximum length of a column name from 30 bytes of EBCDIC, up to 128 bytes with limited support for using the longer column names. The longer column names can be used when the TABLE_COL_NAME_EXPANSION subsystem parameter setting is ON. Although you can now define a column with a name up to 128 bytes, column names with a length greater than 30 bytes of EBCDIC might be truncated on a character boundary. Column names returned in an SQLDA contain 30 bytes at most. APIs that do not use the SQLDA to obtain a column name might return complete column names.

For more information, see [Column names longer than 30 bytes \(Db2 SQL\)](#) and [TABLE_COL_NAME_EXPANSION in macro DSN6SPRM \(Db2 Installation and Migration\)](#).

Db2 support for z/OS continuous compliance

Customers are looking for solutions that provide evidence that they can trust the security of z/OS systems. z/OS 2.5 introduces new SMF type 1154 records that provide evidence of security compliance. Participating components and products can collect and write compliance data to their associated SMF 1154 subtype records. Function level 100 adds the capability to collect evidence on Db2 subsystems' compliance by writing SMF 1154 subtype 81 records. For more information, see [Db2 evidence for z/OS continuous compliance \(Managing Security\)](#) and [What is new in z/OS \(V2R4 - V2R5\)](#).

Enhanced dynamic allocation processing

In function level 100 and z/OS 2.5 or later, dynamic allocation processing supports system work blocks (SWBs) for data sets that are in 64-bit storage. This new dynamic allocation function helps reduce below-the-bar storage usage for address spaces that allocate large numbers of data sets.

To enable this feature, complete one of the following actions:

- Update the ALLOCxx parmlib member to set the SYSTEM SWBSTORAGE value to ATB. The default value is SWA, which indicates that SWBs reside in 24-bit storage or 31-bit storage. ATB indicates that SWBs are allowed to reside in 64-bit storage.
- Issue system command SETALLOC SYSTEM,SWBSTORAGE=ATB.

It is best to update the ALLOCxx parmlib member because the change remains effective across IPLs. If the SETALLOC command is used to enable SYSTEM SWBSTORAGE, you must restart Db2 for the change to take effect.

Previously, the CLIST calculation for data set storage size used 5 KB per open data set. With the new dynamic allocation function, the storage that is required per open data set is reduced to 4 KB. You must adjust the calculation for data set storage size. For more information, see [Calculating data set control block storage \(Db2 Installation and Migration\)](#).

More efficient cleanup for above-the-bar storage

Function level 100 introduces improvements to how Db2 manages and frees above-the-bar storage, especially to reduce the disruptive impact of issuing excessive IARV64 REQUEST(DISCARDDATA) service requests.

Db2 13 no longer issues the IARV64 REQUEST(DICARDDATA) request during thread deallocation or at certain intervals of COMMIT, and enhanced storage management is no longer controlled by the REALSTORAGE_MANAGEMENT subsystem parameter, which is also removed. In Db2 13, the storage is returned to the memory object. A system-level timer drives contraction for the memory object to release unused frames back to z/OS. Also, Db2 13 periodically checks the available free frames before the LPAR starts to page (by using the z/OS calculations for available free frames and LO threshold). If this value becomes lower than 5 times the z/OS calculated OK threshold, the memory object contraction is triggered.

Accounting information on the longest wait times for common suspension types

When Db2 transactions take a long time, it is important to determine:

- Where the transaction time is spent.
- Whether the problem is many short suspensions or a few long suspensions.
- Which resources the suspensions are for.

Before this enhancement, detailed performance traces were required to find this information. This enhancement simplifies the diagnosis process by providing information in Db2 accounting records for the longest wait time for a number of common suspension types. The following suspension types are included:

- IRLM lock suspensions
- Db2 internal latch suspensions
- Waits for Db2 synchronous or asynchronous I/O
- Waits for Db2 service tasks
- Waits for page latches

For more information, see [Chapter 9, “IFCID changes,” on page 51](#) and the DSNWMSG file.

IBM z16 group buffer pool (GBP) residency time

Starting in function level 100, two new statistics are added to relevant group buffer pool statistics storage areas:

- The average time a data area resides in a storage class before it is reclaimed.
- The average time a directory entry resides in a storage class before it is reclaimed.

These new statistics are supported only if Db2 is installed on z/OS 2.4 or later with necessary PTFs applied. Db2 and the coupling facility must be running on IBM z16 or later processors, and the coupling facility must be at CFLEVEL 25 or higher. For more information, see [CFLEVEL and operating system level coexistence \(z/OS MVS Setting Up a Sysplex\)](#).

You can access these metrics with the IFCID record trace and the -DISPLAY GROUPBUFFERPOOL command. For more information, see "DSNB820I: Average residency times" in [DSNB750I \(Db2 Messages\)](#).

Subsystem parameter simplification

Function level 100 introduces changes to the default values for various subsystem parameters to match current best practices. It also removes a number of obsolete subsystem parameters. For a list of these changes, see [Chapter 10, “Subsystem parameter changes in Db2 13,” on page 55](#).

How to migrate to Db2 13 function level 100

To migrate to Db2 13 function level 100, complete the following tasks:

1. In Db2 12, identify and resolve incompatible changes and activate function level 510. You can run the pre-migration job DSNTIJPE in Db2 12 to identify the incompatible changes. For more information, see [Verify Db2 13 premigration activities and activate function level 510 in Db2 12 \(Db2 Installation and Migration\)](#).
2. Verify that every member was restarted with the fallback SPE applied in Db2 12.

Important: Inactive members that never started in Db2 12 with the fallback SPE applied cannot start after function level 500 is activated in Db2 13.

3. Migrate the Db2 subsystem or data sharing group to Db2 13, as described in [Migrating your Db2 subsystem to Db2 13 \(Db2 Installation and Migration\)](#) or [Migrating an existing data sharing group to Db2 13 \(Db2 Installation and Migration\)](#).

Related tasks

[Activating Db2 13 function levels 500 and 501 \(Db2 Installation and Migration\)](#)

[Adopting new capabilities in Db2 13 continuous delivery](#)

In Db2 13, function levels and application compatibility levels control the adoption of most new capabilities by Db2 subsystems and Db2 applications.

Related reference

[-ACTIVATE \(Db2\) \(Db2 Commands\)](#)

Part 2. Changes to plan for in Db2 13

Use this information when you are planning migration to Db2 13 and for planning to adopt new capabilities that Db2 13 introduces.

Db2 for z/OS News from the Lab blog: See the [Db2 for z/OS News from the Lab blog](#) for the latest news about new capabilities and enhancements in Db2 for z/OS continuous delivery, from the IBM experts who design, build, test, and support Db2

[Subscribe today!](#)

This section contains information about incompatible changes that might impact your migration to Db2 13 or the activation of Db2 13 function levels. It also contains summaries of the changes that Db2 13 introduces, including separate sections for changes in the Db2 13 initial release and changes in higher function levels. It also summarizes function the Db2 13 no longer supports, and deprecated function that remains supported, but might be removed eventually.

Related reference

[New, changed, and deleted codes \(Db2 Codes\)](#)

[New, changed, and deleted messages \(Db2 Messages\)](#)

Chapter 3. Incompatible changes in Db2 13

Before migrating your Db2 12 subsystem to Db2 13, applying maintenance in Db2 13, or activating higher function levels, familiarize yourself with incompatible changes that might impact your Db2 environment. Plan to resolve any such applicable incompatible changes that apply to your Db2 environment before or during the Db2 13 migration process, or before applying maintenance.

Function level	Explanation and possible impact	Actions to take
V13R1M100	<p>Automatic rebind of plans and packages created before Db2 11</p> <p>Migration-related <i>automatic binds</i> (also called "autobinds") occur in Db2 13 because it cannot use runtime structures from a plan or package that was last bound in a release earlier than Db2 11. Plans and packages that were bound in Db2 12 can run in Db2 13, without the risk of migration-related autobinds. However, plans and packages that are bound in Db2 13 cannot run on Db2 12 members without an autobind in Db2 12.</p> <p>If you specify YES or COEXIST for the ABIND subsystem parameter, Db2 13 automatically rebinds plans and packages that were bound before Db2 11 when Db2 executes the packages.</p> <p>If you specify NO for the ABIND subsystem parameter, negative SQLCODEs are returned for each attempt to run a package or plan that was bound before Db2 11. SQLCODE -908, SQLSTATE 23510 is returned for packages, and SQLCODE -923, SQLSTATE 57015 is returned for plans until they are rebound in Db2 13.</p>	<p>Rebind all packages that were last bound before Db2 11 in Db2 12, before you migrate to Db2 13. This action is required for any packages run in the last 18 months, before you can migrate to Db2 13. For more information, see Verify Db2 13 premigration activities and activate function level 510 in Db2 12 (Db2 Installation and Migration).</p> <p>For more information about the impacts that migration-related automatic rebinds can have in your Db2 environment and actions that you can take to avoid them, see Rebind old plans and packages in Db2 12 to avoid disruptive autobinds in Db2 13 (Db2 Installation and Migration).</p>
V13R1M100	<p>Removed subsystem parameters</p> <p>Db2 13 enforces use of the default setting for several removed subsystem parameters. For the list of these subsystem parameters, see Chapter 10, "Subsystem parameter changes in Db2 13," on page 55.</p> <p>A Db2 12 environment that uses a different setting than is enforced in Db2 13 might encounter problems at a migration, especially in Db2 data sharing coexistence, where some members run on Db2 12 while others are already migrated to Db2 13.</p>	<p>Before migrating to Db2 13:</p> <ol style="list-style-type: none"> 1. Review the list of removed subsystem parameters, and the settings enforced by Db2 13. 2. If your Db2 12 environment uses different settings, evaluate the potential impact of the new setting to your environment, and make any necessary changes. 3. Ensure that your subsystem parameters module runs with the new settings before migrating to Db2 13.

Function level	Explanation and possible impact	Actions to take
V13R1M100	<p>Removal of DDF_COMPATIBILITY</p> <p>Db2 13 always operates as if this subsystem parameter setting is null.</p>	<p>If the DDF_COMPATIBILITY subsystem parameter is set to any non-null value in Db2 12, evaluate and make any necessary changes so that you can set it to null before migrating to Db2 13.</p>

Related concepts

Changes to plan for in Db2 13

Use this information when you are planning migration to Db2 13 and for planning to adopt new capabilities that Db2 13 introduces.

Related reference

Deprecated function in Db2 13

Certain capabilities that Db2 13 for z/OS supports are *deprecated*, meaning that their use is discouraged. Although they remain supported except as noted below in Db2 13, support is likely to be removed eventually.

Chapter 4. Storage changes in Db2 13

Db2 13 introduces changes to the storage configuration for a Db2 for z/OS installation.

More efficient cleanup for above-the-bar storage

Function level 100 introduces improvements to how Db2 manages and frees above-the-bar storage, especially to reduce the disruptive impact of issuing excessive IARV64 REQUEST(DISCARDDATA) service requests.

Db2 13 no longer issues the IARV64 REQUEST(DICARDDATA) request during thread deallocation or at certain intervals of COMMIT, and enhanced storage management is no longer controlled by the REALSTORAGE_MANAGEMENT subsystem parameter, which is also removed. In Db2 13, the storage is returned to the memory object. A system-level timer drives contraction for the memory object to release unused frames back to z/OS. Also, Db2 13 periodically checks the available free frames before the LPAR starts to page (by using the z/OS calculations for available free frames and LO threshold). If this value becomes lower than 5 times the z/OS calculated OK threshold, the memory object contraction is triggered.

Improved storage monitoring and contraction

Function level 100 introduces the following enhancements to provide storage constraint relief:

- When below-the-bar Db2 storage consumption exceeds 64-percent threshold, Db2 automatically begins contraction of private storage pools.
- When extended common service area (ECSA) storage consumption exceeds the 85-percent threshold, Db2 automatically begins contraction of storage pools that are allocated in the ECSA.

In both cases, the storage contraction stops after storage consumption drops below the threshold.

This enhancement optimizes Db2 13 and improves its performance without changing how you configure, monitor, and use Db2.

Reduced ECSA storage use for distributed data facility (DDF) processing

Function level 100 reduces the amount of ECSA storage that is used for processing DDF threads to be equivalent to processing local threads. The previous recommendation was an extra 2 KB per DDF thread. For more information, see [Calculating the storage requirement for the extended common service area \(Db2 Installation and Migration\)](#).

Reduced ECSA storage for IFI buffers

Db2 13 reduces the use of ECSA storage for IFI buffers from a maximum of 50 MB to a fixed 8 MB.

Function level 100 reduces the use of ECSA storage for IFI buffers to a maximum of 25 MB. Then, after function level 500 is first activated, it is further reduced to 8 MB. The storage behavior that is introduced in function level 500 continues even if you later activate function level 100*.

To compensate for the reduction in ECSA storage, you must set aside an extra 50 MB for HVCOMMON and 25 MB for private storage. You can reduce the ECSA storage after function level 500 is activated and Db2 starts using the new storage pools. When Db2 uses the new storage pools, the use of ECSA for the retrieval of IFI records noticeably decreases. You can monitor use of the new storage pools by starting the statistics trace to collect IFCID 0225. Then, you can check the SHARED / COMMON storage summary report in the formatted IDCID 225 SMF trace record.

For more information about ECSA storage requirements, see [Calculating the storage requirement for the extended common service area \(Db2 Installation and Migration\)](#).

Reduced agent local below-the-bar (BTB) storage

Starting in function level 100, Db2 supports a greater number of concurrent threads, by using above-the-bar (ATB) agent-local storage for statement text and attribute strings for dynamic SQL statements. In earlier releases, Db2 kept a copy of dynamic SQL statement text and attribute strings in agent local below-the-bar (BTB) storage while the statement is being prepared and executed.

For any specific thread, multiple dynamic SQL statements can be executing depending on the nesting level. The maximum length of an SQL statement is 2 MB, but much more storage can be allocated and the consumption of BTB storage could prevent the number of threads from scaling.

This enhancement optimizes Db2 13 and improves its performance without changing how you configure, monitor, and use Db2.

Chapter 5. Command changes in Db2 13

You can use this information to plan for changes in Db2 and related commands at migration to Db2 13 and in continuous delivery.

Command changes in function level 100

The following changes to commands take effect when you migrate to function level 100 in Db2 13. Db2 starts at function level 100 (V13R1M100) during migration to Db2 13, and fallback and coexistence with Db2 12 in data sharing remain possible. Many new capabilities in Db2 13 remain unavailable.

Command	Change introduced	APAR	Incompatibl e change?
-START TRACE (Db2) (Db2 Commands)	IFCID 0369 is added to statistics trace class 1. When -START TRACE (STAT) is issued without the CLASS parameter or with the CLASS(1) parameter, IFCID 0369 is activated by default. For compatibility with previous Db2 releases, statistics class 9, which contains IFCID 0369, is still available.	None	No

For more information about these changes, see [“Function level 100 \(for migrating to Db2 13 - May 2022\)”](#) on page 24.

Related information

[About Db2 and related commands \(Db2 Commands\)](#)

Chapter 6. SQL changes in Db2 13

You can use this information to plan for SQL changes at migration to Db2 13 and in continuous delivery.

SQL changes in function level 501

The following SQL changes take effect in Db2 13 for applications that run at application compatibility V13R1M501 or higher. Function level 501 (V13R1M501) is the first opportunity after migration to Db2 13 for applications to use new features and capabilities that depend on catalog changes in Db2 13.

SQL element	Change introduced	Incompatibl e change?
DEADLOCK_RESOLUTION_PRIORITY built-in global variable	New built-in global variable	No

For more information about these changes, see [“Function level 500 \(for migrating to Db2 13 - May 2022\)”](#) on page 19.

SQL changes in function level 500

The following SQL changes take effect in Db2 13 for applications that run at application compatibility V13R1M500 or higher. Activating function level 500 (V13R1M500) prevents coexistence with and fallback to Db2 12. Function level 500 is also the first opportunity for applications to use many of the new capabilities in Db2 13. However, new capabilities that depend on Db2 13 catalog changes remain unavailable.

SQL element	Change introduced	Incompatibl e change?
ALTER FUNCTION (compiled SQL scalar) (Db2 SQL)	New clauses: AS ROLE or AS USER on the PACKAGE OWNER clause	No
ALTER PROCEDURE (SQL - native) (Db2 SQL)	New clauses: AS ROLE or AS USER on the PACKAGE OWNER clause	No
CREATE FUNCTION (compiled SQL scalar) (Db2 SQL)	New clauses: AS ROLE or AS USER on the PACKAGE OWNER clause	No
CREATE PROCEDURE (SQL - native) (Db2 SQL)	New clauses: AS ROLE or AS USER on the PACKAGE OWNER clause	No
ALTER TABLE (Db2 SQL)	New clause: ALTER PARTITIONING TO PARTITION BY RANGE New concurrency behavior when the DATA CAPTURE clause is specified.	No
CREATE TABLESPACE (Db2 SQL)	The default value for the MAXPARTITIONS clause is changed from 256 to 254.	No
SET CURRENT LOCK TIMEOUT	New statement	No
CURRENT LOCK TIMEOUT special register	New special register	No

SQL element	Change introduced	Incompatibl e change?
AI_ANALOGY	New built-in function	No
AI_SEMANTIC_CLUSTER	New built-in function	No
AI_SIMILARITY	New built-in function	No

For more information about these changes, see [“Function level 500 \(for migrating to Db2 13 - May 2022\)”](#) on page 19.

SQL changes in function level 100

The following SQL changes take effect in Db2 13 for applications that run at application compatibility V13R1M100 or higher. Db2 starts at function level 100 (V13R1M100) during migration to Db2 13, and fallback and coexistence with Db2 12 in data sharing remain possible. Many new capabilities in Db2 13 remain unavailable.

SQL element	Change introduced	APAR	Incompatibl e change?
Column names longer than 30 bytes	Function level 100 extends the maximum length of a column name from 30 bytes of EBCDIC, up to 128 bytes with limited support for using the longer column names. The longer column names can be used when the TABLE_COL_NAME_EXPANSION subsystem parameter setting is ON. Although you can now define a column with a name up to 128 bytes, column names with a length greater than 30 bytes of EBCDIC might be truncated on a character boundary. Column names returned in an SQLDA contain 30 bytes at most. APIs that do not use the SQLDA to obtain a column name might return complete column names.	None	No
CREATE TABLE statement	The PAGESET_PAGENUM subsystem parameter specifies the default for the PAGENUM option, and its default value is changed from ABSOLUTE to RELATIVE. see PAGE SET PAGE NUMBERING field (PAGESET_PAGENUM subsystem parameter) (Db2 Installation and Migration).	None	No
CREATE TABLESPACE statement	The PAGESET_PAGENUM subsystem parameter specifies the default for the PAGENUM option, and its default value is changed from ABSOLUTE to RELATIVE. see PAGE SET PAGE NUMBERING field (PAGESET_PAGENUM subsystem parameter) (Db2 Installation and Migration).	None	No

For more information about these changes, see [“Function level 100 \(for migrating to Db2 13 - May 2022\)”](#) on page 24.

Related reference

[Statements \(Db2 SQL\)](#)

[Reserved words \(Db2 SQL\)](#)

Chapter 7. Utility changes in Db2 13

You can use this information to plan for Db2 Utilities changes at migration to Db2 13 and in continuous delivery.

Utility changes in function level 500

The following changes take effect when you activate function level 500 in Db2 13. Activating function level 500 (V13R1M500) prevents coexistence with and fallback to Db2 12. Function level 500 is also the first opportunity for applications to use many of the new capabilities in Db2 13. However, new capabilities that depend on Db2 13 catalog changes remain unavailable.

Utility	Change introduced	Incompatible change?
LOAD	New option: TABLESAMPLE SYSTEM	No
REORG INDEX	NOSYSUT1 is now the default when SHRLEVEL REFERENCE or CHANGE is specified.	No
REORG TABLESPACE	New option: TABLESAMPLE SYSTEM	No

For more information about these changes, see [“Function level 500 \(for migrating to Db2 13 - May 2022\)”](#) on page 19.

Utility changes in function level 100

The following changes take effect when you migrate to function level 500 in Db2 13. Db2 starts at function level 100 (V13R1M100) during migration to Db2 13, and fallback and coexistence with Db2 12 in data sharing remain possible. Many new capabilities in Db2 13 remain unavailable.

Utility	Change introduced	APAR	Incompatible change?
RECOVER	For certain object types, the RECOVER utility is enhanced to support space level recovery (where DSNUM ALL is specified or is the default), even when the image copies were created at the partition or piece level.	None	No
REPAIR	New option values: <ul style="list-style-type: none">• TYPE X'4002• SUBTYPE 'X'000A'	None	No

For more information about these changes, see [“Function level 100 \(for migrating to Db2 13 - May 2022\)”](#) on page 24.

Related concepts

[Basic information about Db2 utilities \(Db2 Utilities\)](#)

Chapter 8. Catalog changes in Db2 13

You can use this information to plan for catalog changes at migration to Db2 13 and in continuous delivery.

Catalog changes in catalog level V13R1M501

The following catalog changes take effect in Db2 13 when you tailor the Db2 catalog for function level 501 activation.

Catalog object	Change introduced	Incompatible change?
DSNKDX02 catalog index	This index is removed.	No
<u>SYSCOPY catalog table</u>	New column: EVENTID	No
<u>SYSINDEXSPACESTATS catalog table (Db2 SQL)</u>	Changed columns: <ul style="list-style-type: none"> • COPYCHANGES • EXTENTS • REORGAPPENDINSERT • REORGDELETES • REORGINSERTS • REORGPSEUDODELETES • SPACE • STATSDELETES • STATSINSERTS New columns: <ul style="list-style-type: none"> • REORGECSPLITS • REORGSPPLITIME • REORGTOTALSPLITS 	No
<u>SYSPACKAGE catalog table</u>	New column: DEPLEVEL Changed column: VALID	No
<u>SYSPACKCOPY catalog table</u>	New column: DEPLEVEL Changed column: VALID	No
<u>SYSPACKSTMT catalog table</u>	New column: VALID	No
<u>SYSPACKSTMTCOPY catalog table</u>	New table New table space: DSNCB06.SYSTSPSC New indexes: DSNKTX01 DSNKTX02	No
<u>SYSPACKSTMTDEP catalog table</u>	New table New table space: DSNCB06.SYSTSPSD New indexes:	No

Catalog object	Change introduced	Incompatibl e change?
	DSNKNX01 DSNKNX02	
<u>SYSTABLESPACESTATS catalog table (Db2 SQL)</u>	Changed columns: <ul style="list-style-type: none"> • COPYCHANGES • EXTENTS • REORGDELETES • REORGINSERTS • REORGUPDATES • STATSDELETES • STATSINSERTS • STATSUPDATES 	No
<u>SYSUTILITIES catalog table</u>	New table New table space: DSNDB06.SYSTSUTL New indexes: DSNULX01 DSNULX02	No
<u>SYSIBM.DSNSEQ_EVENTID sequence</u>	A Db2-supplied sequence that is intended to be used to generate a value for the EVENTID column when a row is inserted into the SYSIBM.SYSUTILITIES catalog table.	No

For more information about these changes, see [“Function level 501 \(Db2 13 installation or migration - May 2022\)”](#) on page 15.

Directory changes in function level 500

Function level 500 introduces the following directory changes:

SPT01 and SYSLGRNX table spaces are converted to DSSIZE 256 GB

Starting in function level 500, the first time that the REORG TABLESPACE utility runs for the following directory objects, it converts the DSSIZE to 256 GB.

- DSNDB01.SPT01 to resolve issues that are related to the removal of the SPT01_INLINE_LENGTH subsystem parameter by APAR PH24358 in Db2 12.
- DSNDB01.SYSLGRNX in anticipation of future growth in this table for increasing workloads and conversions of non-UTS table space to UTS.

The conversion is automatic and does not require any special utility syntax. It updates the following Db2 catalog table values for each table space:

- The DSSIZE columns in SYSIBM.SYSTABLESPACE and SYSIBM.SYSTABLEPART are updated to 256G.
- A SYSCOPY record is inserted for the table space, with the following values to indicate that REORG changed the DSSIZE: ICTYPE = 'A', STYPE = 'D', TTYPE = '64G'.

If function level 100* is activated, already converted table spaces continue to use the larger DSSIZE, but the REORG utility does not convert unconverted table spaces.

Recovery to a point-in-time (PIT) before REORG converted the DSSIZE reverts the DSSIZE to 64GB. As always, if any one of the catalog or directory objects are recovered to a prior PIT, it is best to recover all catalog and directory objects to the same PIT.

Catalog changes in catalog level V13R1M100

No catalog changes take effect when you tailor the Db2 catalog for migration to Db2 13 function level 100.

Related reference

[Db2 catalog tables \(Db2 SQL\)](#)

Chapter 9. IFCID changes

Db2 13 introduces IFCID changes.

Important: Db2 13 introduces continuous delivery of new capabilities and enhancements in function levels. Most new capabilities become available only after activation of the Db2 13 function level that introduces them, or when applications run with the corresponding application compatibility level. For more information, see [Chapter 18, “Activating Db2 13 function levels,”](#) on page 83.

New trace records

PSPI

The following table gives an overview of new IFCIDs. Serviceability trace records are not included.

IFCID	Trace type and class	Mapping macro	Description
0396	Statistics class 3, Performance class 6	DSNDQW05	Records information about abnormal index splits
0437	Performance class 3	DSNDQW05	Records information about the SET CURRENT LOCK TIMEOUT SQL statement

PSPI

Changes to selected trace records

PSPI

The following table gives an overview of changes to specific IFCIDs. Changes to IFCID 0106, the system parameters record, and changes to serviceability trace records are not included.

IFCID	Function level	Enhancement and description of changes
0001	V13R1M100	<p>DBAT availability improvements:</p> <p>The value in field QDSTNDBA, which records the number of times that DBATs were created, might increase significantly after migration to Db2 13. In previous Db2 releases, field QDSTNDBA excluded DBATs that were created to replace disconnected, pooled DBATs that were terminated because they reached their reuse limit. Starting in Db2 13, the count in field QDSTNDBA includes all DBATs that were created.</p> <p>Fields are added to record:</p> <ul style="list-style-type: none">• The current number of DBATs (distributed server threads) that are active because the associated packages were bound with KEEP DYNAMIC(YES).• The maximum number of DBATs that are active because the associated packages were bound with KEEP DYNAMIC(YES).• The number of DBATs that were terminated since DDF was started.• The number of DBATs that were terminated because they remained in the pool longer than the value specified by the POOLINAC subsystem parameter.

IFCID	Function level	Enhancement and description of changes
		<ul style="list-style-type: none"> • The number of DBATs that were terminated because they were reused more times than the reuse limit. • The number of times that threads that were used by connections from the remote site were terminated because they remained in the pool longer than the value specified by the pool thread timeout value. • The number of times that threads that were used by connections from the remote site were terminated because the TCP/IP socket was closed due to a connection loss.
0001	V13R1M100	Increase number of latch classes to 64: The following changes are made: <ul style="list-style-type: none"> • In the data that is mapped by DSNDQVLS, fields are added for latch classes 33 to 64. • Field QSST_P64DISYES is moved to the end of the data that is mapped by DSNDQSST.
0002, 0003	V13R1M100	Reduce RACF contention: Fields are added to record: <ul style="list-style-type: none"> • The number of checks for the plan execute privilege that were made using the plan authorization cache and were unsuccessful because an applicable entry was not found in the cache. • The number of times that Db2 overwrote an authorization ID in the plan authorization cache.
0002, 0003	V13R1M500	Application timeout and deadlock control: Fields are added to record: <ul style="list-style-type: none"> • The number of times that a SET CURRENT LOCK TIMEOUT statement was executed. • The number of times that the CURRENT LOCK TIMEOUT special register was set from a profile table.
0003	V13R1M100	Increase number of latch classes to 64: The following changes are made: <ul style="list-style-type: none"> • Field QLLLLC is expanded from one byte to two bytes.
0003	V13R1M100	Accounting information on the longest wait times for common suspension types: A section is added to record the following thread-level wait time information: <ul style="list-style-type: none"> • The longest wait for a lock or latch • The longest service task wait time • The longest wait for a page latch • The longest wait for synchronous or asynchronous I/O This information can simplify the task of diagnosing performance issues that are due to excessive wait times for resources.
0003	V13R1M500	SQL Data Insights support:

IFCID	Function level	Enhancement and description of changes
		Fields are added to record information about elapsed time and CPU time that Db2 spent while processing SQL Data Insights functions.
0051, 0052, 0056, 0057, 0148	V13R1M100	Increase number of latch classes to 64: The following changes are made: <ul style="list-style-type: none"> • Fields QW0051LC, QW0052LC, QW0056LC, QW0057LC, and QW0148LC are expanded from one byte to two bytes.
0172	V13R1M500	Application timeout and deadlock control: A field is added to record: <ul style="list-style-type: none"> • For the process that is waiting for a resource, the source of the value that Db2 uses to determine the victim of a deadlock (worth value).
0196	V13R1M500	Application timeout and deadlock control: Fields are added to record the following information about a lock request that times out: <ul style="list-style-type: none"> • For the thread that requests a lock and times out, the source of the timeout interval. The source is the IRLMRWT subsystem parameter, the CURRENT LOCK TIMEOUT special register, or an IRLM internal value. • For the thread that holds the lock, the source of the timeout interval that is set for the thread. The source is the IRLMRWT subsystem parameter, the CURRENT LOCK TIMEOUT special register, or an IRLM internal value.
0230, 0254	V13R1M100	IBM z15 group buffer pool (GBP) residency time: Fields are added to record: <ul style="list-style-type: none"> • The weighted average, in microseconds, of the elapsed time that a data area resides in a group buffer pool before the data area is reclaimed. • The weighted average, in microseconds, of the elapsed time that a directory entry resides in a group buffer pool before the directory entry is reclaimed.
0376	V13R1M500	Change of the default MAXPARTITIONS for partition-by-growth table spaces to 254: Function code 1315001 records when CREATE TABLESPACE or CREATE TABLE statements use a default MAXPARTITIONS value of 256 when the application compatibility level is V13R1M100, but use a default MAXPARTITIONS value of 254 when the application compatibility level is V13R1M500 or later. A record with function code 1315001 is written only when the application compatibility level is V13R1M100.



Related concepts

[Types of Db2 traces \(Db2 Performance\)](#)

Related reference

[Trace field descriptions \(Db2 Performance\)](#)

Chapter 10. Subsystem parameter changes in Db2 13

You can use this information to plan for Db2 subsystem parameter changes at migration to Db2 13 and in continuous delivery.

Subsystem parameter changes in function level 501

The following changes take effect when you activate function level 500 in Db2 13. Function level 501 (V13R1M501) is the first opportunity after migration to Db2 13 for applications to use new features and capabilities that depend on catalog changes in Db2 13.

Subsystem parameter	Change introduced	Incompatible change?
UTILITY_HISTORY	New subsystem parameter.	No

For more information about these changes, see [“Function level 501 \(Db2 13 installation or migration - May 2022\)”](#) on page 15.

Subsystem parameter changes in function level 500

The following changes take effect when you activate function level 500 in Db2 13. Activating function level 500 (V13R1M500) prevents coexistence with and fallback to Db2 12. Function level 500 is also the first opportunity for applications to use many of the new capabilities in Db2 13. However, new capabilities that depend on Db2 13 catalog changes remain unavailable.

Subsystem parameter	Change introduced	Incompatible change?
REORG_INDEX_NOSYSUT1	This subsystem parameter no longer influences use of the NOSYSUT1 behavior for the REORG INDEX utility with SHRLEVEL REFERENCE or SHRLEVEL CHANGE. The default value is also changed to YES, and it is protected value.	No
STATPGSAMP	This subsystem parameter also applies to the collection of inline statistics when the LOAD or REORG TABLESPACE utilities run with the STATISTICS keyword.	No

For more information about these changes, see [“Function level 500 \(for migrating to Db2 13 - May 2022\)”](#) on page 19.

Subsystem parameter changes in function level 100

The following changes take effect when you migrate to function level 500 in Db2 13. Db2 starts at function level 100 (V13R1M100) during migration to Db2 13, and fallback and coexistence with Db2 12 in data sharing remain possible. Many new capabilities in Db2 13 remain unavailable.

Subsystem parameter	Change introduced	APAR	Incompatible change?
AUTHEXIT_CACHEREFRESH	If the value is set to ALL and the z/OS release is 2.5 or later, Db2 refreshes the entries in the plan authorization cache when a resource access on the plan object profile is changed in RACF and the access control authorization exit (DSNX@XAC) is active.	—	No

Subsystem parameter	Change introduced	APAR	Incompatibl e change?
DDF	The default value is changed to AUTO.	—	No
DSMAX	The range is expanded with the maximum value increased from 200000 to 400000. The existing default of 200000 is retained.	—	No
EDM_SKELETON_POOL	The default value is changed from 51200 to 81920 in KB.	—	No
EDMDBDC	The default value is changed from 23400 to 40960 in KB.	—	No
FTB_NON_UNIQUE_INDEX	The default value is changed from NO to YES.	—	No
IRLMRWT	IRLMRWT is now online changeable to support more granular specification of the lock timeout value.	—	No
MAXCONQN	The default value is changed from OFF to ON.	—	No
MAXCONQW	The default value is changed from OFF to ON.	—	No
MAXSORT_IN_MEMORY	The default value is changed from 1000 to 2000 (2 MB) in KB.	—	No
NUMLKTS	The default value is changed from 2000 to 5000.	—	No
NUMLKUS	The default value is changed from 10000 to 20000.	—	No
OUTBUFF	The default value is changed from 4000K to 104857600 (100MB).	—	No
PAGESET_PAGENUM	The default value is changed from ABSOLUTE to RELATIVE.	—	No
SPREG_LOCK_TIMEOUT_M AX	New subsystem parameter	—	No
SRTPOOL	The default value is changed from 10000 to 20000 (20MB) in KB.	—	No
STATIME_MAIN	The default value is changed from 60 to 10.	—	No
TABLE_COL_NAME_EXPANS ION	New subsystem parameter.	—	No

For more information about these changes, see [“Function level 100 \(for migrating to Db2 13 - May 2022\)”](#) on page 24.

Removed subsystem parameters in Db2 13 (function level 100 and higher)

The following table lists subsystem parameters that are removed from this version of Db2 for z/OS. Refer to the information for the earlier version for detailed descriptions of the removed subsystem parameters.

Important: For best results, check the setting used in your Db2 12 environment, especially in data sharing environments. If the current setting does not match the setting listed in the following table,

evaluate whether any other changes are needed to accept the new-behavior settings, before migrating to Db2 13.

Subsystem parameter	Setting used in Db2 13	Description in earlier releases	Incompatible change?
AUTHCACH	4K	Specifies the size (in bytes per plan) of the authorization cache that is to be used if no CACHESIZE is specified on the BIND PLAN subcommand.	No
DDF_COMPATIBILITY	NULL	Controls certain characteristics of a connection between a client application and a Db2 for z/OS data server.	<u>Yes</u>
HONOR_KEEPPDICTIONARY	NO	Specifies whether Db2 honors the LOAD and REORG parameter KEEPPDICTIONARY when tables are converted between basic row format and reordered row format.	No
DSVCI	YES	Controls whether Db2-managed data sets that are created by CREATE TABLESPACE or CREATE INDEX statements are to have variable VSAM control intervals.	No
EXTRAREQ	100	Limits the number of extra DRDA query blocks that Db2 can request from a remote DRDA server.	No
EXTRSRV	100	Limits the number of extra DRDA query blocks that Db2 can return to a DRDA client	No
IMMEDWRI	NO	Determines when updates to group buffer pool-dependent buffers are to be written to the coupling facility.	No
IX_TB_PART_CONV_EXCLUDE	YES	Specifies whether to exclude trailing columns from the table-controlled partitioning keys when table spaces are converted from index-controlled partitioning to table-controlled partitioning.	No
MAXARCH	10000	Controls the maximum number of archive log volumes that are to be recorded in the BSDS.	No
MAXTYPE1	0	Determines the number of inactive DBATs that Db2 is to allow.	No
OPT1ROWBLOCKSORT	DISABLE	Specifies whether Db2 explicitly blocks sort operations when the OPTIMIZE FOR 1 ROW clause is specified on a query.	No
PARA_EFF	50	Controls the efficiency that Db2 assumes for parallelism when Db2 chooses an access path.	No

Subsystem parameter	Setting used in Db2 13	Description in earlier releases	Incompatible change?
PLANMGMTSCOPE	STATIC	Specifies the types of SQL statements for applying the PLANMGMT subsystem parameter setting.	No
REALSTORAGE_MANAGEMENT	AUTO	<p>Specifies whether Db2 should manage real storage consumption.</p> <p>Important: If you currently use a different setting, it is best not to change this setting to AUTO in Db2 12.</p> <p>Db2 13 uses enhanced automatic behavior for real storage management, which differs from the thread-based discard processing used with REALSTORAGE_MANAGEMENT=AUTO in Db2 12 or earlier.</p> <p>Db2 13 manages real storage discard processing at the system level to avoid z/OS RSM serialization from discard requests.</p>	No
RESYNC	2	Specifies the time interval, in minutes, between resynchronization periods.	No
SUBQ_MIDX	ENABLE	Specifies whether to enable or disable multiple index access on some non-Boolean uncorrelated subquery predicates.	No
TRACSTR	NO	Specifies whether the global trace starts automatically when Db2 starts.	No

Related reference

Directory of subsystem parameters, panel fields, and application default values (Db2 Installation and Migration)

Chapter 11. The extended 10-byte RBA and LRSN in Db2 13

As with earlier releases, Db2 13 continues to use the extended 10-byte RBA and LRSN format that was first introduced in Db2 11. No new changes that are related to the extended 10-byte RBA and LRSN format are introduced by Db2 13, and the BSDS conversion was already required at migration to Db2 12. However, if you still have page sets yet to be converted to use extended format, you might sometimes encounter problems with them. This information is included as a reminder to develop plans to convert any page sets that still use the basic 6-byte format.

Differences between the 6-byte and 10-byte formats

The terms "basic" and "extended" are sometimes used to refer to the 6-byte and 10-byte formats. When these terms are used, *basic format* refers to the 6-byte format, and *extended format* refers to the extended 10-byte format.

Conversion of RBA values

A 6-byte RBA value is converted to the 10-byte format value by adding zeros to the 4 most significant bytes. That is, the zeros are added to the left side of the value, as shown in the following table.

6-byte RBA value:	10-byte RBA value:
112233445566	00000000112233445566

Conversion LRSN values

A 6-byte LRSN value is converted to a 10-byte value by adding one zero byte to the left side and 3 bytes added to the right side of the value, as shown in the following table.

6-byte LRSN value:	10-byte LRSN value:
112233445566	00112233445566000000

The 3 bytes on the right side might be zero or x'FF', depending on the situation. For the beginning of an LRSN range, zeros are used. For the end of an LRSN range, x'FF' is used.

Internally, the values that are kept in memory are all 10 bytes, except when they need to be externalized to structures that remain in the 6-byte format. The values are stored internally as 10 bytes even in conversion mode. The conversion from the 10-byte values to 6-byte format is done at end points, such as when a log record is written, or when the PGLOGRBA field in a data or index page is updated.

When 10-byte format values are externalized

Extended RBA and LRSN values are externalized in the following contexts before objects are converted:

Messages

In Db2 13, all messages use 10-byte RBA and LRSN values, so that all messages have consistent formats. Sometimes Db2 needs an LRSN value that is not associated with a specific update. In this case, a log record with a matching LRSN might not exist. Such LRSN values are often generated with non-zero precision in the last 3 bytes, regardless of mode. Such full-precision 10-byte values might be seen in message output.

Database objects

The RBA (non-data sharing) or LRSN (data sharing) of the last change is stored in each page of every table and index.

When objects are in the basic format, the stored RBA or LRSN values are always 6 bytes. In the extended format, the stored RBA or LRSN values are 10 bytes. An installation typically converts objects from basic to extended format by using the REORG utility, but other methods exist. In

addition, the installation can decide which format is used to create new database objects. Database objects can be converted from extended to basic unless prohibited by a subsystem parameter.

Objects in basic format cannot be updated when the RBA or LRSN value is beyond the 6-byte range. For data sharing groups, the 6-byte LRSN range applies to the entire group and the 6-byte RBA range applies to each member. The 6-byte LRSN range does not apply to non-data sharing environments.

Recovery logs

The log records are assigned RBA values so that they can be located. In a data sharing environment, each log record has an associated LRSN value that is based on the time the log record was created. The LRSN value can be used to sequence log records from multiple members in a data sharing group.

All values that are passed to other Db2 components internally are 10-byte values padded with zeros. To all components outside of the log manager, the log always appears to be in the 10-byte format. Conversion of the log content to the new format that supports 10-byte RBA and LRSN values is completed when the installation converts the BSDSs to the 10-byte format. These two actions must be completed in lock step because the old BSDS format cannot accommodate larger RBA and LRSN values. For more information, see [How RBA and LRSN values are displayed \(Db2 Administration Guide\)](#) and [SYSLGRNX table \(Db2 SQL\)](#).

Bootstrap data sets (BSDS)

The BSDS contains the LRSN and RBA values that bound each active and archive log data and a number of others that have various purposes. The BSDS conversion was required at migration to Db2 12, so the BSDS always uses the extended 10-byte format in Db2 12 or later.



Attention: In Db2 subsystems that are not data sharing members, if Db2 is already at risk of reaching the 6-byte RBA limit, it is strongly recommended that you first convert all catalog and directory objects, then convert all user objects to the 10-byte RBA format, as soon as possible.



Attention: After the BSDS is converted to the 10-byte format, Db2 stops issuing messages to warn you about the risk of reaching the 6-byte RBA or LRSN limits. The increased size of all log records also accelerates progress toward the 6-byte RBA logging limit.

You must continuously monitor the RBA and LRSN values until all catalog, directory, and user objects are converted to the 10-byte RBA or LRSN format. Failure to convert page sets before the 6-byte soft logging limit is reached results in failed updates with reason code 00C2026D, and any objects still in the 6-byte format become read-only. RBA or LRSN values greater than x'F00000000000' indicate that your system is at risk of reaching the 6-byte logging limit.

Catalog table columns

The Db2 catalog and directory contain RBA and LRSN information in several tables.

Catalog and directory columns that contain RBA or LRSN values use 10-byte format. The catalog columns might be physically stored as either 6 bytes or 10 bytes. However, the values are converted to the 10-byte format as necessary when they are used in Db2.

Some 6-byte values still exist until a REORG of the affected catalog and directory tables is complete. The 6-byte values are padded with zeros when they are retrieved.

Shared communication area (SCA)

The SCA is used to track and communicate data pertinent to a data sharing group. This data always includes some LRSN and RBA values and there might be many such values, depending on the exception states of database objects.

Utilities

In Db2 13, RBA and LRSN values are displayed in 10-byte format. This 10-byte display is unrelated to migration of the catalog or directory, conversion of individual objects to extended format, or BSDS conversion. For recovery purposes, this 10-byte format is the preferred input format for Db2. When 10-byte RBA or LRSN values are specified as input to Db2, the values are converted to 6-byte format internally, as needed..

Work files

Data pages and space map pages for the work file database use the 10-byte format as soon as they are first accessed in Db2 11 or later (in any migration mode), regardless of whether the Db2

subsystem is migrated from DB2 10 or is a new installation. However, for migrated subsystems, the Db2 catalog is not updated to reflect the format of the work files. For more information about work files, see [Work file database \(Introduction to Db2 for z/OS\)](#).

When object and BSDS formats do not match

You can convert database objects to the 10-byte format when you are ready.



Attention: In Db2 subsystems that are not data sharing members, if Db2 is already at risk of reaching the 6-byte RBA limit, it is strongly recommended that you first convert all catalog and directory objects, then convert all user objects to the 10-byte RBA format, as soon as possible.

In Db2 subsystems that are not data sharing members, always convert all Db2 catalog, directory, and user objects to use the extended 10-byte RBA format before you convert the BSDS, especially if Db2 is close to reaching the logging limit for the 6-byte RBA. Failure to convert page sets to the 10-byte RBA format before Db2 reaches the 6-byte logging limit results in failed updates with reason code 00C2026D. No updates are allowed for any object that is still in the 6-byte format.

You must continuously monitor the RBA and LRSN values until all catalog, directory, and user objects are converted to the 10-byte RBA or LRSN format. Failure to convert page sets before the 6-byte soft logging limit is reached results in failed updates with reason code 00C2026D, and any objects still in the 6-byte format become read-only. RBA or LRSN values greater than x'F00000000000' indicate that your system is at risk of reaching the 6-byte logging limit.

If an object is in basic format and the log uses the 10-byte format, the LRSN that is stored in PGLOGRBA is truncated to fit. If a database object is in the extended format, and the log remains in the 6-byte format, LRSN values that are stored in the object are padded with zeros to the 10-byte format. Outside of data sharing environments, similar rules apply to RBA values.

When an object is in extended format and some members of the data sharing group have BSDS and logs in different formats, the order of updates is maintained. However, LRSN values from some members must be padded with zeros.

For a simple example, consider a data sharing group with two members:

- M10 is a member that has logs in the 10-byte format.
- M6 is a member that has logs in the 6-byte format.

Assume that the same data sharing group has two tables:

- TExt is a table with extended format.
- TBasic is a table with basic format.

The following illustration shows how a sequence of updates might look for the example data sharing group. These time values are for illustrative purposes. They are not representative of typical LRSN values because they correspond to updates that were completed in December, 1908.

Time	Update	Content of PGLOGRBA or PGBigRBA
001000000001000002	M10 updates TBasic	100000000001 A
001000000001000003	M10 updates TExt	00100000000001000003 B
001000000001000004	M6 updates TExt	00100000000002000000 C
001000000001000005	M6 updates TBasic	100000000002 D
001000000002000001	M10 updates TBasic	100000000003 E
001000000002000003	M10 updates TExt	00100000000002000003 F

Each of the example updates can be from different transactions, and the last two must be from separate transactions.

The logic ensures that the PGLOGRBA or PGBigRBA does not decrease, even though the two subsystems are logging updates with different formats.

- For the first update **A**, the LRSN is truncated before it is placed in PGLOGRBA.
- The value for the second update **B** stores the 10-byte format with full precision.
- For the third update **C**, member M6 must generate an LRSN value that is greater than the 6 bytes that correspond to the old LRSN value. (If the LRSN is beyond the 6-byte range, updates are not allowed).

- For the fourth update **D**, member M6 again generates a value that is greater than the existing PGLOGRBA or PGBigRBAvalue.
- For the fifth update **E**, M10 must generate a larger value. A value greater than 0010000002FFFFFF is used because the TBasic table uses a 6-byte format LRSN.
- For the last update **F**, the only requirement is that the LRSN must be greater than the existing value, so the time of the log record is used.

The log entry for the fifth update **E** occurs later in the log for M10 (a higher RBA value) because it was delayed in generating the LRSN value. This situation requires that the last two updates are from different transactions. Otherwise, the sixth update must wait for the fifth update to complete to ensure that the sixth transaction has a later LRSN and a later sequence in the log.

Related tasks

[What to do before RBA or LRSN limits are reached \(Db2 Administration Guide\)](#)

Related information

[Reading log records \(Db2 Administration Guide\)](#)

[Db2 11 for z/OS Technical Overview \(IBM Redbooks\)](#)

Chapter 12. Function that Db2 13 no longer supports

Certain features that were supported in Db2 12 are no longer supported in Db2 13.

Function level or APAR	Removed support	Behavior in Db2 13 and alternative
V13R1M100	Various subsystem parameters are removed in Db2 13.	See Adjust subsystem parameter settings for parameters removed in Db2 13 (Db2 Installation and Migration) .

Deprecated function in Db2 13

Certain capabilities that Db2 13 supports are deprecated, meaning that their use is discouraged. Although they currently remain supported in Db2 13, support is likely to be removed eventually. Avoid creating new dependencies that rely on deprecated function, and develop plans to remove any dependencies on such function.

For more information, see [Chapter 13, “Deprecated function in Db2 13,”](#) on page 65.

Chapter 13. Deprecated function in Db2 13

Certain capabilities that Db2 13 for z/OS supports are *deprecated*, meaning that their use is discouraged. Although they remain supported except as noted below in Db2 13, support is likely to be removed eventually.

Avoid creating new dependencies that rely on deprecated function, and develop plans to remove any dependencies on such function.

Table 1. *Deprecated functions in Db2 13*

Deprecated function	Recommended alternative	Support removed
Basic row format table spaces	Use reordered row format. Starting in Db2 12, any table space that uses basic row format is automatically converted to reordered row format when you run one of the following utilities: <ul style="list-style-type: none">• LOAD REPLACE with the ROWFORMAT RRF option, or LOAD REPLACE without the ROWFORMAT option. The ROWFORMAT option is deprecated and will be removed eventually.• REORG TABLESPACE with the ROWFORMAT RRF option, or REORG TABLESPACE without the ROWFORMAT option. The ROWFORMAT option is deprecated and will be removed eventually.	—
BIND PLAN command MEMBER option	Use BIND PACKAGE commands to bind DBRMs into packages explicitly.	—
COPY utility CHANGELIMIT option	Use the DSNACCOX stored procedure to determine if the object needs to be copied.	—
CHECK_FASTREPLICATION subsystem parameter	Use the default value REQUIRED.	—
DEPLOY option of the BIND PACKAGE command	Deploy native SQL procedures and compiled SQL scalar functions to multiple environments by issuing the same CREATE or ALTER statements separately in each Db2 environment.	—

Table 1. Deprecated functions in Db2 13 (continued)

Deprecated function	Recommended alternative	Support removed
DISALLOW_SEL_INTO_UNION subsystem parameter.	Modify applications to remove any use of UNION or UNION ALL as the outermost from-clause of a SELECT INTO statement. Then set DISALLOW_SEL_INTO_UNION to YES.	—
DSNRLMTxx table format from before Db2 11	Use the current format for resource limit facility (RLF) tables. See Convert RLF tables to the current format (Db2 Installation and Migration) .	—
SYSPROC.DSNTBIND stored procedure	Use the SYSPROC.ADMIN_COMMAND_DSN stored procedure.	—
SYSPROC.DSNUTILS stored procedure	Use the SYSPROC.DSNUTILU stored procedure.	—
DSNTPSMP SQL procedure processor	The SQL procedure processor, DSNTPSMP, is one of several methods that you can use to create and prepare an external SQL procedure, which are also deprecated.	—
SYSPROC.DSNWZP	Use the SYSPROC.ADMIN_INFO_SYSPARM stored procedure.	—
External SQL procedures	Use native SQL procedures, which are more fully supported, easier to maintain, and typically perform better than external SQL procedures. For more information, see Migrating an external SQL procedure to a native SQL procedure (Db2 Application programming and SQL)	—
Hash-organized tables	Alter tables to drop hash organization, and , create indexes to support fast index traversal in Db2 12 or higher. For more information, see Fast index traversal (Db2 Performance) .	Beginning in Db2 12 with application compatibility level V12R1M504, Db2 no longer supports creating hash-organized tables or altering tables to use hash-organization. Existing hash organized tables remain supported. However, that support is likely to be removed in the future.
LOAD utility IDENTITYOVERRIDE option	Use the OVERRIDE(IDENTITY) option.	—
LOAD utility PERIODOVERRIDE option	Use the OVERRIDE(SYSTEMPERIOD) option.	—

Table 1. Deprecated functions in Db2 13 (continued)

Deprecated function	Recommended alternative	Support removed
LOAD utility TRANSIDOVERRIDE option	Use the OVERRIDE(TRANSID) option.	—
NEWFUN SQL processing option	Use SQLLEVEL. NEWFUN is ignored if SQLLEVEL is specified.	—
ODBC 2.0 function	See Deprecated ODBC functions (Db2 Programming for ODBC) .	—
PassTickets for RACF-protected user IDs.	Use client certificate authentication.	—
PREVENT_NEW_IXCTRL_PART subsystem parameter	Use the default value YES.	—
REORG INDEX utility LEAFDISTLIMIT and REPORT only options	Use the DSNACCOX stored procedure to determine whether the object needs to be reorganized	—
REORG INDEX utility UNLOAD ONLY option	Use the UNLOAD utility.	—
REORG INDEX utility UNLOAD PAUSE option	Use the DIAGNOSE utility to stop the process.	—
REORG TABLESPACE utility UNLOAD EXTERNAL option	Use the UNLOAD utility.	—
REORG TABLESPACE utility INDREFLIMIT and REPORTONLY options	Use the DSNACCOX stored procedure to determine whether the object needs to be reorganized.	—
REORG TABLESPACE utility OFFPOSLIMIT and REPORTONLY options	Use the DSNACCOX stored procedure to determine whether the object needs to be reorganized.	—
REORG TABLESPACE utility UNLOAD ONLY option	Use the UNLOAD utility.	—
REORG TABLESPACE utility UNLOAD PAUSE option	Use the UNLOAD utility FORMAT INTERNAL option.	—
REPAIR VERSIONS utility	Use the REPAIR CATALOG utility.	—
SNA communication methods, including the VTAM interface	Use TCP/IP communication only. You can disable SNA communication by specifying a value in the DB2 TCP/IP IPNAME field on panel DSNTIPR.	—
DB2XML.SOAPHTTPPC supplied user-defined function	Use the DB2XML.SOAPHTTPPC user-defined function.	—
DB2XML.SOAPHTTPPV supplied user-defined function	Use the DB2XML.SOAPHTTPPV user-defined function.	—
Synonyms	Use aliases when writing new SQL statements or creating portable applications. Aliases behave the same for the Db2 family of products.	—

Table 1. *Deprecated functions in Db2 13 (continued)*

Deprecated function	Recommended alternative	Support removed
SYSIBM.SYSROUTINES_OPTS catalog table	This catalog table supports the DSNTPSMP SQL procedure processor for creating and preparing external SQL procedures, which are also deprecated.	—
SYSIBM.SYSROUTINES_SRC catalog table	This catalog table supports the DSNTPSMP SQL procedure processor for creating and preparing external SQL procedures, which are also deprecated.	—
Non-UTS table spaces for base tables, including segmented (non-UTS), partitioned (non-UTS), and simple table spaces.	<p>Use partition-by-growth or partition-by-range universal table spaces instead.</p> <p>In Db2 12, Packages bound with APPLCOMPAT(V12R1M504) or higher cannot create objects of the following types:</p> <ul style="list-style-type: none"> • New partitioned (non-UTS) table spaces • New segmented (non-UTS) table spaces • New tables in existing segmented (non-UTS) table spaces • New tables in existing simple table spaces <p>You cannot create new simple table spaces in any supported Db2 release.</p>	—
TEMP database	Db2 uses the work file database instead.	—

Related tasks



[Preparing your system to install or migrate to Db2 13 \(Db2 Installation and Migration\)](#)

Part 3. Adopting new capabilities in Db2 13 continuous delivery

In Db2 13, function levels and application compatibility levels control the adoption of most new capabilities by Db2 subsystems and Db2 applications.

About this task

Function levels are specified by strings that correspond to the Db2 version, release, and maintenance value. The format is *VvvRrMmmm*, where *vv* is the version, *r* is the release, and *mmm* is the modification level. For example, V13R1M501 identifies function level 501. For a list of all available function levels in Db2 13, see [Chapter 2, “Db2 13 function levels,”](#) on page 15. Often function level identifiers are abbreviated. For example, “function level 501” refers to V13R1M501.

Tip:  You can determine the catalog level and function level for a Db2 subsystem or data sharing group, and the code levels of individual subsystems or members, by issuing DISPLAY GROUP commands. For more information, see [Chapter 15, “Determining the Db2 code level, catalog level, and function level,”](#) on page 75 .

Procedure

To manage the adoption of new capabilities in Db2 13, use the following overall process:

1. Apply maintenance to bring the Db2 subsystem to the required code level or higher.

Tip: Apply the maintenance for a code level well before you tailor the catalog level or activate a function level. By doing so, you can verify that Db2 can continue run at the required code level, while you still have the opportunity to identify and remove any problematic maintenance items.

Important: Do not attempt to start Db2 at any code level that is lower than the highest ever activated function level, even at the lower star (*) function level. Activate a function level only after you are satisfied that Db2 can continue to run at the required code level.

2. If necessary, update the Db2 catalog.

You can use a single CATMAINT job that specifies the target function level. If the target function level requires multiple catalog level updates, the CATMAINT job processes each update in sequential order. If a later update in the sequence fails, the previous successful updates do not roll back, and the catalog level remains at the highest level reached. If that occurs, you can correct the reason for the failure and resubmit the same CATMAINT job. Some function levels do not require catalog changes.

Important: Do not attempt to start Db2 at a lower code level after any part of the CATMAINT job for a higher function level completes. Run the CATMAINT job only after you are satisfied that Db2 can continue to run at the necessary code level. The code to tolerate catalog changes is contained in the code level that delivers the CATMAINT job.

3. Activate the higher function level.

Some new capabilities and enhancements become available immediately. Optimization enhancements become available after the next full prepare of the SQL statements. The application compatibility level of each application continues to control the use of new SQL capabilities.

4. When you are ready for applications, and objects such as routines or triggers, to use new SQL capabilities of the higher function level, rebind or alter them at the higher application compatibility level. Do this only after you are satisfied that Db2 13 is stable at the higher function level.

You might need to adjust your applications for incompatible changes before they can run at the higher application compatibility level.

Tip: Do not raise the default application compatibility level of the Db2 subsystem immediately after migrating or activating a new function level. Instead, wait until applications have been verified to work correctly at the higher function level, and any incompatibilities have been resolved. For details,

see [Enabling default application compatibility with function level 500 or higher \(Db2 Application programming and SQL\)](#).

Related concepts

[What's new in Db2 13](#)

Db2 13 for z/OS brings leading-edge innovation to reinforce Db2 for z/OS as a foundation for enterprise computing within the hybrid cloud world.

[Application compatibility levels in Db2 \(Db2 Application programming and SQL\)](#)

Related tasks

[Managing application incompatibilities \(Db2 Application programming and SQL\)](#)

Related reference

[Db2 13 function levels](#)

New Db2 capabilities and enhancements are continuously delivered in a single maintenance stream as the code becomes ready. You can activate the new capabilities in a data sharing group or Db2 subsystem after a function level is delivered. A *function level* corresponds to a single PTF that enables the activation of a specific set of enhancements that shipped in previous prerequisite or co-requisite PTFs. The activation of a function level results in the activation of all lower function levels.

Related information

[PI70406: Add ACTIVATE mode to the Db2 installation CLIST](#)



Chapter 14. Function levels and related levels in Db2 13

Enhancements to Db2 are enabled for use when you activate function levels.

Each *function level* corresponds to a single APAR that enables a set of enhancements that were previously delivered in the service stream. A particular function level might enable one or several enhancements.

Before you can activate a function level, your data sharing group or Db2 subsystem must be at the appropriate catalog level, and every member must be at the minimum required code level.

In most cases, you can activate a higher function level without separately activating each lower function level above the currently activated function level. However, activating a higher function level also activates all capabilities that are introduced by all function levels lower than the one being activated.

Tip:  You can determine the catalog level and function level for a Db2 subsystem or data sharing group, and the code levels of individual subsystems or members, by issuing DISPLAY GROUP commands. For more information, see [Chapter 15, “Determining the Db2 code level, catalog level, and function level,”](#) on page 75 

Function level identifiers

In most cases, code levels, catalog levels, function levels, and application compatibility levels are specified in commands and message output by nine-character strings that correspond to the Db2 version, release, and modification value. However, descriptions of function levels in documentation often refer only to the modification part of the values.

The format is *VvvRrMmmm*, where *vv* is the version, *r* is the release, and *mmm* is the modification level. For example, V13R1M501 identifies function level 501. For a list of all available function levels in Db2 13, see [Chapter 2, “Db2 13 function levels,”](#) on page 15.

Code levels

The *code level* of a Db2 subsystem or data sharing member indicates that the necessary APAR and any prerequisite new function code, defect fixes, and other service items for a corresponding function level are applied. Because new function levels are delivered in the same service stream as other maintenance items, the code level is likely to increase as you routinely apply maintenance to a subsystem or member. If you proactively apply maintenance, you can expect the code level to be higher than the catalog level or function level as you prepare to adopt of new Db2 capabilities.

If you remove maintenance items that support or are otherwise related to a code level, Db2 reverts to a lower code level. However, you cannot start Db2 at a lower code level after tailoring the catalog to a higher catalog level or activating a higher function level. For this reason, it is essential that you tailor the catalog at a higher catalog level or activate a function level only after you are certain that Db2 can continue to run at the corresponding code level.

Tip: Apply the maintenance for a code level well before you tailor the catalog level or activate a function level. By doing so, you can verify that Db2 can continue run at the required code level, while you still have the opportunity to identify and remove any problematic maintenance items.

In a data sharing group, each member can be at a different code level. However a function level is always activated at the group level. That is, if any data sharing member is not at the minimum required code level when you attempt to activate a function level, the ACTIVATE command fails. The DSNU757I message output indicates the current and required code levels.

Each code level is identified by the same identifier as the function level that it enables. The format is *VvvRrMmmm*, where *vv* is the version, *r* is the release, and *mmm* is the modification level.

Important: Do not attempt to start Db2 at any code level that is lower than the highest ever activated function level, even at the lower star (*) function level. Activate a function level only after you are satisfied that Db2 can continue to run at the required code level.

In DISPLAY GROUP command output, the DB2 LVL column indicates the code level of each data sharing member or subsystem in a six-character string that contains the Db2 version, release, and modification values. The format is *vvrmmm*, where *vv* is the version, *r* is the release, and *mm* is the modification level.

Catalog levels

A *catalog level* of a data sharing group or subsystem indicates that a particular CATMAINT utility UPDATE LEVEL job was run on the Db2 catalog, and the data sharing group or subsystem is ready for the activation of certain function levels.

Each function level requires a specific catalog level. However, not every function level requires a new catalog level. If the catalog is not at the minimum required level when you attempt to activate a function level, the ACTIVATE command fails. The message output indicates the current and required catalog levels.

The catalog level is updated when you submit a CATMAINT utility job by tailoring and running the DSNTIJTC sample job. You can use a single CATMAINT job that specifies the target function level. If the target function level requires multiple catalog level updates, the CATMAINT job processes each update in sequential order. If a later update in the sequence fails, the previous successful updates do not roll back, and the catalog level remains at the highest level reached. If that occurs, you can correct the reason for the failure and resubmit the same CATMAINT job.

Whereas different data sharing members can be at different code levels in a data sharing group, a data sharing group has a single catalog level.

Important: Do not attempt to start Db2 at a lower code level after any part of the CATMAINT job for a higher function level completes. Run the CATMAINT job only after you are satisfied that Db2 can continue to run at the necessary code level. The code to tolerate catalog changes is contained in the code level that delivers the CATMAINT job.

Each catalog level is identified by the same identifier as the lowest function level that requires it. The format is *VvvRrMmmm*, where *vv* is the version, *r* is the release, and *mmm* is the modification level. For example function level 501 requires catalog level V13R1M501.

When you first migrate to Db2 13 the catalog level is V13R1M100. The structure of the catalog does not change until you tailor the catalog for function level 501, as described in [Activating Db2 13 function levels 500 and 501 \(Db2 Installation and Migration\)](#).

To find when the catalog level changed, you can check the SYSIBM.SYSLEVELUPDATES catalog table or check for message DSNG014I on the console.

Function levels

A *function level* enables a particular set of new Db2 capabilities and enhancements that were previously delivered in the single continuous stream of Db2 code. It includes code that supports new capabilities, defect fixes, and preventive service items. Before you can use the new capabilities of a function level, you must activate the function level, or a higher function level. Activation of a function level implies activation of the capabilities that are introduced by all lower function levels.

Important: Do not attempt to start Db2 at any code level that is lower than the highest ever activated function level, even at the lower star (*) function level. Activate a function level only after you are satisfied that Db2 can continue to run at the required code level.

In data sharing groups, function levels are activated at the group level. That is, if any data sharing member is not at the minimum required code level when you attempt to activate a function level, the ACTIVATE command fails. The DSNU757I message output indicates the current and required code levels.

To find when the function level changed, you can check the SYSIBM.SYSLEVELUPDATES catalog table or check for message DSNG014I on the console.

Before applications can use new SQL capabilities that a function level introduces, the applications must run at the equivalent application compatibility level or higher.

Application compatibility levels

You can use the *application compatibility* level of applications, and objects such as routines or triggers, to control the adoption and use of new and changed SQL capabilities that are introduced in function levels. Generally, applications, and routines or triggers, cannot use new or changed SQL capabilities unless the effective application compatibility level is equivalent to or higher than the function level that introduced the changes. The application compatibility level applies to most SQL statements, including data definition statements (such as CREATE and ALTER statements) and data control statements (such as GRANT and REVOKE statements).

The corresponding function level or higher must be activated when you bind packages at an application compatibility level. However, if you activate a lower function level (or * function level), applications can continue to run with the higher application compatibility level. To prevent the continued use of SQL capabilities introduced in the higher function level, you must also modify the application and change the effective application compatibility level to the lower level.

For IBM data server clients or drivers that need to exploit Db2 for z/OS capabilities that are delivered with function level V12R1M501 or greater, you need to take additional steps. See [VnnRnMnnn application compatibility levels for data server clients and drivers \(Db2 Application programming and SQL\)](#) for details.

Tip: Do not raise the default application compatibility level of the Db2 subsystem immediately after migrating or activating a new function level. Instead, wait until applications have been verified to work correctly at the higher function level, and any incompatibilities have been resolved. For details, see [Enabling default application compatibility with function level 500 or higher \(Db2 Application programming and SQL\)](#).

Application compatibility levels for Db2 13 are identified by the same identifier as the corresponding function level. The format is *VvvRrMmmm*, where *vv* is the version, *r* is the release, and *mmm* is the modification level.

Db2 13 supports the following application compatibility levels in most contexts:

VvvRrMmmm

Compatibility with the behavior of the identified Db2 function level. For example, V13R1M501 specifies compatibility the highest available function level. The equivalent function level or higher must be activated. For a list of supported Db2 13 function levels, see [Chapter 2, “Db2 13 function levels,”](#) on page 15. Db2 13 also supports values from Db2 12. See [Db2 12 function levels](#).

V12R1

Compatibility with the behavior of Db2 12 function level 500. This value this value has the same result as specifying V12R1M500.

V11R1

Compatibility with the behavior of Db2 11 new-function mode.

V10R1

Compatibility with the behavior of DB2 10 new-function mode.

For more information about application compatibility levels, see [Application compatibility levels in Db2 \(Db2 Application programming and SQL\)](#).

Star (*) function levels

You might activate a lower function level, called a *star (*) function level*, if you encounter problems when you activate a higher function level. Any function level lower than the highest function level that was ever activated is always a star (*) function level.

The output from DISPLAY GROUP and ACTIVATE commands identify star (*) function levels by the function level identifier followed by an asterisk, hence the name. For example, assume that you activate function level 500 after function level 501 was previously activated. V13R1M500* in the message output

indicates that the Db2 data sharing group or subsystem is at function level 500* (you might say, "function level 500 star").

Important: Activating a lower star (*) function level does not enable you to remove maintenance and start Db2 at any code level lower than the catalog level or highest ever activated function level.

Activating a lower star (*) function level by itself does not immediately disable all use of capabilities at higher function levels. Instead, it provides flexibility for limiting or disabling the use of capabilities at higher function levels, while the problems encountered in higher function levels are resolved.

At a star (*) function level, Db2 continues to tolerate objects, packages, and structures that were created or bound at higher function levels. Also, in any context where the effective application compatibility level remains at the higher level, new SQL capabilities from the higher level can still be used. For packages, they can still be executed, rebound, and automatically bound. However, an explicit bind of such packages succeeds only when the APPLCOMPAT bind option is equivalent to the activated star (*) function level or lower. Similar rules apply for the application compatibility levels of native SQL procedures, compiled SQL scalar functions, and advanced triggers. The result is that applications that use capabilities at a higher function level can continue to do so if they are not related to the reason for activating the lower function level. To stop the use of all SQL capabilities at the higher function level, you must also set all effective application compatibility levels at the lower level.

Related concepts

[Application compatibility levels in Db2 \(Db2 Application programming and SQL\)](#)

Related reference

[Db2 13 function levels](#)

New Db2 capabilities and enhancements are continuously delivered in a single maintenance stream as the code becomes ready. You can activate the new capabilities in a data sharing group or Db2 subsystem after a function level is delivered. A *function level* corresponds to a single PTF that enables the activation of a specific set of enhancements that shipped in previous prerequisite or co-requisite PTFs. The activation of a function level results in the activation of all lower function levels.

[-DISPLAY GROUP \(Db2\) \(Db2 Commands\)](#)

Related information

[DSN7100I \(Db2 Messages\)](#)

[DSNG014I \(Db2 Messages\)](#)

[Video: Db2 for z/OS—Delivering New Capabilities Faster \(YouTube: 1:33:25\)](#)

Chapter 15. Determining the Db2 code level, catalog level, and function level

Before you can activate a Db2 function level, you must ensure that the Db2 subsystem or data sharing group is at the appropriate code level and catalog level.

Procedure

GUIP To determine the code level, catalog level, and function level of a Db2 subsystem or data sharing group:

1. Issue a DISPLAY GROUP command.
2. Examine the DISPLAY GROUP output in message DSN7100I.

The DB2 LVL value indicates the code levels of each Db2 subsystem or data sharing member.

Important: Do not attempt to start Db2 at any code level that is lower than the highest ever activated function level, even at the lower star (*) function level. Activate a function level only after you are satisfied that Db2 can continue to run at the required code level.

Examples

The following examples show output from DISPLAY GROUP commands:

```
-DB2A DISPLAY GROUP DETAIL
```

Example: Data sharing group with coexisting Db2 13 and Db2 12 members

In the following example, two members are migrated to Db2 13 and ready for the activation of new function in Db2 13. However, the code level of member DB2B indicates that it is not migrated to Db2 13, which means the group is not ready for activation of function level 500. The DB2 LVL value for DB2B is 121510, which indicates that all Db2 12 function levels are activated.

```
-DISPLAY GROUP DETAIL
DSN7100I  -DB2A DSN7GCMD
*** BEGIN DISPLAY OF GROUP(DSNCAT ) CATALOG LEVEL(V13R1M100)
          CURRENT FUNCTION LEVEL(V13R1M100)
          HIGHEST ACTIVATED FUNCTION LEVEL(V13R1M100)
          HIGHEST POSSIBLE FUNCTION LEVEL(V13R1M100)
          PROTOCOL LEVEL(2)
          GROUP ATTACH NAME(DSNG)
-----
DB2      SUB      DB2      SYSTEM      IRLM
MEMBER  ID   SYS  CMDPREF  STATUS  LVL   NAME  SUBSYS  IRLMPROC
-----
DB2A    1  DB2A  -DB2A    ACTIVE  131501  MVSA  DJ2A   DB2AIRLM
DB2B    2  DB2B  -DB2B    ACTIVE  121510  MVSB  DJ2B   DB2BIRLM
DB2C    3  DB2C  -DB2C    ACTIVE  131501  MVSC  DJ2C   DB2CIRLM
-----
DISPLAY SUBGROUP ATTACH INFORMATION FOR GROUP ATTACH DSNG
-----
SCA  STRUCTURE SIZE: 12288 KB, STATUS= AC,   SCA IN USE: 8 %
LOCK1 STRUCTURE SIZE: 12288 KB
NUMBER LOCK ENTRIES: 1048576
NUMBER LIST ENTRIES: 23073, LIST ENTRIES IN USE: 7
SPT01 INLINE LENGTH: 32138
*** END DISPLAY OF GROUP(DSNCAT )
DSN9022I  -DB2C DSN7GCMD 'DISPLAY GROUP ' NORMAL COMPLETION
```

Example: Running a DISPLAY GROUP command run on a Db2 12 member during coexistence with Db2 13

The following example output shows the result of running the following command on a Db2 12 member in the data sharing group from the previous example:

```
-DB2B DISPLAY GROUP DETAIL
```

Because the command was run on a Db2 12 member, the highest possible function level indicates V12R1M510, which is the highest level that you can activate by issuing an ACTIVATE command. However, the current function level indicates V13R1M100 because at least one member has been migrated to Db2 13.

```
-DISPLAY GROUP DETAIL
DSN7100I -DB2B DSN7GCMD
*** BEGIN DISPLAY OF GROUP(DSNCAT ) CATALOG LEVEL(V13R1M100)
CURRENT FUNCTION LEVEL(V13R1M100)
HIGHEST ACTIVATED FUNCTION LEVEL(V13R1M100)
HIGHEST POSSIBLE FUNCTION LEVEL(V12R1M510)
PROTOCOL LEVEL(2)
GROUP ATTACH NAME(DSNG)
-----
DB2          SUB          DB2  SYSTEM  IRLM
MEMBER      ID   SYS  CMDPREF  STATUS  LVL   NAME  SUBSYS  IRLMPROC
-----
DB2A        1  DB2A -DB2A    ACTIVE  131501 MVSA  DJ2A   DB2AIRLM
DB2B        2  DB2B -DB2B    ACTIVE  121510 MVSB  DJ2B   DB2BIRLM
DB2C        3  DB2C -DB2C    ACTIVE  131501 MVSC  DJ2C   DB2CIRLM
-----

DISPLAY SUBGROUP ATTACH INFORMATION FOR GROUP ATTACH DSNG
-----
SCA  STRUCTURE SIZE:    12288 KB, STATUS= AC,   SCA IN USE:    8 %
LOCK1 STRUCTURE SIZE:    12288 KB
NUMBER LOCK ENTRIES:    1048576
NUMBER LIST ENTRIES:    23073, LIST ENTRIES IN USE:    7
SPT01 INLINE LENGTH:    32138
*** END DISPLAY OF GROUP(DSNCAT )
DSN9022I -DB2B DSN7GCMD 'DISPLAY GROUP ' NORMAL COMPLETION
```

Example: Data sharing group with all active members migrated to Db2 13 code, before the activation of function level 500

The following DISPLAY GROUP output illustrates a data sharing group with all active members migrated to Db2 13 and ready for the activation of function level 500.

```
DSN7100I -DB2B
DSN7GCMD
*** BEGIN DISPLAY OF GROUP(DSNCAT ) CATALOG
LEVEL(V13R1M100)
CURRENT FUNCTION
LEVEL(V13R1M100)
HIGHEST ACTIVATED FUNCTION
LEVEL(V13R1M100)
HIGHEST POSSIBLE FUNCTION
LEVEL(V13R1M500)
PROTOCOL
LEVEL(2)
GROUP ATTACH
NAME(DSNG)
-----
DB2          SUB          DB2  SYSTEM
IRLM        ID   SYS  CMDPREF  STATUS  LVL   NAME  SUBSYS
MEMBER      ID   SYS  CMDPREF  STATUS  LVL   NAME  SUBSYS
IRLMPROC
-----
DB2A        1  DB2A -DB2A    ACTIVE  131501 MVSA  DJ2A
DB2AIRLM
DB2B        2  DB2B -DB2B    ACTIVE  131501 MVSB  DJ2B
DB2BIRLM
DB2C        3  DB2C -DB2C    ACTIVE  131501 MVSC  DJ2C
DB2CIRLM
```

```

-----
DISPLAY SUBGROUP ATTACH INFORMATION FOR GROUP ATTACH
DSNG
-----

SCA  STRUCTURE SIZE:   12288 KB, STATUS= AC,   SCA IN USE:   9
%
LOCK1 STRUCTURE SIZE:   12288
KB
NUMBER LOCK ENTRIES:
1048576
NUMBER LIST ENTRIES:   23073, LIST ENTRIES IN USE:
17
SPT01 INLINE LENGTH:
32138
*** END DISPLAY OF
GROUP(DSNCAT )
DSN9022I -DB2B DSN7GCMD 'DISPLAY GROUP ' NORMAL COMPLETION

```

Example: Data sharing group with all active members migrated to Db2 13 after the activation of function level 500

The following DISPLAY GROUP output illustrates a data sharing group with function level 500 activated.

```

DSN7100I -DB2A DSN7GCMD
*** BEGIN DISPLAY OF GROUP(DSNCAT ) CATALOG LEVEL(V13R1M500)
CURRENT FUNCTION LEVEL(V13R1M500)
HIGHEST ACTIVATED FUNCTION LEVEL(V13R1M500)
HIGHEST POSSIBLE FUNCTION LEVEL(V13R1M500)
PROTOCOL LEVEL(2)
GROUP ATTACH NAME(DSNG)
-----
DB2      SUB
MEMBER  ID  SYS  CMDPREF  STATUS  DB2  SYSTEM  IRLM
          LVL  NAME  SUBSYS  IRLMPROC
-----
DB2A      1  DB2A  -DB2A    ACTIVE  131501  MVSA    DJ2A    DB2AIRLM
DB2B      2  DB2B  -DB2B    ACTIVE  131501  MVSB    DJ2B    DB2BIRLM
DB2C      3  DB2C  -DB2C    ACTIVE  131501  MVSC    DJ2C    DB2CIRLM
-----
DISPLAY SUBGROUP ATTACH INFORMATION FOR GROUP ATTACH DSNG
-----
SCA  STRUCTURE SIZE:   12288 KB, STATUS= AC,   SCA IN USE:   9 %
LOCK1 STRUCTURE SIZE:   12288 KB
NUMBER LOCK ENTRIES:   1048576
NUMBER LIST ENTRIES:   23073, LIST ENTRIES IN USE:   0
SPT01 INLINE LENGTH:   32138
*** END DISPLAY OF GROUP(DSNCAT )
DSN9022I -DB2A DSN7GCMD 'DISPLAY GROUP ' NORMAL COMPLETION

```

Example: Data sharing group with an encryption key label assigned to all members

If the subsystem parameter ENCRYPTION_KEYLABEL is specified for the members of a data sharing group, issue the following command to display the key label:

```
-DISPLAY GROUP DETAIL
```

The output is similar to the following:

```

DSN7100I -DB2C DSN7GCMD
*** BEGIN DISPLAY OF GROUP(DSNCAT ) CATALOG LEVEL(V13R1M500)
CURRENT FUNCTION LEVEL(V13R1M500)
HIGHEST ACTIVATED FUNCTION LEVEL(V13R1M500)
HIGHEST POSSIBLE FUNCTION LEVEL(V13R1M500)
PROTOCOL LEVEL(2)
GROUP ATTACH NAME(DSNG)
ENCRYPTION KEY LABEL (SYSTEM.KEY01)

```

What to do next

1. If necessary, apply maintenance to the Db2 subsystem or data sharing group members for the code level required by the target function level, and repeat this task.
2. Activate the target function level as described in [Chapter 18, “Activating Db2 13 function levels,” on page 83](#).

Related concepts

[Function levels and related levels in Db2 13](#)

Enhancements to Db2 are enabled for use when you activate function levels.

Related reference

[-DISPLAY GROUP \(Db2\) \(Db2 Commands\)](#)

Related information

[DSN7100I \(Db2 Messages\)](#)

Chapter 16. Testing Db2 function level activation


Before you activate a Db2 function level, you can optionally test whether the Db2 subsystem or data sharing group is ready for activation of the target function level.

Before you begin

Important: When you check the readiness of your Db2 environment for a function level, be careful to specify the TEST option with the ACTIVATE command. After any successful completion of the ACTIVATE command without TEST, Db2 must remain at the higher code level. That is, you cannot remove any PTFs that the code level requires, even at a lower star (*) function level. You can also use the DISPLAY GROUP command to determine the highest function level that your Db2 environment supports, without risk of inadvertent function level activation. For more information, see [Chapter 15, “Determining the Db2 code level, catalog level, and function level,”](#) on page 75.

Procedure

To test activation of a Db2 function level, complete the following steps:

1.  Issue an ACTIVATE command with the TEST option and specify the target Db2 function level to test.

```
-ACTIVATE FUNCTION LEVEL (V13R1M500) TEST
```

2. Examine the DSNU757I message.

The DSNU757I message indicates whether the group is ready for the specified level. Because TEST is specified, the output includes detailed information about each active member of the data sharing group. In this example, all of the members are at the required code level and catalog level so that function level 500 can be activated.

```
DSNU757I  -DB2A DSNUGCCA
*** BEGIN ACTIVATE FUNCTION LEVEL (V13R1M500)
          GROUP ELIGIBLE FOR FUNCTION LEVEL (V13R1M500)
          CATALOG LEVEL (V13R1M500)
          CURRENT FUNCTION LEVEL (V13R1M100)
          PREVIOUS HIGHEST FUNCTION LEVEL (V13R1M100)
          HIGHEST POSSIBLE FUNCTION LEVEL (V13R1M500)
```

DB2 MEMBER	ID	CURRENT CODE-LEVEL	CAPABLE FUNCTION LEVELS		STATUS
			LOWEST	HIGHEST	
DB2A	1	V13R1M500	V13R1M100	V13R1M500	ELIGIBLE
DB2B	2	V13R1M500	V13R1M100	V13R1M500	ELIGIBLE
DB2C	3	V13R1M500	V13R1M100	V13R1M500	ELIGIBLE

```
DSN9022I  -DB2A DSNZACMD '-ACTIVATE FUNC' NORMAL COMPLETION
```

 **GUI**

What to do next

1. If necessary, apply maintenance to the Db2 subsystem or data sharing group members for the code level required by the target function level, and repeat this task.
2. Activate the target function level as described in [Chapter 18, “Activating Db2 13 function levels,”](#) on page 83.

Related concepts

[Function levels and related levels in Db2 13](#)

Enhancements to Db2 are enabled for use when you activate function levels.

Related reference

[-ACTIVATE \(Db2\) \(Db2 Commands\)](#)

Related information

[DSNU757I \(Db2 Messages\)](#)

Chapter 17. Identifying applications that are incompatible with catalog updates

Before you activate a Db2 function level, you might need to use the CATMAINT utility to tailor the catalog for the target function level. You can identify applications, activities, and Db2 resources that might be incompatible with catalog migrations and updates and take appropriate actions beforehand to minimize the possibility of a failed catalog update.

Procedure

To identify applications that might interfere with a catalog update, take one of the following actions:

- Issue the DISPLAY BLOCKERS command, which displays locks and claims that active threads hold against specified databases.
You can use the optional DETAIL keyword to receive additional report information about each lock or claim.
- Invoke the BLOCKING_THREADS built-in function, which returns a table that contains one row for each lock or claim that threads hold against specified databases.

What to do next

If incompatible applications exist, take one of the following actions:

- Terminate any active threads that hold locks or claims against the specified databases. For more information, see [Canceling threads \(Db2 Administration Guide\)](#).
- Run the CATMAINT utility during a time window when the incompatible applications are not running.
- Schedule a planned outage for running the CATMAINT utility.

Related reference

[-DISPLAY BLOCKERS \(Db2\) \(Db2 Commands\)](#)

[BLOCKING_THREADS \(Db2 SQL\)](#)

Chapter 18. Activating Db2 13 function levels

You control the activation and adoption of new features in Db2 13 by activating function levels and specifying the application compatibility level. You can also continue to apply corrective and preventative service without adopting new feature function.

Before you begin

Ensure that no incompatible applications will interfere with the catalog update. For details, see [Chapter 17, “Identifying applications that are incompatible with catalog updates,”](#) on page 81.

Determine the function level to activate. In most cases, you can activate a higher function level without separately activating each lower function level above the currently activated function level. However, activating a higher function level also results in the activation of all lower function levels. Before activating a function level, familiarize yourself with the new capabilities and changes that all lower function levels introduce:

- [Chapter 2, “Db2 13 function levels,”](#) on page 15
- [Chapter 3, “Incompatible changes in Db2 13,”](#) on page 35
-
- [Part 2, “Changes to plan for in Db2 13,”](#) on page 33

About this task

The ACTIVATE command controls the activation of new function in Db2. You can tailor jobs for updating the Db2 catalog and activating Db2 function levels by running the Db2 installation CLIST.

Tip: You can also use z/OSMF to automate running the jobs for this task. For more information, see [Chapter 19, “Activating Db2 function levels by using z/OSMF,”](#) on page 87.

Procedure

To activate capabilities and enhancements that are introduced by Db2 function levels, complete the following steps:

1. If Db2 13 is at function level 100 or 500, follow the steps in [Activating Db2 13 function levels 500 and 501 \(Db2 Installation and Migration\)](#).
2. Issue a DISPLAY GROUP command to check that the code level of the Db2 subsystem or each data sharing group member supports your target function level.
In the DSN7100I message, the DB2 LVL column indicates the code level. For more information and examples, see [Chapter 15, “Determining the Db2 code level, catalog level, and function level,”](#) on page 75.
3. If necessary, apply maintenance, such as PTFs and RSUs, to bring your Db2 subsystem or data sharing group members up to the required code level for your target function level.
4. Run the installation CLIST, as described in [Tailoring Db2 13 installation and migration jobs with the CLIST \(Db2 Installation and Migration\)](#).
 - a) On panel DSNTIPA1, specify values in the ACTIVATE, INPUT MEMBER, and OUTPUT MEMBER fields.
 - In the INSTALL TYPE field, specify ACTIVATE.
 - In the INPUT MEMBER field, specify the name of the CLIST output member that you created when you installed or migrated to Db2 13, or most recently activated a Db2 13 function level.
 - In the OUTPUT MEMBER field, specify a new member name, to save your changes for future use.
 - b) On panel DSNTIPT, verify the SAMPLE LIBRARY field value, which is the name of the output data set that is to be created. An asterisk appears at the far left of this field if the data set already exists. If

the data set already exists, the CLIST replaces the members that it customizes for activation of the new function level.

- c) On panel DSNTIP00, specify the target function level in the TARGET FUNCTION LEVEL field.
The format is *VvvRrMmmm*, where *vv* is the version, *r* is the release, and *mmm* is the modification level.
The value is used in the ACTIVATE command in the DSNTIJAF job and in the CATMAINT utility control statement in the DSNTIJTC job.
- d) Optional: If all Db2 applications can be bound and run at the target function level, modify the APPLCOMPAT and SQLLEVEL subsystem parameter settings. Otherwise, leave these fields unchanged.
- e) Proceed through the remaining panels, and wait for the CLIST to tailor the jobs for the activation process.
The output data set contains the tailored jobs for the activation process.

5. Run the DSNTIJIC job to take an image copy of the Db2 catalog and directory.

6. Run the DSNTIJTC job to run the CATMAINT utility to tailor the Db2 catalog for the target function level.

You can use a single CATMAINT job that specifies the target function level. If the target function level requires multiple catalog level updates, the CATMAINT job processes each update in sequential order. If a later update in the sequence fails, the previous successful updates do not roll back, and the catalog level remains at the highest level reached. If that occurs, you can correct the reason for the failure and resubmit the same CATMAINT job.

Important: Do not attempt to start Db2 at a lower code level after any part of the CATMAINT job for a higher function level completes. Run the CATMAINT job only after you are satisfied that Db2 can continue to run at the necessary code level. The code to tolerate catalog changes is contained in the code level that delivers the CATMAINT job.

When the catalog level change completes, Db2 records the change in the SYSIBM.SYSLEVELUDPATES catalog table and issues message DSN014I to the console.

7. Optional: Test the Db2 code level and catalog level for readiness to activate of the function level.

For more information and examples, see [Chapter 16, “Testing Db2 function level activation,” on page 79](#).

8. Run the DSNTIJAF job to issue an ACTIVATE command for the target function level.

Important: Do not attempt to start Db2 at any code level that is lower than the highest ever activated function level, even at the lower star (*) function level. Activate a function level only after you are satisfied that Db2 can continue to run at the required code level.

If the command completes successfully, Db2 issues message DSN9022I. Message DSNU757I indicates the result of the activate command.

GUI For example, the following message indicates successful activation:

```
DSNU757I  -DB2A DSNUGCCA
*** BEGIN ACTIVATE FUNCTION LEVEL (V13R1M500)
          FUNCTION LEVEL (V13R1M500) SUCCESSFULLY ACTIVATED
          CATALOG LEVEL (V13R1M100)
          CURRENT FUNCTION LEVEL (V13R1M500)
          PREVIOUS HIGHEST FUNCTION LEVEL (V13R1M100)
          HIGHEST POSSIBLE FUNCTION LEVEL (V13R1M500)
DSN9022I  -DB2A DSNZACMD '-ACTIVATE FUNC' NORMAL COMPLETION
```

When the function level change completes, Db2 records the change in the SYSIBM.SYSLEVELUDPATES catalog table and issues message DSN014I to the console.

GUI

More actions are required after the ACTIVATE command completes successfully before most types of new capabilities and enhancements in the function level can be used. For example, new

SQL capabilities require that applications use the appropriate application compatibility level, and optimization enhancements apply only after full prepare of the SQL statements.

9. After you are ready for applications to use the new capabilities in the function level, rebind them at the corresponding application compatibility level. For more information, see [Chapter 21, “Controlling Db2 application compatibility,”](#) on page 91.

Tip: Proceed with the following steps only if all Db2 applications can be bound and run at the target function level.

Optionally, when you are ready for all applications to use the new capabilities of the target function level, you can run the following jobs:

- a. Run DSNTIJUZ to modify the subsystem parameter module with the APPLCOMPAT value that was specified on panel DSNTIP00.
- b. Run DSNTIJOZ job to issue SET SYSPARM command to bring the APPLCOMPAT subsystem parameter changes online.
- c. Run DSNTIJUA job to modify the Db2 data-only application defaults module with the SQLLEVEL value that was specified on panel DSNITP00.

What to do next

Complete any of the following actions:

- If the new function level includes optimization enhancements, Db2 must process a full prepare before any SQL statements can benefit. Whether a full prepare occurs depends on the statement type:
 - For static SQL statements, after bind or rebind of the package.
 - For non-stabilized dynamic SQL statements, immediately, unless the statement is in the dynamic statement cache.
 - For stabilized dynamic SQL statements, after invalidation, free, or changed application compatibility level.
- If you did not run the jobs to update the APPLCOMPAT and SQLLEVEL subsystem parameters, resolve any application incompatibilities and increase the application compatibility level of your applications after you are satisfied that Db2 is stable at the target function level, as described in [Chapter 21, “Controlling Db2 application compatibility,”](#) on page 91.
- If you encounter regressions or other problems when you activate a Db2 13 function level, minimize the impact to your applications while you resolve the problems by following the general approaches described in [Chapter 22, “Responding to problems after function level activation,”](#) on page 93.

Related concepts

[Function levels and related levels in Db2 13](#)

Enhancements to Db2 are enabled for use when you activate function levels.

[Application compatibility levels in Db2 \(Db2 Application programming and SQL\)](#)

Related reference

[Db2 13 function levels](#)

New Db2 capabilities and enhancements are continuously delivered in a single maintenance stream as the code becomes ready. You can activate the new capabilities in a data sharing group or Db2 subsystem after a function level is delivered. A *function level* corresponds to a single PTF that enables the activation of a specific set of enhancements that shipped in previous prerequisite or co-requisite PTFs. The activation of a function level results in the activation of all lower function levels.

[-ACTIVATE \(Db2\) \(Db2 Commands\)](#)

[CATMAINT \(Db2 Utilities\)](#)

Related information

[DSNU757I \(Db2 Messages\)](#)

[DSNG014I \(Db2 Messages\)](#)

Chapter 19. Activating Db2 function levels by using z/OSMF

You can use z/OSMF workflows to automate the running of jobs for activating new capabilities in Db2 function levels.

Before you begin

Important: Before you activate function level 500 or higher for the first time, read [Activating Db2 13 function levels 500 and 501 \(Db2 Installation and Migration\)](#).

Ensure that no incompatible applications will interfere with the catalog update. For details, see [Chapter 17, “Identifying applications that are incompatible with catalog updates,”](#) on page 81.

Determine the function level to activate. In most cases, you can activate a higher function level without separately activating each lower function level above the currently activated function level. However, activating a higher function level also results in the activation of all lower function levels. Before activating a function level, familiarize yourself with the new capabilities and changes that all lower function levels introduce:

- [Chapter 2, “Db2 13 function levels,”](#) on page 15
- [Chapter 3, “Incompatible changes in Db2 13,”](#) on page 35
-
- [Part 2, “Changes to plan for in Db2 13,”](#) on page 33

About this task

You can run the jobs for activating new capabilities in Db2 function levels by using z/OSMF workflows.

Procedure

To activate a Db2 function level by using z/OSMF, complete the following steps:

1. Issue a DISPLAY GROUP command to check that the code level of the Db2 subsystem or each data sharing group member supports your target function level.
In the DSN7100I message, the DB2 LVL column indicates the code level. For more information and examples, see [Chapter 15, “Determining the Db2 code level, catalog level, and function level,”](#) on page 75.
2. If necessary, apply maintenance, such as PTFs and RSUs, to bring your Db2 subsystem or data sharing group members up to the required code level for your target function level.
3. Run the installation CLIST, as described in [Tailoring Db2 13 installation and migration jobs with the CLIST \(Db2 Installation and Migration\)](#).
 - a) On panel DSNTIPA1, specify YES in the USE Z/OSMF WORKFLOW field.
 - b) On panel DSNTIPA1, specify ACTIVATE in the INSTALL TYPE field.
 - c) On panel DSNTIP00, specify the target function level in the TARGET FUNCTION LEVEL field.
The format is *VvvRrMmmm*, where *vv* is the version, *r* is the release, and *mmm* is the modification level.
The value is used in the ACTIVATE command in the DSNTIJAF job and the CATMAINT utility control statement in the DSNTIJTC job.
4. In z/OSMF, run the DSNTIWAF workflow.
The DSNTIWAF workflow always runs the DSNTIJAF job to run an ACTIVATE command for the target function level. It runs various other jobs depending on the specific situation.

Related tasks

[Automating Db2 migration by using z/OS Management Facility \(Db2 Installation and Migration\)](#)

Related reference

[-ACTIVATE \(Db2\) \(Db2 Commands\)](#)

[CATMAINT \(Db2 Utilities\)](#)

Job DSNTIJOZ: bring Db2 subsystem parameter changes online

Job DSNTIJOZ runs a SET SYSPARM command to bring subsystem parameter changes online.

The z/OSMF workflows for migrating to Db2 13 and activating Db2 function levels use this job.

Related tasks

[Activating Db2 function levels by using z/OSMF](#)

You can use z/OSMF workflows to automate the running of jobs for activating new capabilities in Db2 function levels.

[Automating Db2 migration by using z/OS Management Facility \(Db2 Installation and Migration\)](#)

Related reference

[-SET SYSPARM \(Db2\) \(Db2 Commands\)](#)

Chapter 20. Updating Db2 initialization parameters for function level activation

After you activate a function level, you can enable applications to begin using new capabilities by default by updating the APPLCOMPAT subsystem parameter and the SQLLEVEL value in the application defaults module.

About this task

Tip: Do not raise the default application compatibility level of the Db2 subsystem immediately after migrating or activating a new function level. Instead, wait until applications have been verified to work correctly at the higher function level, and any incompatibilities have been resolved. For details, see [Enabling default application compatibility with function level 500 or higher \(Db2 Application programming and SQL\)](#).

Procedure

To update initialization parameters for function level activation, complete the following steps:

1. Run job DSNTIJUZ.

Job DSNTIJUZ defines the Db2 data-only subsystem parameter module (DSNZPxxx), which consists of the expansion of the following macros: DSN6ARVP, DSN6FAC, DSN6GRP, DSN6LOGP, DSN6SPRM, and DSN6SYSP.

You might need to make the following adjustments before running the job:

- If you added a STEPLIB DD statement to the Db2 start procedures ahead of *prefix*.SDSNEXIT and *prefix*.SDSNLOAD, you can move the SYSLMOD output to that library.
- If you changed the prefix for the Db2 distribution libraries, edit DSNTIJUZ to correct the data set names.
- If you have not run the SMP/E ACCEPT job (DSNACEP1) of FMID HDBDD10, edit DSNTIJUZ so that the SMP/E temporary data set (SMPTLIB) is included in the concatenation for the ADSNLOAD DD statement in step DSNTIZL. This action ensures that members DSNFSYSP, DSNJARVP, DSNJLOGP, DSNTSPRM, DSNVDIR1, DSNZMSTR, and DSN3DIR1 are linked with the subsystem parameter load module. SMPTLIB is *hlq*.HDBCC10.F2, where *hlq* is from the GLOBAL SMP/E zone. Use the following SMP/E statements to get DSPREFIX:

```
SET BOUNDARY (GLOBAL).  
LIST DDDEF ( SMPTLIB ).
```

Insert the DSPREFIX value after SDSNLOAD and ADSNLOAD.

When DSNTIJUZ completes, the DSNTINST CLIST performs calculations by using the values that you specified for some of the parameter values that you entered on the panels. These calculations appear in the macro descriptions.

For more information, see [Job DSNTIJUZ: define the Db2 data-only subsystem parameter module \(Db2 Installation and Migration\)](#).

2. Run job DSNTIJUA.

Job DSNTIJUA defines the Db2 data-only application defaults module.

You might need to make the following adjustments before running the job:

- If you added a STEPLIB DD statement to the Db2 start procedures ahead of *prefix*.SDSNEXIT and *prefix*.SDSNLOAD, you can move the SYSLMOD output to that library.
- If you changed the prefix for the Db2 distribution libraries, edit DSNTIJUA to correct the data set names.

- If you have not run the SMP/E ACCEPT job (DSNACEP1) of FMID HDBDD10, edit DSNTIJUA so that the SMP/E temporary data set (SMPTLIB) is included in the concatenation for the ADSNLOAD DD statement in step DSNTIZQ. This action ensures that member DSNARIB is linked with the application defaults module. SMPTLIB is *hlq*.HDBCC10.F2, where *hlq* is from the GLOBAL SMP/E zone. Use the following SMP/E statements to get DSPREFIX:

```
SET    BOUNDARY (GLOBAL).  
LIST  DDDEF ( SMPTLIB ).
```

Insert the DSPREFIX value after SDSNLOAD and ADSNLOAD.

For more information, see [Job DSNTIJUA: define data-only application defaults module \(Db2 Installation and Migration\)](#).

Chapter 21. Controlling Db2 application compatibility

You can use the *application compatibility level* of your applications to control the adoption of new capabilities and enhancements, and the impact of incompatible changes. The result is that you can separate the Db2 13 migration process, and the activation of Db2 13 function levels, from your application updates for adoption of new function and resolution of incompatibilities.

Before you begin

Activate the function level that introduces the new SQL capabilities that your applications will use. For details, see [Part 3, “Adopting new capabilities in Db2 13 continuous delivery,” on page 69](#).

About this task

You can change the application compatibility level for each application when you are ready for it to run with the features and behavior of a higher Db2 version or function level. The application compatibility level applies to all SQL statements, including data definition statements.

After function level 500 or higher is activated, you can continue run applications with the features and behavior of previous versions or specific Db2 13 function levels.

Procedure

To control the adoption of new SQL capabilities, and the impact of incompatible changes on your Db2 applications, and objects such as routines and triggers, use the following process:

1. After the activation of a new function level (including function level 500 after migration to Db2 13), continue to run your applications at the same application compatibility level until you are satisfied that your Db2 13 environment is stable at the new function level.
2. Rebind the packages for SPUFI and the productivity-aid sample programs (such as DSNTDP2, DSNTDP4, DSNTIAD, and DSNTIAUL) at the higher application compatibility level, so that database administrators can begin using the new SQL data definition capabilities.
For more information about preparing the sample programs, see [Db2 productivity-aid sample programs \(Db2 Application programming and SQL\)](#).
3. Identify the highest priority applications for the use of new capabilities. Then, identify and resolve any incompatibilities, as described in [Managing application incompatibilities \(Db2 Application programming and SQL\)](#).
4. Bind or rebind your high-priority applications at the higher application compatibility level.

For best results, apply the following approaches when you complete this step:

- Rebind packages for static SQL applications. Specify use of the PLANMGMT bind option so that you can revert to a previous package copy if a regression occurs.
 - Rebind packages for dynamic SQL applications. The application compatibility level is also among the matching criteria for both cached and stabilized dynamic statements. When it changes, cached dynamic statements exit the cache and require a full prepare at the next execution, and stabilized dynamic SQL statements are no longer stabilized and subject to full prepare and access path change. It is best to re-stabilize such statements only after you are satisfied that no access path regression has occurred.
 - Rebind distributed packages in separate collections and switch the applications to using the new collections.
5. Repeat the two preceding steps for any applications, and any objects such as routines or triggers, that require the use of new SQL capabilities.
 6. After incompatible changes are resolved for most applications, rebind any remaining applications that must continue to run compatibly with the lower level, and explicitly specify the lower application compatibility level.

7. After all applications are either ready to run at the higher level or explicitly bound at the lower level, increase the default application compatibility level, as described in [Enabling default application compatibility with function level 500 or higher \(Db2 Application programming and SQL\)](#).

Related concepts

[Application compatibility levels in Db2 \(Db2 Application programming and SQL\)](#)

[Function levels and related levels in Db2 13](#)

Enhancements to Db2 are enabled for use when you activate function levels.

Related tasks

[Responding to problems after function level activation](#)

If you encounter regression or other problems after the activation of a new Db2 13 function level, you can take certain actions to minimize the impact to your applications while you resolve the underlying problems.

[VnnRnMnnn application compatibility levels for data server clients and drivers \(Db2 Application programming and SQL\)](#)

Chapter 22. Responding to problems after function level activation

If you encounter regression or other problems after the activation of a new Db2 13 function level, you can take certain actions to minimize the impact to your applications while you resolve the underlying problems.

About this task

You might activate a lower function level, called a *star (*) function level*, if you encounter problems when you activate a higher function level. Any function level lower than the highest function level that was ever activated is always a star (*) function level.

The output from DISPLAY GROUP and ACTIVATE commands identify star (*) function levels by the function level identifier followed by an asterisk, hence the name. For example, assume that you activate function level 500 after function level 501 was previously activated. V13R1M500* in the message output indicates that the Db2 data sharing group or subsystem is at function level 500* (you might say, "function level 500 star").

Important: Activating a lower star (*) function level does not enable you to remove maintenance and start Db2 at any code level lower than the catalog level or highest ever activated function level.

Activating a lower star (*) function level by itself does not immediately disable all use of capabilities at higher function levels. Instead, it provides flexibility for limiting or disabling the use of capabilities at higher function levels, while the problems encountered in higher function levels are resolved.

At a star (*) function level, Db2 continues to tolerate objects, packages, and structures that were created or bound at higher function levels. Also, in any context where the effective application compatibility level remains at the higher level, new SQL capabilities from the higher level can still be used. For packages, they can still be executed, rebound, and automatically bound. However, an explicit bind of such packages succeeds only when the APPLCOMPAT bind option is equivalent to the activated star (*) function level or lower. Similar rules apply for the application compatibility levels of native SQL procedures, compiled SQL scalar functions, and advanced triggers. The result is that applications that use capabilities at a higher function level can continue to do so if they are not related to the reason for activating the lower function level. To stop the use of all SQL capabilities at the higher function level, you must also set all effective application compatibility levels at the lower level.

Important: Do not attempt to start Db2 at any code level that is lower than the highest ever activated function level, even at the lower star (*) function level. Activate a function level only after you are satisfied that Db2 can continue to run at the required code level.

Important: Do not attempt to start Db2 at a lower code level after any part of the CATMAINT job for a higher function level completes. Run the CATMAINT job only after you are satisfied that Db2 can continue to run at the necessary code level. The code to tolerate catalog changes is contained in the code level that delivers the CATMAINT job.

Procedure

Use the following general approaches to minimize the impact of problems at a new function level, while you resolve the underlying issues:

1. If issues occur after you rebind packages at a higher application compatibility level, do not immediately revert to a lower star (*) function level. Instead, use REBIND SWITCH(PREVIOUS) to revert to the previous package.
This option is available only if you used PLANMGMT at the previous bind or rebind of the package.
2. If necessary, issue an ACTIVATE command or tailor and run job DNSTIJAF to activate the lower star (*) function level.

Activating the star (*) function level does not prevent the use of new SQL capabilities. You might continue to run applications not causing or related to the regression or problem at the higher application compatibility level. Such application can continue to use capabilities of the higher function level, unless you bind them at the lower application compatibility level.

3. If you must prevent all use of capabilities at the higher function level, bind all packages at the application compatibility level that corresponds to the star (*) function level.
4. If the default application compatibility level was set at the higher function level than the star (*) function level, reduce the default application compatibility level to that lower level to prevent bind failures.

For instructions for setting the default application compatibility, see [Enabling default application compatibility with function level 500 or higher \(Db2 Application programming and SQL\)](#).

Related tasks

Controlling Db2 application compatibility

You can use the *application compatibility level* of your applications to control the adoption of new capabilities and enhancements, and the impact of incompatible changes. The result is that you can separate the Db2 13 migration process, and the activation of Db2 13 function levels, from your application updates for adoption of new function and resolution of incompatibilities.

Chapter 23. How Db2 function levels are documented

Generally, Db2 for z/OS documentation assumes that the highest available function level is activated, and that your applications run with the equivalent application compatibility level in effect. However, new and changed content are marked for changes introduced by function levels.

[FL 500](#) Throughout the Db2 13 information, new and changed content for function levels are marked like this paragraph, with a link to the function level that introduced the changes. You can click the link to see the overview page for the function level. If an entire topic is new, you'll find a single function level overview page link near the beginning of the new topic.

In reference information, syntax diagrams always reflect the highest available function level, and syntax diagrams never include internal revision marks. The associated option descriptions also reflect the highest available function level. However, they are marked with revision marks and links to the function level overview page. In cases where reference materials continue to describe the behavior of previous function levels or application compatibility levels, the differences are generally described in the "Notes" section at the end of the topic.

Related tasks

[Adopting new capabilities in Db2 13 continuous delivery](#)

In Db2 13, function levels and application compatibility levels control the adoption of most new capabilities by Db2 subsystems and Db2 applications.

Related reference

[Db2 13 function levels](#)

New Db2 capabilities and enhancements are continuously delivered in a single maintenance stream as the code becomes ready. You can activate the new capabilities in a data sharing group or Db2 subsystem after a function level is delivered. A *function level* corresponds to a single PTF that enables the activation of a specific set of enhancements that shipped in previous prerequisite or co-requisite PTFs. The activation of a function level results in the activation of all lower function levels.

Information resources for Db2 for z/OS and related products

You can find the online product documentation for Db2 for z/OS in IBM Documentation.

IBM Documentation is the home of all online product documentation for Db2 for z/OS and related products, including [PDF format manuals](#).

Notices

This information was developed for products and services offered in the US. This material might be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

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, MD-NC119
Armonk, NY 10504-1785 US*

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

*Intellectual Property Licensing Legal and Intellectual Property Law IBM Japan Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan*

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 jurisdictions 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.

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 Director of Licensing IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785 US*

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 document 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.

The performance data and client examples cited are presented for illustrative purposes only. Actual performance results may vary depending on specific configurations and operating conditions.

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



Programming interface information

This information is intended to help you to learn about and plan to use Db2 13 for z/OS. This information also documents General-use Programming Interface and Associated Guidance Information and Product-sensitive Programming Interface and Associated Guidance Information provided by Db2 13 for z/OS.

General-use Programming Interface and Associated Guidance Information

General-use Programming Interfaces allow the customer to write programs that obtain the services of Db2 13 for z/OS.

General-use Programming Interface and Associated Guidance Information is identified where it occurs by the following markings:

 General-use Programming Interface and Associated Guidance Information... 

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 may 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 markings:

 Product-sensitive Programming Interface and Associated Guidance Information... 

Trademarks

IBM, the IBM logo, and [ibm.com](http://www.ibm.com)[®] are trademarks or registered marks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at "Copyright and trademark information" at: <http://www.ibm.com/legal/copytrade.shtml>.

Linux[®] is a registered trademark of Linus Torvalds 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.

Java[™] and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Terms and conditions for product documentation

Permissions for the use of these publications are granted subject to the following terms and conditions:

Applicability: These terms and conditions are in addition to any terms of use for the IBM website.

Personal use: You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of IBM.

Commercial use: You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

Rights: Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

Privacy policy considerations

IBM Software products, including software as a service solutions, (“Software Offerings”) may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user, or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering’s use of cookies is set forth below.

This Software Offering does not use cookies or other technologies to collect personally identifiable information.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, see IBM’s Privacy Policy at <http://www.ibm.com/privacy> and IBM’s Online Privacy Statement at <http://www.ibm.com/privacy/details> the section entitled “Cookies, Web Beacons and Other Technologies” and the “IBM Software Products and Software-as-a-Service Privacy Statement” at <http://www.ibm.com/software/info/product-privacy>.

Glossary

The glossary is available in IBM Documentation

For definitions of Db2 for z/OS terms, see [Db2 glossary \(Db2 Glossary\)](#).

Index

Numerics

- 10-byte format
 - extended format
 - RBA and LRSN [59](#)
 - RBA and LRSN [59](#)

C

- catalog levels
 - determining [75](#)
- code levels
 - determining [75](#)
- commands
 - changes [39](#)

D

- Db2 13
 - overview of new function [3](#)
- Db2 commands
 - DISPLAY GROUP
 - examples [75](#)
- Db2function levels
 - determining [75](#)
- DISPLAY GROUP
 - command examples [75](#)

E

- enhancements by function level [15](#), [71](#), [83](#)

F

- function level 500 [19](#)
- function level activation
 - incompatible applications [81](#)
- function levels
 - determining [75](#)

G

- general-use programming information, described [100](#)
- GUPI symbols [100](#)

I

- IFCID (instrumentation facility component identifier)
 - changed [51](#)
 - new [51](#)
- incompatibilities of releases [35](#)
- incompatible applications
 - identifying [81](#)

L

- links
 - non-IBM Web sites [100](#)
- LRSN
 - extended [59](#)

M

- migration
 - considerations [35](#)

O

- overview
 - Db2 13 new function [3](#)

P

- product-sensitive programming information, described [100](#)
- programming interface information, described [100](#)
- PSPI symbols [100](#)

R

- RBA
 - extended [59](#)
- release incompatibilities [35](#)

S

- SQL statements
 - changes [41](#)

U

- utilities
 - changes [45](#)



Product Number: 5698-DB2
5698-DBV

GC28-2784-00

