Db2 12 for z/OS

What's New?

IBM
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 Db2® 12 for z/OS® Product Documentation.

2019-10-14 edition
This edition applies to Db2 12 for z/OS (product number 5650-DB2), Db2 12 for z/OS Value Unit Edition (product number 5770-AF3), 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 2017.
US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.
## Contents

**About this information**
- Terminology and citations ......................................................... ix
- How to send your comments ....................................................... x

**Part 1. What's new in the Db2 12 base release**

<table>
<thead>
<tr>
<th>Chapter 1. Enhancements in recent Db2 12 APARs (function levels 100 or 500)</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>REPAIR utility enhancement to allow write of Db2 diagnostic log record (PH11871)</td>
<td>3</td>
</tr>
<tr>
<td>Hybrid Transactional Analytical Processing support for accelerated static queries (PH14116)</td>
<td>3</td>
</tr>
<tr>
<td>Validation for default buffer pools in the installation CLIST (PH12117)</td>
<td>3</td>
</tr>
<tr>
<td>Console message for inoperative packages (PH13550)</td>
<td>4</td>
</tr>
<tr>
<td>Ability to control workload balancing of accelerated queries based on accelerator version (PH05477)</td>
<td>4</td>
</tr>
<tr>
<td>New job DSNTIJPM premigration report 24: active log data sets larger than 4 GB (PH11514)</td>
<td>4</td>
</tr>
<tr>
<td>New job DSNTIJPM premigration report 23: SQL routines created in DB2 9 (PH07660)</td>
<td>4</td>
</tr>
</tbody>
</table>

### Statistics profile options included in utility output (APAR PH11423)
- Creating artifacts for provisioning Db2 subsystem with the CLIST (APAR PH09857) | 5 |
- Allowing the PORT and SECPORT to be the same for static location aliases (APAR PH08188) | 5 |
- Setting the client application compatibility property is optional when the client is Db2 for Linux, UNIX, and Windows Version 11.1 Fix pack 1 or later (APAR PH08482) | 5 |
- Ability to delete only FlashCopy image copies (APAR PH04023) | 6 |
- Loading multiple input data sets in a single job (APAR PI96136) | 6 |
- IBM Db2 AI for z/OS 1.2.0 (APAR PH07782) | 6 |
- Improve RUNSTATS performance by making page sampling the default behavior (APAR PH07220) | 6 |
- New environment variable to control the MQListener log size (APAR PH07230) | 7 |
- Support for sysplex group authentication (APAR PI94236) | 7 |
- Enhancements to insert algorithm 2 (APAR PH02052) | 7 |
- New in-memory resource limit table automatic refresh capability and enhancements to START RLIMIT (APAR PH06082) | 8 |
- Db2 exploitation of asynchronous cross-invalidation for coupling facility cache structures (PH05193) | 8 |
- Specify the default padding behavior for REORG TABLESPACE with UNLOAD EXTERNAL or DISCARDING processing (APAR PH00317) | 8 |
- IBM Db2 AI for z/OS 1.1.0.1 (APAR PH05323) | 9 |
- Avoid external sorts for single-column column groups at the utility job level (APAR PH03678) | 9 |
- New installation jobs for some DSNTIJMV job tasks (APAR PH02971) | 9 |
- Allow ROTATE PARTITION for materialized query tables or tables having dependent materialized query tables (APAR PH00194) | 10 |
- New alternate format for timestamp values (APAR PH03263) | 10 |
- Db2 command line processor (CLP) support for user name and credentials for running SQL files (APAR PH00335) | 11 |
- UNLOAD utility uses REGISTER NO by default in Db2 12 (APAR PI99075) | 11 |
- Wide (suffix-W) API for SQLGetDiagRec() to support Unicode wide character type (APAR PH05953) | 11 |
- Specify a DDF IPNAME (PI99403) | 12 |
- Specify an HFS or zFS directory for z/OSMF artifacts (PI97635) | 12 |
- User-defined function with MODIFIES SQL DATA from a subselect (PI93887) | 12 |
- New console message for remote client information (enabled by APAR PI89903) | 13 |
- HFS DRBM support for COBOL and PL/I (enabled by APAR PI88171) | 13 |
Chapter 2. Application enablement in the Db2 12 initial release ............................................................ 23
Improved availability when altering index compression .................................................................... 35
Support for preserve mirror options for recovery ............................................................................... 34
Support for maintaining session data on the target server................................................................. 29
More flexibility in defining application periods for temporal tables................................................... 27
Support for temporal logical transactions............................................................................................ 27
PERCENTILE function support............................................................................................................ 28
DRDA fast load.................................................................................................................................. 28
ODBC enhancements.......................................................................................................................... 28
Obfuscated source code for SQL routines and triggers ...................................................................... 29
Support for maintaining session data on the target server................................................................. 29
Resource limits for static SQL statements............................................................................................ 29
Db2 REST services improve efficiency and security.............................................................................. 30
Modern language support Db2 for z/OS application development..................................................... 30
DevOps with Db2: Automated deployment of applications with IBM UrbanCode Deploy................. 31

Chapter 3. Administrator function in the Db2 12 initial release ............................................................ 33
Single-phase catalog migration in Db2 12 ............................................................................................ 33
Installation or migration without requiring SYSAADM........................................................................ 34
Support for preserve mirror options for recovery.............................................................................. 34
Improved availability when altering index compression....................................................................... 35
Part 2. What's new in Db2 12 function levels........................................................ 59

Chapter 4. Query performance in the Db2 12 initial release........................................ 49
  Automated statistics collection.................................................................................. 49
  Static plan stability enhancements.......................................................................... 50
  Query performance enhancements......................................................................... 50
  User-defined table function performance improvements...................................... 50
  Improved performance and reliability of index access with list prefetch........... 51

Chapter 5. OLTP performance in the Db2 12 initial release........................................ 53
  Improved INSERT performance for unclustered data................................................. 53
  Improved EDM pool management.......................................................................... 53
  In-memory buffer pools.......................................................................................... 54
  Improvements for the dynamic statement cache..................................................... 54
  Improved random index access............................................................................. 55
  Improved lock avoidance checking for data sharing environments........................ 55
  Optimized scheduling of dynamic prefetch.............................................................. 55
  Reduced log write latency....................................................................................... 55
  Support for estimating benefits of increasing buffer pool size............................ 56
  Scalability enhancements...................................................................................... 56
  Increased size of active log data sets....................................................................... 56
  Instrumentation enhancements............................................................................. 56
  Integration with IBM DS8870 Easy Tier Multi-Temperature Management........ 57

Part 2. What's new in Db2 12 function levels........................................................ 59

Chapter 6. Function level 505 (activation enabled by APAR PH09191—June, 2019)........ 61

Chapter 7. Function level 504 (activation enabled by APAR PH07672—April, 2019)........ 65

Chapter 8. Function level 503 (activation enabled by APAR PH00506—October, 2018)...... 71

Chapter 9. Function level 502 (activation enabled by APAR PI95511—May, 2018).......... 75

Chapter 10. Function level 501 (activation enabled by APAR PI70535—May, 2017)......... 79
About this information

This information provides an executive overview of new function in Db2 12 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 12 for z/OS”. References to other Db2 products use complete names or specific abbreviations.

Important: To find the most up to date content, always use IBM® Knowledge Center, which is continually updated as soon as changes are ready. PDF manuals are updated only when new editions are published, on an infrequent basis.

How Db2 function levels are documented:

Generally, this information assumes that the highest available function level is activated, and that your applications run with the equivalent application compatibility value. However, changes for function levels are marked in the documentation. For more information, see Chapter 34, “How Db2 function levels are documented,” on page 191.

Overview of Db2 12 new function availability

The availability of new function depends on the type of enhancement, the activated function level, and the application compatibility levels of applications. In the initial Db2 12 release, most new capabilities are enabled only after the activation of function level 500 or higher.

Virtual storage enhancements

Virtual storage enhancements become available at the activation of the function level that introduces them or higher. Activation of function level 100 introduces all virtual storage enhancements in the initial Db2 12 release. That is, activation of function level 500 introduces no virtual storage enhancements.

Subsystem parameters

New subsystem parameter settings are in effect only when the function level that introduced them or a higher function level is activated. Most subsystem parameter changes in the initial Db2 12 release take effect in function level 500. For a list of these changes, see Chapter 20, “Subsystem parameter changes in Db2 12,” on page 145.

Optimization enhancements

Optimization enhancements become available after the activation of the function level that introduces them or higher, and full prepare of the SQL statements. When 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

Activation of function level 100 introduces all optimization enhancements in the initial Db2 12 release. That is, function level 500 introduces no optimization enhancements.

SQL capabilities

New SQL capabilities become available after the activation of the function level that introduces them or higher, for applications that run at the equivalent application compatibility level or higher. New SQL capabilities in the initial Db2 12 release become available in function level 500 for applications that run at the equivalent application compatibility level or higher. You can continue to run SQL statements compatibly with lower function levels, or previous Db2 releases, including
Db2 11 and DB2® 10. For details, see Application compatibility levels in Db2 (Db2 Application programming and SQL)

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.

About the Db2 brand change: IBM is rebranding DB2 to Db2. As such, there will be changes to all the Db2 offerings. For example, "DB2 for z/OS" is now referred to as "Db2 for z/OS," beginning with Db2 11. While IBM implements the change across the Db2 family of products, you might see references to the original name "DB2 for z/OS" or "DB2" in different IBM web pages and documents. "DB2 for z/OS" and "Db2 for z/OS" refer to the same product, when the PID, Entitlement Entity, version, modification, and release information match. For more information, see Revised naming for IBM Db2 family products.

The following terms are used as indicated:

Db2
Represents either the Db2 licensed program or a particular Db2 subsystem.

Tivoli® OMEGAMON® XE for Db2 Performance Expert on z/OS
Refers to any of the following products:
• IBM Tivoli OMEGAMON XE 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
Your feedback helps IBM to provide quality information. Please send any comments that you have about this book or other Db2 for z/OS documentation.

Send your comments by email to db2zinfo@us.ibm.com and include the name of the product, the version number of the product, and the number of the book. If you are commenting on specific text, please list the location of the text (for example, a chapter and section title or a help topic title).
Part 1. What's new in the Db2 12 base release

Db2 12 for z/OS takes Db2 to a new level, both extending the core capabilities and empowering the future. Db2 12 extends the core with new enhancements to scalability, reliability, efficiency, security, and availability. Db2 12 also empowers the next wave of applications in the cloud, mobile, and analytics spaces.

Db2 for z/OS news: Get the latest news about new capabilities and enhancements in Db2 for z/OS, from the IBM experts who design, build, test, and support Db2. For details, see News about Db2 for z/OS from the IBM lab.

This information might sometimes also refer to Db2 12 for z/OS as "Db2" or "Version 12."

About the Db2 brand change: IBM is rebranding DB2 to Db2. As such, there will be changes to all the Db2 offerings. For example, "DB2 for z/OS" is now referred to as "Db2 for z/OS," beginning with Db2 11. While IBM implements the change across the Db2 family of products, you might see references to the original name "DB2 for z/OS" or "DB2" in different IBM web pages and documents. "DB2 for z/OS" and "Db2 for z/OS" refer to the same product, when the PID, Entitlement Entity, version, modification, and release information match. For more information, see Revised naming for IBM Db2 family products.

Continuous delivery and Db2 12 function levels

Db2 12 introduces continuous delivery of new capabilities and enhancements in a single service stream as soon as they are ready. 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.

For details about function levels in Db2 and how to activate them, see:

- Part 2, “What's new in Db2 12 function levels,” on page 59
- Part 4, “Adopting new capabilities in Db2 12 continuous delivery,” on page 167
- Exploring IBM Db2 for z/OS Continuous Delivery (IBM Redpaper)
- IBM Db2 12 for z/OS Technical Overview (IBM Redbooks)

Db2 12 function levels 100 and 500 represent the initial set of new capabilities for Db2 12. In some respects, these function levels are comparable to conversion mode (which corresponds to function level 100) and new-function mode (which corresponds to function level 500) from past releases.

Tip: For the most current information, view documentation for Db2 continuous delivery and function levels in IBM Knowledge Center. The continuous delivery process introduces changes to Db2 more frequently than PDF manuals are refreshed.

New capabilities and enhancements in the Db2 12 base release

Most new capabilities in Db2 12 are introduced in Db2 12 function levels. However, some become available immediately in the base Db2 12 release, or when you apply maintenance.

Function levels 100 and 500 activate the initial set of enhancements and capabilities in Db2 12. For migration from Db2 11, most new capabilities become available in Db2 12 only after the activation of function level 500 or higher. For more information about function level 500, see Chapter 11, “Function level 500 (activated at Db2 12 installation or after migration—October, 2016),” on page 81.

Overview of Db2 12 new function availability

The availability of new function depends on the type of enhancement, the activated function level, and the application compatibility levels of applications. In the initial Db2 12 release, most new capabilities are enabled only after the activation of function level 500 or higher.
Virtual storage enhancements

Virtual storage enhancements become available at the activation of the function level that introduces them or higher. Activation of function level 100 introduces all virtual storage enhancements in the initial Db2 12 release. That is, activation of function level 500 introduces no virtual storage enhancements.

Subsystem parameters

New subsystem parameter settings are in effect only when the function level that introduced them or a higher function level is activated. Most subsystem parameter changes in the initial Db2 12 release take effect in function level 500. For a list of these changes, see Chapter 20, “Subsystem parameter changes in Db2 12,” on page 145.

Optimization enhancements

Optimization enhancements become available after the activation of the function level that introduces them or higher, and full prepare of the SQL statements. When 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

Activation of function level 100 introduces all optimization enhancements in the initial Db2 12 release. That is, function level 500 introduces no optimization enhancements.

SQL capabilities

New SQL capabilities become available after the activation of the function level that introduces them or higher, for applications that run at the equivalent application compatibility level or higher. New SQL capabilities in the initial Db2 12 release become available in function level 500 for applications that run at the equivalent application compatibility level or higher. You can continue to run SQL statements compatibly with lower function levels, or previous Db2 releases, including Db2 11 and DB2 10. For details, see Application compatibility levels in Db2 (Db2 Application programming and SQL)

Related information

IBM Db2 12 for z/OS Technical Overview (IBM Redbooks)
Db2 12 for z/OS Performance Topics (IBM Redbooks)
Db2 12 for z/OS Optimizer
Exploring IBM Db2 for z/OS Continuous Delivery (IBM Redpaper)
Db2 12 Technical Overview Part 1
Db2 12 Technical Overview Part 2
Chapter 1. Enhancements in recent Db2 12 APARs (function levels 100 or 500)

A number of enhancements are introduced to the Db2 12 base release, that is function levels 100 and 500, after Db2 12 general availability. Newest APARs based on their release date are listed first.

REPAIR utility enhancement to allow write of Db2 diagnostic log record (PH11871)

This APAR introduces a new REPAIR WRITELOG option to provide the ability for external applications to write a specific Db2 diagnostic log record to support the automatic refresh of replication target tables following a non-recoverable event.

Related reference
Syntax and options of the REPAIR control statement (Db2 Utilities)

Hybrid Transactional Analytical Processing support for accelerated static queries (PH14116)

APAR PH14116 / PTF UI54294 introduces Db2 12 for z/OS support for Hybrid Transactional Analytical Processing (HTAP) of accelerated static queries. Db2 12 support for HTAP of accelerated dynamic queries was introduced by APAR PI91620 / PTF UI54293.

As is the case for accelerated analytical dynamic queries, accelerated analytical static queries now have access to the latest transactional data that has been committed in Db2. Db2 analytical static queries that are run on the analytics accelerator can now return results based on real-time transactional data with no data replication latency, so both your transactional and analytical static queries run on a single system where it seems that real-time analytical processing on real-time data occurs even for static queries that run on an accelerator. Data replication latency now no longer impacts Db2 static query result consistency when using an analytics accelerator.

This HTAP functionality involves Db2 12 for z/OS as the database for the real-time transactional data, IBM Db2 Analytics Accelerator for z/OS Version 5.1 or Version 7.1 as the data repository for the analytical data, and IBM InfoSphere® Data Replication Change Data Capture for z/OS V10.2.1 to replicate the real-time transactional data from the Db2 database to the analytics accelerator. Additional PTFs are required for IBM Db2 Analytics Accelerator for z/OS and for IBM InfoSphere Data Replication Change Data Capture for z/OS.

Related reference
ACCELERATIONWAITFORDATA bind option (Db2 Commands)

Validation for default buffer pools in the installation CLIST (PH12117)

APAR PH12117 introduces validation that default buffer pool names specified in the installation CLIST are defined and have a non-zero VPSIZE. To support this new capability the fields on panels DSNTIP1 and DSNTIP2 are reorganized with the following changes:

• All 4-KB buffer pool size fields are now specified on panel DSNTIP1, instead of being split across two panels.
• The default buffer pool names are now specified on panel DSNTIP2, instead of DSNTIP1.

DSNTIP2 also now validates that each specified default buffer pool exists and has a VPSIZE of greater than 0. For example, if you specify BP1 for the DEFAULT 4-KB BUFFER POOL FOR USER DATA field, the value of the BP1 field in DSNTIP1 must be 1 or greater. DSNTIP2 issues the existing DSNT566I message to report any validation errors.
Console message for inoperative packages (PH13550)

Db2 will now issue a DNST500I console message when a package has become inoperative.

Before this APAR, there was no notification when a package became inoperative and needed to be rebound. This meant that when a package became inoperative, there could be a delay in informing the DBA of the issue with the result being an application outage.

Ability to control workload balancing of accelerated queries based on accelerator version (PH05477)

In a Db2 for z/OS environment that includes a mixture of IBM Db2 Analytics Accelerator for z/OS V5 and V7 accelerator servers, PH05477 enables the ability to control workload balancing such that queries can be accelerated to either a V5 accelerator or a V7 accelerator based on workload volume. Prior to PH05477, workload balancing was limited to a single version of an accelerator.

To control this workload balancing behavior, a new option (10 - BALANCE WORKLOAD BETWEEN VERSIONS) has been added to the QUERY_ACCEL_OPTIONS subsystems parameter.

New job DSNTIJPM premigration report 24: active log data sets larger than 4 GB (PH11514)

APAR PH11514 adds report 24 to the premigration job DSNTIJPM. Report 24 identifies active log data sets greater than 4 GB in size. Before function level 500 is activated in Db2 12, the maximum supported active log data set size is 4 GB. Consequently, during migration to Db2 12, if the size of an active log data set in Db2 11 is greater than 4 GB, Db2 12 abnormally terminates during startup and issues message DSNJ158I. You can use the report to avoid this situation.

New job DSNTIJPM premigration report 23: SQL routines created in DB2 9 (PH07660)

APAR PH07660 adds report 23 to the premigration job DSNTIJPM. Report 23 identifies native SQL procedures and compiled SQL scalar functions created in DB2 9. Such routines are subject to automatic binds after migration to Db2 12. You can use the report to regenerate these routines before migration to avoid performance problems that can result from too many concurrent automatic bind service tasks.
Run premigration queries (DSNTIJPM) (Db2 Installation and Migration)

Statistics profile options included in utility output (APAR PH11423)

APAR PH11423 makes it easier to see the profile options that are being used when gathering statistics. When you run a utility with the USE PROFILE option, the profile options are listed in the SYSPRINT output data set. Prior to this APAR, if you wanted to see the profile options that are used, you first had to run the OPTIONS utility with the PREVIEW option to see the profile options. Then you had to execute the utility. With this APAR, you can see the profile options and execute the utility all in one step. The profile options continue to be displayed in the same message, DSNU1376I, as they were prior to this APAR by OPTIONS PREVIEW.

Related concepts
Statistics profiles (Db2 Performance)

Related tasks
Executing statements in preview mode (Db2 Utilities)

Related information
DSNU1376I (Db2 Messages)

Creating artifacts for provisioning Db2 subsystem with the CLIST (APAR PH09857)

This APAR introduces changes to support creating z/OSMF artifacts for provisioning and de-provisioning standalone Db2 subsystems (no data sharing) by running the installation CLIST. It requires z/OSMF Version 2.3 with APAR PH09130 or higher.

It also delivers a sample Db2 software service template to create services that rapidly provision from scratch one or multiple standalone Db2 subsystems, in IBM Cloud™ Provisioning and Management for z/OS. The files z/OSMF artifacts of the sample software service template are compressed into the Db2ProvisionSystemNonDS.pax file, and this pax file is installed by default in the directory specified by the DDDEF created for SDSNASMP.

Related tasks
Creating artifacts for provisioning Db2 subsystems by running the installation CLIST (Db2 Installation and Migration)
Creating services for provisioning Db2 for z/OS subsystems (DBaaS) (Db2 Installation and Migration)

Allowing the PORT and SECPORT to be the same for static location aliases (APAR PH08188)

Prior to APAR PH08188, Db2 restricted making the normal PORT and SECPORT the same for static location aliases. After the PTF for APAR PH08188 is applied, the PORT and SECPORT can be the same, which means that any TCP/IP communication with Db2 as a server requires SSL connection setup. The rules for the main PORT and SECPORT, as well as any location alias PORT and SECPORT, will be the similar.

Setting the client application compatibility property is optional when the client is Db2 for Linux, UNIX, and Windows Version 11.1 Fix pack 1 or later (APAR PH08482)

Prior to APAR PH08482, in a client-server environment in which the data server was Db2 for z/OS at function level V12R1M501 or later, setting the property value for client application compatibility (clientApplCompat or ClientApplCompat) was required. If that property value was not set, when a client or driver requested a connection with a data server at function level V12R1M501 or later, the request was rejected with SQLCODE -30025.

After the PTF for APAR PH08482 is applied, a Db2 for z/OS data server at function level V12R1M501 or later allows connections from a client or driver for which the client application compatibility property is not set. However, the data server continues to reject connections if the client, driver, or Db2 Connect Server gateway is at a level that does not support the APPLCOMPAT bind option value of the Db2 for z/OS package.
Related tasks
V12R1Mnnn application compatibility levels for data server clients and drivers (Db2 Application programming and SQL)

Related reference
IBM Data Server Driver for JDBC and SQLJ properties for Db2 for z/OS (Db2 Application Programming for Java)

Ability to delete only FlashCopy image copies (APAR PH04023)

APAR PH04023 changes the MODIFY RECOVERY utility so that you can delete only FlashCopy image copies if equivalent sequential image copies exist. This new capability supports an efficient backup procedure that uses minimal disk space: create a FlashCopy image copy virtually instantaneously, create a sequential image copy from the FlashCopy image copy, and then delete the FlashCopy image copy. To delete this FlashCopy image copy, use the new FLASHCOPY ONLY option of MODIFY RECOVERY.

In previous Db2 releases, if you wanted to delete the FlashCopy image copy by using MODIFY RECOVERY, you had to also delete sequential image copies that met the same deletion criteria. Also, if you deleted the FlashCopy image copy by using mechanisms outside of Db2, Db2 did not know that copy was deleted and therefore did not have reliable information about backups.

Related tasks
Backing up data efficiently by using FlashCopy image copies (Db2 Utilities)

Related reference
MODIFY RECOVERY (Db2 Utilities)

Loading multiple input data sets in a single job (APAR PI96136)

In Db2 12, you can load data from multiple data sets into the same target object in a single LOAD utility job. Previously, to load data from multiple data sets, you needed to either run multiple LOAD jobs or concatenate the data sets in the JCL. In some cases, concatenating the data sets is not possible. For example, if you run the LOAD utility by calling the DSNUTILU stored procedure, you cannot concatenate the data sets first. In Db2 12, you can now specify multiple input data sets in the INDDN option of the LOAD statement. You can specify either DD names or templates.

Related reference
Syntax and options of the LOAD control statement (Db2 Utilities)

IBM Db2 AI for z/OS 1.2.0 (APAR PH07782)

This APAR introduces support for Db2ZAI 1.2.0. Additional capabilities added by this release include the ability to detect Db2 system performance exceptions and to provide recommended actions for tuning which are based on your environment.

Another change is that Db2ZAI 1.2.0 now offers a simplified architecture, eliminating the need for a Linux environment.

Improve RUNSTATS performance by making page sampling the default behavior (APAR PH07220)

One way to improve the performance of RUNSTATS is for the utility to use sampling instead of scanning every row and page. RUNSTATS can use either row sampling or page sampling. Page sampling is typically faster in both elapsed time and CPU time. This APAR introduces a new subsystem parameter, STATPGSAMP, that you can use to make page sampling the default behavior for RUNSTATS for universal table spaces. See PAGE-LEVEL SAMPLING field (STATPGSAMP subsystem parameter) (Db2 Installation and Migration).

(FL 505) In Db2 12 function level 505 and higher, the SYSTEM option, which means the same as NO in function level 504 and lower, is changed to mean the same as YES.
New environment variable to control the MQListener log size (APAR PH07230)

This APAR lets you control the maximum size of the MQListener HFS log file using new environment variable MQLSNLWR.

Before this APAR, the HFS log file into which MQListener writes diagnostic information could grow until it reached the HFS limit. For many users, that file size was much too big. After this APAR is applied, users can specify the maximum size for the log file. MQListener creates a copy of and reinitializes the log file when it reaches the maximum size.

Related reference
Environment variables for logging and tracing MQListener (Db2 Application programming and SQL)

Support for sysplex group authentication (APAR PI94236)

This APAR introduces support for sysplex group authentication, which is a security enhancement that enables the use of multi-factor authentication and RACF PassTickets in Db2 sysplex environments. Multi-factor authentication and RACF PassTickets are two distinct methods of providing additional authentication credentials at logon to verify a user's identity. Requiring an additional authentication credential ensures that a user's account cannot be compromised if one of their credentials is discovered.

Specifically, this APAR eliminates connection authentication failures that occur when DRDA sysplex workload balancing or a seamless failover client is being used. Prior to this APAR, connection authentication failures would occur because the drivers that perform sysplex workload balancing or seamless failover attempt to connect with multiple members of a data sharing group on behalf of the original client application connection. These extra connection attempts fail because the additional authentication credential expires or becomes void because it's valid only for the initial connection request and cannot be reused for subsequent connection requests.

Related tasks
Enabling sysplex group authentication for sysplex WLB-enabled clients (Managing Security)

Related reference
AUTH EXIT CACHE REFWR (AUTHEXIT_CACHEREFRESH subsystem parameter) (Db2 Installation and Migration)

Enhancements to insert algorithm 2 (APAR PH02052)

This APAR changes insert algorithm behavior so that when a storage or contention issue disables insert algorithm 2 (IAG2) and forces an INSERT statement to be run at the lower algorithm level, Db2 will automatically attempt to re-enable IAG2 at a set of predefined intervals. Before this APAR, when IAG2 became disabled due to a temporary condition, you needed to stop and restart the object to manually re-enable the insert algorithm, even when the situation that caused the failure had been resolved.

PH02052 also provides some usability and serviceability enhancements in the form of:

• Updates to message DSNI055I, which now provides more detailed explanations of the situations that can cause insert algorithm 2 to become disabled, as well as more details about potential actions that you can take in situations that might require user intervention
• A new message, DSNI087I, which lets you know when insert algorithm 2 has been automatically re-enabled after a failure
• Two new IFCID 2 system level counters that can help you optimize insert algorithm 2 for your environment
• Enhanced diagnostic log records to indicate the successful re-enablement of insert algorithm 2
• Non-critical space map updates under s latch to improve performance
• The prevention of data page contention between concurrent insert algorithm 1 and insert algorithm 2 transactions
• The prevention of ABENDs such as DSNIBHUN:500C and DSNKINSL:5033, which result from insert algorithm 1 and insert algorithm 2 running concurrently on the same table space
New in-memory resource limit table automatic refresh capability and enhancements to START RLIMIT (APAR PH06082)

This APAR introduces an automatic refresh capability for in-memory resource limit tables. When you update the active set of resource limit tables, Db2 detects the resource limit changes and automatically refreshes the in-memory resource limit tables with these changes. You can now also issue a START RLIMIT command that specifies the current active set of resource limit tables without first issuing a STOP RLIMIT command.

Before this APAR, in-memory resource limit tables were not updated when you changed active resource limit tables until the resource limit facility was restarted. As a result, the changes did not take effect immediately for subsequent threads. Also, to start resource limit tables that were already active, you had to first stop the tables with the STOP RLIMIT command.

This APAR also adds the SCOPE(GROUP) keyword to the START RLIMIT command, providing the ability to start the resource limit facility for all members of a data sharing group. Previously, you could issue the START RLIMIT command for only a single member at a time.

Related tasks
Specifying and changing resource limits (Db2 Performance)
Starting and stopping resource limit tables (Db2 Performance)

Related reference
-START RLIMIT (Db2) (Db2 Commands)

Db2 exploitation of asynchronous cross-invalidation for coupling facility cache structures (PH05193)

In a Db2 data sharing group, the elapsed time of writes to the group buffer pool can be quite long and unpredictable. This situation occurs especially in environments where the data sharing members are long distances from each other. z/OS APAR OA54688, which is available starting in z/OS V2R2, provides improved performance for transactions that involve multiple coupling facility cache requests by processing cross-invalidation requests asynchronously. With APAR PH05193, Db2 exploits this new z/OS capability to improve performance of group buffer pool write operations. Although data sharing groups whose members are far from each other can experience more substantial performance improvement, data sharing groups whose members are close together can also benefit.

Related reference
IFCID changes
Db2 12 introduces IFCID changes.

Related information
OA54688: NEW FUNCTION - ASYNCHRONOUS CROSS-INVALIDATION FOR COUPLING FACILITY CACHE STRUCTURES

Specify the default padding behavior for REORG TABLESPACE with UNLOAD EXTERNAL or DISCARD processing (APAR PH00317)

This APAR introduces a new subsystem parameter that specifies the default NOPAD behavior for the REORG TABLESPACE utility with DISCARD or UNLOAD EXTERNAL processing. With this APAR, you can
change the default padding behavior for the REORG TABLESPACE utility by setting the REORG_TS_NOPAD_DEFAULT subsystem parameter value. The default value is YES. The value of this subsystem parameter applies for REORG TABLESPACE utility control statements that omit the NOPAD keyword. However, the REORG TABLESPACE utility continues to honor user-specified NOPAD YES or NO keyword values in the utility control statement.

Before this APAR, if the REORG TABLESPACE utility control statement does not specify the NOPAD keyword the REORG utility always uses NOPAD YES for the default, with the result that the REORG never pads variable length column data in records discarded or unloaded, unless NOPAD NO is explicitly specified.

Related reference
REORG TS NOPAD DEFAULT (REORG_TS_NOPAD_DEFAULT subsystem parameter) (Db2 Installation and Migration)

IBM Db2 AI for z/OS 1.1.0.1 (APAR PH05323)

This APAR introduces support for Db2ZAI 1.1.0.1. Additional capabilities added by this release include the ability to more accurately predict the benefit of parallel processing for static SQL (which can result in improved elapsed time and lower chargeable CPU) and the ability to determine whether sort processes can benefit from increased memory utilization.

Further, the ability to narrow the scope of operations to the package level has been added, allowing gradual exposure of applications to the benefits of machine learning and the ability to expand usage as confidence increases. Another change is that Db2ZAI 1.1.0.1 will activate with a minimum Function Level of 500 provided that the required Db2 maintenance level (PH05323) is also available on all members of the data sharing group.

Avoid external sorts for single-column column groups at the utility job level (APAR PH03678)

When collecting statistics for a single-column column group, avoiding external data sorts can improve utility performance. Previously, you could control the sort behavior by using the STATCLGRST subsystem parameter. STATCLGRST specifies the amount of memory a utility can use for sorting for single-column groups instead of invoking an external sort program. However, STATCLGRST is a global setting, and if the value is large, excessive amounts of memory might be allocated for all utility jobs. Now, you can control this behavior at the utility level by using the new keyword STATCLGMEMSRT. This keyword applies to RUNSTATS, LOAD, and REORG TABLESPACE.

Related reference
STATISTICS COLGROUP DATA SORT STG LIMIT field (STATCLGSRT subsystem parameter) (Db2 Installation and Migration)
LOAD (Db2 Utilities)
RUNSTATS (Db2 Utilities)
REORG TABLESPACE (Db2 Utilities)

New installation jobs for some DSNTIJMV job tasks (APAR PH02971)

APAR PH02971 separates the various tasks in the DSNTIJMV installation job into several new jobs. This change addresses issues with separation of duties, provide more granular control over the tasks to complete for installation or migration, and to facilitate automation of installation, migration, and provisioning. This APAR PH02971 changes or introduces the following jobs for installation of or migration to Db2 12.

DSNTIJMV
The existing DSNTIJMV job is updated to just define Db2 to IRLM and MVS. Other actions previously completed by this job are now completed by the new jobs.
The new DSNTIJMA job adds Db2 and IRLM address space startup procedures to the system procedure library.

The new DSNTIJMW job adds the address space startup procedures for the Db2-supplied WLM environments to the system procedure library.

The new DSNTIJMJ job creates environment files for the Db2-supplied Java™ WLM environment.

The new DSNTIJMP job copies the Db2 language procedures to the system procedure library.

The jobs that the CLIST generates depend on the installation or migration scenario. For example, the installation CLIST generates DSNTIMW, DSNTIJMJ, and DSNTIJMP only for installation and migration of a subsystem or the first member of a data sharing group. Job DSNTIJMA is always generated, but the procedures that it adds change depending on the scenario.

Related tasks
Installation step 3: Define Db2 address space startup procedures, language procedures, and the Java WLM environment: DSNTIJMA, DSNTIJMW, DSNTIJMP, DSNTIJMJ (Db2 Installation and Migration)
Installation step 2: Define Db2 to IRLM and MVS: DSNTIJMV (Db2 Installation and Migration)
Migration step 13: Define Db2 12 to IRLM and MVS: DSNTIJMV (Db2 Installation and Migration)
Migration step 14: update Db2 address space start and language procedure definitions: DSNTIJMA, DSNTIJMW, DSNTIJMP, DSNTIJMJ (Db2 Installation and Migration)
Define the new Db2 data sharing member to IRLM and MVS: DSNTIJMV (Db2 Installation and Migration)
Define address space startup procedures for the new Db2 data sharing member: DSNTIJMA (Db2 Installation and Migration)

Allow ROTATE PARTITION for materialized query tables or tables having dependent materialized query tables (APAR PH00194)

This APAR removes a restriction against rotating partitions when altering materialized query tables (MQTs) or tables with MQTs defined on them. Before this APAR, the ROTATE PARTITION clause of ALTER TABLE statements was restricted if the tables being altered were MQTs or tables with dependent MQTs.

Related tasks
Altering an existing materialized query table (Db2 Performance)

Related reference
ALTER TABLE (Db2 SQL)

New alternate format for timestamp values (APAR PH03263)

This APAR introduces support for a new string representation for timestamp values. Expanding the allowable formats for timestamp values might decrease the amount of changes needed to port applications to Db2 12.

The new format has minus signs between the date and time portions and colons in the time portions of the string, similar to the following: yyyym-ddd-hh:mm:ss. This format is valid for both TIMESTAMP WITH TIME ZONE and TIMESTAMP WITHOUT TIME ZONE values. Refer to the description of TIMESTAMP string-constant in Datetime constants for details about all allowable formats for timestamp values.

Related concepts
Datetime constants (Db2 SQL)
Db2 command line processor (CLP) support for user name and credentials for running SQL files (APAR PH00335)

This APAR introduces support for the Db2 command line processor to accept username and credential parameters when it is invoked to run SQL files. The credentials can be a password or RACF PassTicket and you can specify them in the new -u option flag when you start the Db2 command line processor.

Example: Specifying a user ID and password or RACF PassTicket to run an SQL file

Suppose that you want to use the Db2 command line processor to connect to a server and your system administrator provides the following connection URL for the server:

```
syzs1.abc.com:5021/ABCL0C1
```

You can create a file named `script.sql` that contains the following statement:

```
CONNECT TO syzs1.abc.com:5021/ABCL0C1
```

Also suppose that the alias db2 has been set for the Db2 command line processor name, and your user ID is myid01. If your password is mypw01, you can use the following command to connect to the server:

```
db2 -f script.sql -u myid01/mypw01
```

If your credential is the RACF PassTicket phrase TMOK73SJ, you can use the following command to connect to the server:

```
db2 -f script.sql -u myid01/TMOK73SJ
```

Related reference
- Start syntax for the Db2 command line processor (Db2 Commands)
- CONNECT (Db2 command line processor) (Db2 Commands)

Related information
- The RACF secured signon PassTicket

UNLOAD utility uses REGISTER NO by default in Db2 12 (APAR PI99075)

This APAR changes the default behavior for the UNLOAD utility with SHRLEVEL CHANGE ISOLATION UR to REGISTER YES in Db2 12, which means that pages that are read by the UNLOAD utility are registered with the coupling facility in data sharing. In Db2 11 the UNLOAD utility always uses this behavior.

The new REGISTER NO behavior works as designed in Db2 12 to reduce data sharing overhead, in situations where the possible reduced currency of result data sets can be tolerated. To take advantage of the reduced data sharing overhead, specify REGISTER NO in utility jobs.

Related reference
- Syntax and options of the UNLOAD control statement (Db2 Utilities)

Wide (suffix-W) API for SQLGetDiagRec() to support Unicode wide character type (APAR PH05953)

The ODBC driver currently supports two entry points for each function that passes and accepts character string arguments: a generic API and a wide (suffix-W) API. Support for EBCDIC, ASCII, and UTF-8 encoded character string arguments is provided through APIs in the generic form. Support for UCS-2 encoded character string arguments is provided through APIs in wide form.

This APAR adds support for the wide API form for SQLGetDiagRec(), which is needed in order to support the ibm_db npm module on z/OS.

Related concepts
- Application encoding schemes and Db2 ODBC (Db2 Programming for ODBC)

Related reference
- Suffix-W API function syntax (Db2 Programming for ODBC)
Specify a DDF IPNAME (PI99403)

This APAR introduces support for specifying an IPNAME value when TCP/IP only communication is required. Before this APAR, the Db2 installation process did not support configuration of TCP/IP only communication.

You can specify the IPNAME value in the DB2 TCP/IPNAME field on panel DSNTIPR.

This APAR also reorganizes and renames the following panel fields:

- **DB2 VTAM LUNAME** field (Db2 Installation and Migration) replaces DB2 NETWORK LUNAME on panel DSNTIPR.
- **DB2 VTAM PASSWORD** field (Db2 Installation and Migration) replaces DB2 NETWORK PASSWORD on panel DSNTIPR.
- TCP/IP PORT replaces DRDA PORT, and this field is moved to panel DSNTIPR
- The following fields are also moved to panel DSNTIPR:
  - SECURE PORT field (Db2 Installation and Migration)
  - RESYNC PORT field (Db2 Installation and Migration)
- The following fields are moved to panel DSNTIP5:
  - DDF THREADS field (CMTSTAT subsystem parameter) (Db2 Installation and Migration)
  - MAX INACTIVE DBATS field (MAXTYPE1 subsystem parameter) (Db2 Installation and Migration)
  - IDLE THREAD TIMEOUT field (IDTHTOIN subsystem parameter) (Db2 Installation and Migration)
  - EXTENDED SECURITY field (EXTSEC subsystem parameter) (Db2 Installation and Migration)

Related concepts
Specifying a generic LU name and IPNAME value for the data sharing group (Db2 Data Sharing Planning and Administration)

Related reference
- DB2 TCP/IP IPNAME field (Db2 Installation and Migration)
- DSNJU003 (change log inventory) (Db2 Utilities)

Specify an HFS or zFS directory for z/OSMF artifacts (PI97635)

This APAR reduces the number of manual steps for installation and migration activities with z/OSMF, by introducing support for specifying an HFS or zFS directory for storing z/OSMF artifacts. Before this APAR, you must manually import the artifacts into HFS or zFS from MVS data sets that the Db2 installation CLIST generates.

You can specify the directory in the PATH FOR Z/OSMF ARTIFACTS field on panel DSNTIPM1, when you run the Db2 installation CLIST and specify YES in the USE Z/OSMF WORKFLOW field on panel DSNTIPA1. The user ID that runs the Db2 installation CLIST must have authority to create and write content to the specified directory.

Related reference
- Installation panels for installing or migrating with z/OSMF (Db2 Installation and Migration)

User-defined function with MODIFIES SQL DATA from a subselect (PI93887)

This APAR introduces support in Db2 12 for invoking a user-defined function that is defined with MODIFIES SQL DATA in a subselect. Previously, Db2 issued SQLCODE -740 for this case.

The documentation of the MODIFIES SQL DATA option is also changed to include the following note:

**Recommendation:** If you invoke a function defined with the MODIFIES SQL DATA option in a subselect, make sure that the SQL statements inside the function do not modify any object that is referenced by that subselect.
New console message for remote client information (enabled by APAR PI89903)

This APAR introduces message DSNL076I, which is issued to the z/OS system console for SQL statements that are routed to a server that does not support the requested function.

It identifies the remote client user and environment, the clientApplCompat value supplied by the remote client, and the name and APPLCOMPAT value of the associated Db2 package.

HFS DRBM support for COBOL and PL/I (enabled by APAR PI88171)

Before this APAR, you can use the Db2 coprocessor and generate the DRBM as an HFS file when you invoke the C/C++ compiler from the z/OS UNIX System Services command line. This APAR introduces similar support for the COBOL Version 6.2 or newer and PL/I Version 5.2 or newer compilers.

The following example invokes the coprocessor to compile a sample COBOL program:

```
cob2 myprogram.cbl -c myprogram -dbrmlib -qsql
```

The following example invokes the coprocessor to compile a sample PL/I program:

```
pli -c -qpp=sql -qdbrmlib -qrent myprogram.pli
```

The COBOL and PL/I compilers pass the HFS filename for the DRBM by using the same convention as the C/C++ compiler to the coprocessor service. However, for COBOL and PL/I, the file name cannot be specified with the DBRMLIB option and the file name is always generated, as described in DRBMLIB.

CATMAINT utility LEVEL supports function level values (enabled by APAR PI88058)

This APAR introduces support for Db2 12 function level values for the CATMAINT utility LEVEL option. When you run the CATMAINT utility to tailor the catalog for a new function level, you can specify either the target function level or the corresponding catalog level.

If you specify a function level value, Db2 determines the appropriate catalog level to tailor, and message DSNU7771 indicates the result. If the catalog is already at the appropriate level, message DSNU7661 indicates that no catalog update is required.

Related tasks
Processing SQL statements by using the Db2 coprocessor (Db2 Application programming and SQL)
Binding a DBRM that is in an HFS file to a package or collection (Db2 Application programming and SQL)
Automatic remigration binds are removed in release coexistence and fallback (enabled by APAR PI87675)

This APAR removes the possibility of disruptive repeating rebinds during release coexistence and fallback and remigration scenarios. The ABIND subsystem parameter is changed so that COEXIST behavior is used if YES is specified.

During release coexistence, or during fallback and re-migration, running a plan or package for the first time on the lower Db2 release results in an automatic bind. However, no automatic bind is triggered when a plan or package first runs again on the higher Db2 release.

When you are certain the Db2 is stable on the higher release and coexistence is no longer an issue, you can explicitly rebind the packages on the higher release.

Related concepts
Automatic binds in coexistence (Db2 Installation and Migration)

Related reference
AUTO BIND field (ABIND subsystem parameter) (Db2 Installation and Migration)

DSNTIJSG installation job no longer creates SYSIBM EXPLAIN tables (enabled by APAR PI86450)

This APAR changes the DSNTIJSG installation job to remove CREATE statements for EXPLAIN tables with the SYSIBM qualifier. Db2 no longer uses these tables.

Related tasks
Creating EXPLAIN tables (Db2 Performance)

Related reference
EXPLAIN tables (Db2 Performance)

DSNTIJUZ split into task-specific installation jobs (enabled by APAR PI85657)

This APAR introduces several new jobs for various activities that were previously completed by the single DSNTIJUZ installation and migration job:

- Job DSNTIJUZ: define the Db2 data-only subsystem parameter module (Db2 Installation and Migration)
- Job DSNTIJUA: define data-only application defaults module (Db2 Installation and Migration)
- Job DSNTIJUM: define Db2 data-only offline message generator CCSID (DSNHMCID) module (Db2 Installation and Migration)
- Job DSNTIJUL: update DDF related BSDS information (Db2 Installation and Migration)

Before this APAR, the DSNTIJUZ job completes all of these activities, which often happen separately from the others, and for some only rarely. For example, updating the subsystem parameter module is a frequent task when new subsystem parameters are delivered in the service stream, or to update the default application compatibility (APPLCOMPAT) level. Continuous delivery and function levels in Db2 12 are likely to make this a more frequent activity. A similar more frequent activity with continuous delivery is rebuilding the Db2 application defaults module, such as to change the default SQL processing level. However, a rebuild of Db2 offline message generator CCSID module has never yet been required at migration, nor for applying Db2 service.

With the separate jobs, you can choose the jobs that you need for specific activities, without tailoring DSNTIJUZ to remove or disable unneeded job steps.

This APAR also modifies installation panels used by the z/OSMF installation and migration processes for Db2 12. In the following panels, the step for defining Db2 initialization parameters is split to use the new jobs.
Related concepts
Sample workflow artifacts for installation and migration with z/OS Management Facility (Db2 Installation and Migration)

Related tasks
Updating subsystem parameter and application default values (Db2 Installation and Migration)
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.

Migration step 11: Define Db2 12 initialization parameters: DSNTIJUZ, DSNTIJUA, and DSNTIJUM (Db2 Installation and Migration)
Installation step 6: Define Db2 initialization parameters: DSNTIJUZ, DSNTIJUA, DSNTIJUM, and DSNTIJUL (Db2 Installation and Migration)
Defining Db2 initialization parameters for enabling data sharing (Db2 Installation and Migration)
Defining Db2 initialization parameters for new data sharing members (Db2 Installation and Migration)
Defining Db2 12 initialization parameters for migrating data sharing members (Db2 Installation and Migration)

Related reference
Installation panels for installing or migrating with z/OSMF (Db2 Installation and Migration)

MQListener automatic restart after Db2 restart (enabled by APAR PI84698)
This APAR introduces automatic restart of MQListener processing after Db2 is stopped and restarted, with the -reconnectDB2 option of the db2mq1n1 and db2mq1n2 zun commands. Before this enhancement, you must restart the MQlistener after stopping and restarting Db2, which can cause service interruptions.
If you specify -reconnectDB2 Y, MQListener automatically reconnects and resumes processing after Db2 is stopped and restarted.
You can specify -reconnectDB2 N to continue with the previous behavior.

Related tasks
Configuring MQListener tasks (Db2 Application programming and SQL)

Hybrid Transactional Analytical Processing support for accelerated queries (enabled by APAR PI91620)
APAR PI91620 introduces Db2 for z/OS support for Hybrid Transactional Analytical Processing (HTAP) of accelerated dynamic queries.
With this new functionality, accelerated analytical dynamic queries have access to the latest transactional data that has been committed in Db2. Db2 analytical dynamic queries that are run on the analytics accelerator can now return results based on real-time transactional data with no data replication latency, so both your transactional and analytical queries run on a single system where it seems that real-time analytical processing on real-time data occurs even for queries that run on an accelerator. Data replication latency no longer impacts Db2 dynamic query result consistency when using an analytics accelerator.
This HTAP functionality involves Db2 12 for z/OS as the database for the real-time transactional data, IBM Db2 Analytics Accelerator for z/OS Version 5.1 as the data repository for the analytical data, and IBM InfoSphere Data Replication Change Data Capture for z/OS V10.2.1 to replicate the real-time transactional data from the Db2 database to the analytics accelerator. Additional PTFs are required for IBM Db2 Analytics Accelerator for z/OS and for IBM InfoSphere Data Replication Change Data Capture for z/OS.

Related reference
CURRENT QUERY ACCELERATION WAITFORDATA (Db2 SQL)
SET CURRENT QUERY ACCELERATION WAITFORDATA (Db2 SQL)
Making queries wait for incremental updates

IBM zHyperLink: database synchronous read I/O exploitation (enabled by APAR PI82575)

This APAR introduces support for performance improvements derived from IBM zHyperLink support. In a well-tuned environment, this feature can improve average response latencies for database page reads in online transactions.

The ZHYPERLINK subsystem parameter is introduced for enabling this feature, along with instrumentation to estimate the potential benefit of using the zHyperLink read support, and instrumentation to report the results of zHyperLink requests.

Related concepts
Read operations (Db2 Data Sharing Planning and Administration)

Related reference
DB2 zHyperLinks SCOPE field (ZHYPERLINK subsystem parameter) (Db2 Installation and Migration)

Multiple captures on a single proxy for GDPS Continuous Availability with zero data loss (APAR PI80787)

The GDPS® Continuous Availability with zero data loss solution provides disaster recovery with continuous application availability for enterprise businesses using the z/OS operating system. When Db2 is in a GDPS Continuous Availability with zero data loss environment, IFI applications issue READS calls for IFCID 0306 to a proxy data sharing group, and the proxy data sharing group captures log records from a source data sharing group.

Before this APAR, the GDPS Continuous Availability with zero data loss solution supported only a single IFCID 306 reader on the proxy member. To minimize the capture latency during a maintenance window, Db2 is enhanced to support multiple IFCID 306 readers on the proxy.

Related tasks
Reading complete log data for the GDPS Continuous Availability with zero data loss solution (Db2 Administration Guide)

Suppress SQL warning messages for OPEN and FETCH from SPUFI, DSNTEP2, DSNTEP4, and DSNTIAUL (enabled by APAR PI79053)

This APAR introduces the QUIET option for the TOLWARN parameter of SPUFI and the DSNTEP2, DSNTEP4, and DSNTIAUL programs. QUIET is the same as YES, except that the program suppresses all SQL warning messages from OPEN or FETCH statements if the SQLCODE is 0 or greater.

Related tasks
Controlling toleration of warnings in SPUFI (Db2 Application programming and SQL)

Related reference
DSNTEP2 and DSNTEP4 (Db2 Application programming and SQL)
DSNTIAUL (Db2 Application programming and SQL)

Real-time statistics collection of RUNSTATS-related columns starts at object creation (enabled by APAR PI79234)

This APAR enhances real-time statistics collection for RUNSTATS-related columns in the SYSIBM.SYSTABLESPACESTATS and SYSIBM.SYNTAXESPACESTATS catalog tables. With this APAR applied, the STATSLASTTIME column indicates the timestamp of the last time that the RUNSTATS utility is run on the table space or partition, or the time that table space or partition was created. The following columns are also updated to indicate the value since the object was created:

- STATSINSERTS
Support for collection of more accurate real storage usage statistics (enabled by APAR PI78979)

IFCID 0225 records contain new fields for collection of statistics about the number of discarded pages that are eligible for page steal. The IFCID 0225 record that is written at the end of the first one-minute statistics interval after midnight local time contains those statistics if the following conditions are true:

- Subsystem parameter REALSTORAGE_MANAGEMENT is set to ON or AUTO.
- A trace for IFCID 0503 is started.
- There are discarded frames.
- The PTF for z/OS APAR OA50366 is applied.

You can use those statistics in conjunction with existing real storage usage fields to calculate accurate storage usage once each day. By examining daily storage statistics for multiple weeks, you can determine whether there is an unusual increase in real storage usage.

See the description of IFCID 0225 in prefix.SDSNIVPD(DSNWMSGS) for details.

Important: You might see a performance impact during the time that the new statistics are collected. To avoid unexpected side effects, you might want to enable this enhancement in collaboration with IBM Support.

Explicitly-defined hidden ROWID columns (enabled by APAR PI77310)

This APAR removes a restriction against explicitly defining a ROWID column with the IMPLICITLY HIDDEN attribute. Columns with the IMPLICITLY HIDDEN attribute are not returned for statements that do not explicitly list the column name in the select-clause. For example, the result for SELECT * does not include any implicitly hidden columns. To be included in the result, implicitly hidden columns must be explicitly specified in the select list.

Related concepts
- Row ID values (Db2 SQL)
- ROWID data type (Introduction to Db2 for z/OS)

New LOAD utility IGNORE options (enabled by APAR PI77159)

This APAR introduces support for the following new IGNORE options for the LOAD utility.

- **PART**
  Specifies that records that do not satisfy any partition being loaded are ignored.

- **CONV**
  Specifies that records that cause a conversion error are ignored.

- **VALPROC**
  Specifies that records that fail a validation procedure are ignored.

- **IDERROR**
  Specifies that records that have an identity column value that is out of range are ignored.
DUPKEY
Specifies that records that cause a duplicate key error are ignored.

Related reference
Syntax and options of the LOAD control statement (Db2 Utilities)

Changes to address problems after Db2 table definition changes (enabled by APARs PI86880 and PI88940)

Db2 for z/OS database administrators often need to alter table definitions to support evolving requirements of applications. For table spaces with multiple tables, the following problems can occur:

- In certain situations, such as when a table is copied from one subsystem to another, the table definitions in the catalog might not match the data in the table space, which can lead to data integrity issues. This situation occurs most frequently when alterations are made to multiple tables in a table space.
- If a table space is at the maximum version level of 255, and at least one table in the table space is at version 0, you cannot recycle any table space versions. Until you recycle some table space versions, you cannot perform any more version-generating ALTER operations on tables in the table space.

These APARs provide a number of changes that improve your ability to synchronize the table definitions in the catalog with the data in a table.

Db2 processing changes:

- During UPDATE, INSERT, REORG, or LOAD processing on tables that have had no version-changing ALTER operations, Db2 adds information to the table spaces to make them self-describing. This action eliminates the need to rely on catalog and directory information to determine column definitions.
- After CREATE TABLE processing, and as soon as a table definition is complete, Db2 adds information to the table space to make the table self-describing. The new table has the version number of the table space.

Important: The previously mentioned Db2 processing changes can result in the addition of system pages to a table space, which increases the total size of the table space. You might also see an increase in the processing time for data definition statements, due to the allocation of additional pages in the table space.

Actions that you can take:

- You can run the REPAIR utility with the INSERTVERSIONPAGES and SETCURRENT options to insert missing metadata into a table space, or update table version numbers.

  INSERTVERSIONPAGES adds information to a table space or partition that has not had a version-changing ALTER operation, to make the table space self-describing. SETCURRENTVERSION, which must be specified with INSERTVERSIONPAGES and SHRLEVEL NONE, synchronizes the table version numbers for tables in a table space with the table space version number. The table space version number is the highest version number for any table in the table space. After you use SETCURRENTVERSION to update the versions of any tables that are at version 0, you can run the REORG and MODIFY RECOVERY utilities to recycle version numbers that are now unused.

- You can run the REORG or LOAD utility to insert missing metadata into a table space.

Related concepts
Table space versions (Db2 Administration Guide)

Related tasks
Copying tables from one subsystem to another (Db2 Utilities)

Related reference
Syntax and options of the REPAIR control statement (Db2 Utilities)
ADMIN_INFO_SQL stored procedure enhanced to collect UDF information (enabled by APAR PI73268)

This APAR enhances the ADMIN_INFO_SQL stored procedure and the DSNADMSB program to support the collection of information about user-defined functions (UDFs).

Related concepts
User-defined functions (Db2 SQL)

Related reference
ADMIN_INFO_SQL stored procedure (Db2 SQL)
DSNADMSB (Db2 Utilities)

LOAD utility support for more date and time formats (APAR PI69064)

With this APAR the LOAD utility supports a wider range of data and time formats. It introduces the following new DATE EXTERNAL (date-format) and TIME EXTERNAL (time-format) specifications.

(date-format)
The format of the date representation, as shown in the following table.

<table>
<thead>
<tr>
<th>date-format value</th>
<th>Format</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATE_A</td>
<td>mm-dd-yyyy</td>
<td>10 bytes</td>
</tr>
<tr>
<td>DATE_B</td>
<td>mm-dd-yy</td>
<td>8 bytes</td>
</tr>
<tr>
<td>DATE_C</td>
<td>yyyy-mm-dd</td>
<td>10 bytes</td>
</tr>
<tr>
<td>DATE_D</td>
<td>yy-mm-dd</td>
<td>8 bytes</td>
</tr>
<tr>
<td>DATE_E</td>
<td>dd-mm-yyyy</td>
<td>10 bytes</td>
</tr>
<tr>
<td>DATE_F</td>
<td>dd-mm-yy</td>
<td>8 bytes</td>
</tr>
<tr>
<td>DATE_G</td>
<td>yyyy-ddd</td>
<td>8 bytes</td>
</tr>
<tr>
<td>DATE_H</td>
<td>yy-ddd</td>
<td>6 bytes</td>
</tr>
<tr>
<td>DATE_I</td>
<td>mmdyyyy</td>
<td>8 bytes</td>
</tr>
<tr>
<td>DATE_J</td>
<td>mmdyy</td>
<td>6 bytes</td>
</tr>
<tr>
<td>DATE_K</td>
<td>yyyyymmdd</td>
<td>8 bytes</td>
</tr>
<tr>
<td>DATE_L</td>
<td>yymmd</td>
<td>6 bytes</td>
</tr>
<tr>
<td>DATE_M</td>
<td>dmmyyyy</td>
<td>8 bytes</td>
</tr>
<tr>
<td>DATE_N</td>
<td>dmmmyy</td>
<td>6 bytes</td>
</tr>
<tr>
<td>DATE_O</td>
<td>yyyyddd</td>
<td>7 bytes</td>
</tr>
<tr>
<td>DATE_P</td>
<td>yyyyddd</td>
<td>5 bytes</td>
</tr>
</tbody>
</table>

If a date format is specified for a field that is used in a field specification, the field specification must also use the specified date format.

Notes:
1. If the format includes separators, the separator can be any single-byte character that can be converted to a single-byte EBCDIC character. However, if the input file has a delimited format, you
cannot specify the same character that is used for a delimiter, including the COLDEL, CHARDEL, and DECTP delimiters. For more information about delimiter restrictions, see Loading delimited files (Db2 Utilities).

2. If a two-digit year is specified, it is expanded to a four-digit year. If the two-digit year is less than the sum of the two-digit current year plus 50, then the current century is used in the four-digit year. For example, assume that the current year is 2017. If the two-digit year is 67, 1967 is used. However, if the two-digit year is 66, 2066 is used.

(time-format)
The specific format of the time representation, as shown in the following table.

<table>
<thead>
<tr>
<th>time-format value</th>
<th>Format</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>TIME_A</td>
<td>hh.mm.ss</td>
<td>8 bytes</td>
</tr>
<tr>
<td>TIME_B</td>
<td>hh.mm</td>
<td>5 bytes</td>
</tr>
<tr>
<td>TIME_C</td>
<td>hh.mm AM or hh.mm PM</td>
<td>8 bytes</td>
</tr>
<tr>
<td>TIME_D</td>
<td>hhmmss</td>
<td>6 bytes</td>
</tr>
<tr>
<td>TIME_E</td>
<td>hhmm</td>
<td>4 bytes</td>
</tr>
</tbody>
</table>

If a time format is specified for a field that is used in a field specification, the field specification must also use the specified time format.

Note:
1. If the format includes separators, the separator can be any single-byte character that can be converted to a single-byte EBCDIC character.

Related reference
- Syntax and options of the LOAD control statement (Db2 Utilities)

LOAD REPLACE SHRLEVEL REFERENCE (enabled by APAR PI69095)
This APAR introduces support for LOAD REPLACE SHRLEVEL REFERENCE, which enabled concurrent read application access to the target table while new data is being loaded into a set of shadow data sets. A new SWITCH phase is introduced to the LOAD utility to switch access between the original and shadow data sets.

Related reference
- Syntax and options of the LOAD control statement (Db2 Utilities)

LOAD RESUME SUPPORT FOR INLINE COPY (enabled by APAR PI81724)
To avoid unavailability of the table space or table space partition, the LOAD syntax has been enhanced to allow the COPYDDN or RECOVERYDDN option to be specified when RESUME YES is specified. A full image copy is taken at the end of LOAD processing, after potential BACKOUT processing.

Related reference
- Syntax and options of the LOAD control statement (Db2 Utilities)

Reduced cost for collecting Db2 frequency statistics (enabled by APAR PI76730)
Your enterprise now has more control over performance when you collect frequency statistics for single-column column groups through RUNSTATS or inline statistics jobs. By adjusting the value of a new subsystem parameter, STATCLGSRT, you can potentially improve performance and reduce costs.

The STATCLGSRT subsystem parameter specifies the amount of memory that Db2 can use to avoid a sort operation for RUNSTATS or other inline statistics utility jobs when FREQVAL is specified on a COLGROUP
that identifies one or more single-column column groups. By increasing the value of STATCLGSRT to allocate more memory, you enable Db2 to collect the statistics you need without performing a costly sort.

**Related tasks**
Reducing the cost of collecting statistics (Db2 Performance)

**Related reference**
STATISTICS COLGROUP DATA SORT STG LIMIT field (STATCLGSRT subsystem parameter) (Db2 Installation and Migration)
Syntax and options of the LOAD control statement (Db2 Utilities)
Syntax and options of the REORG TABLESPACE control statement (Db2 Utilities)
RUNSTATS TABLESPACE syntax and options (Db2 Utilities)

**Support for tracing native SQL routines (enabled by APAR PI44721)**

SQL procedures in Db2 provide the ability to trace native SQL routines. Those SQL procedures are:

- **DSN8.CREATE_DGTT**
  Creates a declared temporary table that is used by the sample trace procedures.

- **DSN8.ENABLE**
  Enables the SQL routine trace message buffer.

- **DSN8.GET_LINE**
  Returns a single line from the SQL routine trace message buffer. The returned data does not include an end-of-line character sequence.

- **DSN8.GET_LINES**
  Returns one or more lines from the SQL routine trace message buffer, and stores the lines in an array. The returned data does not include end-of-line character sequences.

- **DSN8.NEW_LINE**
  Writes an end-of-line character sequence to the SQL routine trace message buffer.

- **DSN8.PUT**
  Writes a line to the SQL routine trace message buffer, without an end-of-line character sequence.

- **DSN8.PUT_LINE**
  Writes a line to the SQL routine trace message buffer, with an end-of-line character sequence.

- **DSN8.DISABLE**
  Disables the SQL routine trace message buffer.

To define the SQL procedures and variables that are used by the SQL procedures, run job DSNTEJTR.

**Related reference**
DSN8.CREATE_DGTT stored procedure (Db2 SQL)
DSN8.DISABLE stored procedure (Db2 SQL)
DSN8.ENABLE stored procedure (Db2 SQL)
DSN8.NEW_LINE stored procedure (Db2 SQL)
DSN8.GET_LINE stored procedure (Db2 SQL)
DSN8.GET_LINES stored procedure (Db2 SQL)
DSN8.PUT stored procedure (Db2 SQL)
DSN8.PUT_LINE stored procedure (Db2 SQL)
Objects that are used by the sample trace stored procedures (Db2 SQL)
Chapter 2. Application enablement in the Db2 12 initial release

The initial release of Db2 12 introduces new capabilities and enhancements for application enablement.

**Important:** Db2 12 introduces continuous delivery of new capabilities and enhancements in function levels. Most Db2 12 enhancements and changes become available only after the activation of the function level that delivers them. For more information, see Chapter 29, “Activating Db2 12 function levels,” on page 179. Each new or changed item is labeled below with a link to information about the Db2 12 function level that delivered it.

**Related information**
IBM Db2 12 for z/OS Technical Overview (IBM Redbooks)

### Additional support for triggers

Db2 12 introduces support for **advanced triggers**, while continuing to support triggers as supported by previous releases. Triggers of this type are now referred to as **basic triggers**.

Any trigger created before the activation of function level 500 or higher is a basic trigger. For more information about the differences between basic and advanced triggers, see Triggers (Introduction to Db2 for z/OS). The following changes apply only to advanced triggers.

In Db2 12, Db2 for z/OS, an advanced trigger definition can:

- Define and reference SQL variables
- Include more types of traditional SQL statements than in previous releases, including dynamic SQL statements
- Include SQL PL control statements
- Reference global variables, or assign values to global variables
- Explicitly specify options, including bind options
- Include SQL comments
- Include definitions for multiple versions of the trigger

In addition, the following changes are introduced to advanced triggers:

- All transition variables are nullable
- An ALTER TRIGGER statement can be used to change trigger options or regenerate the trigger body
- An advanced trigger can be debugged

The CREATE TRIGGER (advanced) statement supports the OR REPLACE clause, which enables the use of one CREATE statement to define a trigger or update an existing trigger depending on whether the trigger already exists. The OR REPLACE option can also be used with a CREATE TRIGGER statement to define a new version of a trigger, or to replace an existing version of a trigger.

**Related reference**
CREATE TRIGGER (advanced) (Db2 SQL)
CREATE TRIGGER (basic) (Db2 SQL)
ALTER TRIGGER (basic) (Db2 SQL)
**Additional support for arrays**

Db2 12 enhances array support in SQL statements.

Db2 for z/OS Db2 11 provided support for defining and manipulating arrays within SQL PL routines, and for passing arrays as parameters between routines.

Array support in Db2 12 adds the following enhancements:

- The ability to define a global variable as an array type. A global variable that is defined as an array type can be manipulated within an SQL PL context, or outside an SQL PL context.
- The following ARRAY_AGG function improvements:
  - ARRAY_AGG can be invoked without an ORDER BY clause.
  - ARRAY_AGG can be used with associative arrays.

**Related concepts**

Array types (Db2 SQL)
Global variables (Db2 SQL)

**Related reference**

ARRAY_AGG (Db2 SQL)
CREATE VARIABLE (Db2 SQL)

**Additional support for global variables**

Db2 12 enhances global variable support.

The following enhancements are included in Db2 12:

- Definition of a global variable with a BLOB, CLOB, or DBCLOB data type
- In a SET assignment-statement, specification of global variables as the target, and a row-subselect as the source.

**Related reference**

CREATE VARIABLE (Db2 SQL)

**Additional support for pureXML®**

Db2 12 enhances XML performance in several ways.

In Db2 12 of Db2 for z/OS, you can perform multiple document updates within a single XMLMODIFY invocation.

The updates can be in one of the following expressions in the xquery-update-constant of the XMLMODIFY function:

- A sequence expression
- A FLWOR expression that contains an updating expression in its return clause
- A conditional expression that contains an updating expression in its THEN or ELSE clause

In addition, Db2 12 improves the performance of XML queries by optimizing the choice of access plans.

The performance of XMLTable is also improved, in terms of the CPU time for constructing rows out of repeated elements that contain name-value pairs.

The XSLTRANSFORM function provides the ability to transform an XML document into a different data format. The output format can be any form possible for the XSLT processor, including, but not limited to, XML, HTML, and plain text.
Related reference
XMLMODIFY (Db2 SQL)

Additional support for JSON

Db2 12 adds support for the requirements imposed by web and mobile applications, with two ways of working with JSON: the Java driver for the JSON API or through SQL extensions.

The JSON_VAL function is enhanced in that the first argument is no longer required to be a BLOB column. In Db2 12 the first parameter can be:

- a view column
- a CASE expression
- a table expression with UNION ALL
- a trigger transition variable
- a SQL PL variable or parameter

Related concepts
Programming with JSON (Programming with JSON)

MERGE statement enhancements

Db2 12 introduces changes to the SQL MERGE statement that provide more functionality and improve compatibility with the Db2 family.

Enhancements to the MERGE statement include support for:

- table-reference as an alternative way of specifying source data for the MERGE statement
- Multiple MATCHED clauses
- Additional predicates with MATCHED or NOT MATCHED
- DELETE operations
- IGNORE and the SIGNAL statement as actions

Related reference
MERGE (Db2 SQL)

SQL pagination support

Db2 12 introduces SQL syntax that enables dividing a query result table into separate pieces, so that, for example, mobile and web applications can present a subset of the rows of the result table at a time.

- With the growth of web and mobile applications, application developers are looking for more efficient ways to develop good-performing applications. Db2 12 introduces the OFFSET clause to define the number of rows to skip from the beginning of the query result table:

  SELECT * FROM TAB OFFSET 10 ROWS FETCH FIRST 10 ROWS ONLY;

- Also available with Db2 12 is data-dependent pagination, which uses row value expressions in a basic predicate, which enables an application to access part of a Db2 result table based on a logical key value:
WHERE (LASTNAME, FIRSTNAME) > (SMITH, JOHN)

Prior to Db2 12, such a statement had to be coded as:

WHERE (LASTNAME = 'SMITH' AND FIRSTNAME > 'JOHN') OR (LASTNAME > SMITH)

**Related tasks**
Accessing part of a result set based on data position (Db2 Application programming and SQL)

**Related reference**
offset-clause (Db2 SQL)
Basic predicate (Db2 SQL)
fetch-clause (Db2 SQL)

---

**Support for Unicode columns in an EBCDIC table**

Db2 12 extends support for Unicode columns in EBCDIC tables. A Unicode column in an EBCDIC table that is defined with Db2 12 is subject to fewer restrictions than if the column had been defined using Db2 11.

As businesses become more global, there is a need to store a wider set of characters. For example, with the expansion of EU countries, the customer name from a newly joined country might not be stored correctly in an SBCS EBCDIC column. Db2 12 addresses this need by providing a byte-based Unicode column of CCSID 1208 or CCSID 1200 in an EBCDIC table.

Unicode columns created in Db2 11 can be migrated to Db2 12 Unicode columns.

**Related concepts**
Unicode column in an EBCDIC table (Db2 SQL)

**Related tasks**
Migrating a Db2 11 Unicode column in an EBCDIC table to a Db2 12 Unicode column in an EBCDIC table (Db2 SQL)

---

**Piece-wise deletion of data**

Db2 12 introduces functionality that helps to mitigate the effects of locking, when potentially millions of rows could be affected by a DELETE statement.

Db2 12 adds FETCH FIRST n ROWS ONLY syntax to the searched DELETE statement to limit the number of rows that are deleted in a single SQL statement.

**Related reference**
DELETE (Db2 SQL)
fetch-clause (Db2 SQL)
Support for temporal referential constraint

Db2 12 introduces referential constraints for application-period temporal tables.

In previous versions, temporal referential integrity could only be enforced using triggers or stored procedures. In Db2 12, a temporal referential constraint can be defined for an application-period temporal table that contains a BUSINESS_TIME period. You can specify the PERIOD BUSINESS_TIME clause in the definition of a referential constraint that enforces a temporal referential constraint for an application-period temporal table.

Related concepts
Temporal tables and data versioning (Db2 Administration Guide)

Related tasks
Creating an application-period temporal table (Db2 Administration Guide)

More flexibility in defining application periods for temporal tables

In Db2 12, you can create an application-period temporal table with a BUSINESS_TIME period that is inclusive-inclusive. In previous versions, only inclusive-exclusive periods were supported.

For application-period temporal tables, an application period indicates the period of time when a row is valid. This period is defined by a begin column and an end column. In previous versions of Db2, the begin value of a period was always inclusive, but the end value of a period was always exclusive. This type of period is called an inclusive-exclusive period. For example, if the begin column has a value of ‘01/01/2015’ and the end column has a value of ‘03/21/2015’, the row is valid from 01/01/2015 to 3/20/2015, including both of those dates. The row is not valid on 03/21/2015.

Beginning in Db2 12, you can define an application period to be inclusive-inclusive. For an inclusive-inclusive period, the end value is considered to be part of the period. To specify an inclusive-inclusive period, use the new INCLUSIVE keyword in the PERIOD BUSINESS_TIME clause of the CREATE TABLE or ALTER TABLE statement. By default, an application period is inclusive-exclusive.

Whether you should use an inclusive-inclusive period or an inclusive-exclusive period depends on what you think is the most natural way to express a certain period of time. For example, for an insurance policy, an inclusive-inclusive period might make sense. A policy can be valid from July 1 to December 31, where both end points are part of the period. In contrast, an inclusive-exclusive period might make sense for hotel reservations. You check in on June 30 and check out on July 4. In this case, you do not want July 4 to be included in the nights that the hotel room is reserved. An inclusive-exclusive period might also be useful when the end value is the end of the month. You can specify a period from February 1 through March 1, not including March 1. By specifying the period this way, you do not have to figure out whether the last day of February is the 29th or the 28th.

Related concepts
Temporal tables and data versioning (Db2 Administration Guide)

Related tasks
Creating an application-period temporal table (Db2 Administration Guide)

Related reference
CREATE TABLE (Db2 SQL)
ALTER TABLE (Db2 SQL)

Support for temporal logical transactions

Db2 12 introduces temporal logical transaction support for system-period temporal tables (SYSTEM_TIME periods).

Temporal logical transaction support applies to SYSTEM_TIME periods and applications using Db2 system-period temporal tables.
Db2 now supports logical units of temporal work that are not determined by commit and rollback. The values for row-begin and row-end columns are determined by applications based on a built-in global variable that you set.

**Related concepts**
Temporal tables and data versioning (Db2 Administration Guide)

---

**PERCENTILE function support**

Db2 12 introduces the PERCENTILE_CONT and PERCENTILE_DISC functions, which compute a percentile value within a given group.

PERCENTILE_CONT treats the group as a continuous distribution, meaning that if the percentage falls between two values, the result is interpolated between those two values. For example, the median (PERCENTILE_CONT(0.5)) of a group of 6 numbers would be average of the 3rd and 4th number, because the interpolated middle value between those two is found.

PERCENTILE_DISC treats the group as a set of discrete values. The result is not interpolated and is always one of the values in the group. PERCENTILE_DISC(0.5) over a group of 6 numbers returns the 3rd number.

**Related reference**
PERCENTILE_CONT (Db2 SQL)
PERCENTILE_DISC (Db2 SQL)

---

**DRDA fast load**

Db2 12 introduces DRDA fast load, which enables quick and easy loading of data from files residing on distributed clients.

The Db2 Call Level Interface (CLI) APIs and Command Line Processor (CLP) have been enhanced to support remote loading of data to Db2 for z/OS. This new feature is supported in all Db2 client packages.

Before using DRDA fast load, you must bind the DSNUT121 package at each location from which you want to load data. The following example binds the DSNUT121 package at a remote location:

```
BIND PACKAGE(location.DSNUT121) 
MEMBER(DSNUGSQL) - 
ACTION(ADD) ISOLATION(CS) ENCODING(EBCDIC) - 
VALIDATE(BIND) CURRENTDATA(NO) - 
LIBRARY('prefix.SDSNBKRM')
```

---

**ODBC enhancements**

Db2 12 introduces enhancements in the Db2 ODBC driver for z/OS to improve performance and portability.

Enhancements to the ODBC driver include support for:

- New ODBC INI keyword KEEPDYNAMIC and connection attribute SQL_ATTR_KEEP_DYNAMIC
- Support for the TIMESTAMP WITH TIMEZONE data type

**Related concepts**
Programming for ODBC (Db2 Programming for ODBC)
Obfuscated source code for SQL routines and triggers

Db2 12 introduces a method for protecting the encapsulated logic in SQL stored procedures, SQL functions, and triggers. In obfuscated data definition statements for SQL procedures, SQL functions, and triggers, the SQL logic is rendered unreadable, enabling the delivery of SQL routines and triggers without sharing the intellectual property of the SQL PL logic.

The data definition statements for routines and triggers can execute in obfuscated form. In obfuscated statements, only the name of the routine or trigger, the parameters, and the WRAPPED keyword are readable. The rest of the statement is encoded in such a way that it is not readable but can be decoded by a database server that supports obfuscated statements.

You can use the WRAP built-in function to create the obfuscated form of the data definition statements for SQL procedures, SQL functions, routines, and triggers. The CREATE_WRAPPED stored procedure also supports the deployment of obfuscated routines and functions by calling the WRAP built-in function.

Important: The obfuscation algorithm is not strong encryption, and it is not meant for use in security contexts.

Related tasks
Obfuscating source code of SQL procedures, SQL functions, and triggers (Db2 Administration Guide)

Related reference
WRAP (Db2 SQL)
CREATE_WRAPPED stored procedure (Db2 SQL)
CREATE FUNCTION (compiled SQL scalar) (Db2 SQL)
CREATE FUNCTION (inlined SQL scalar) (Db2 SQL)
CREATE FUNCTION (SQL table) (Db2 SQL)
CREATE PROCEDURE (SQL - native) (Db2 SQL)
CREATE TRIGGER (basic) (Db2 SQL)
CREATE TRIGGER (advanced) (Db2 SQL)

Support for maintaining session data on the target server

Db2 12 introduces an enhancement to reduce network traffic through the use of session tokens on the Db2 server.

Prior to Db2 12, in order to support transaction rerouting and pooling, distributed clients needed session data (special registers and global variables) to be returned and replayed.

With Db2 12, session data is maintained on the Db2 server when clients interchange session tokens. This enhancement results in reduced network traffic.

Related reference
Built-in session variables (Db2 SQL)

Resource limits for static SQL statements

The resource limit facility can be used to limit CPU execution times and resource consumption by static SQL statements that are bound in packages.

In previous versions of Db2, limits specified in the resource limit facility applied only to dynamic SQL statements.

Db2 12 introduces the ability to specify reactive governing for static SQL statements by adding new rows in resource limit facility tables. Static SQL statements are governed by rows that specify the RLFFUNC='A' (for DSNRLSTxx tables) and RLFFUNC='B' (for DSNRLMTxx tables) values in the resource limit tables.
You can use this function to cancel poorly performing SQL statements that consume too many resources, especially in high-volume OLTP environments.

The value of the RLFENABLE subsystem parameter controls whether resource limits apply to dynamic SQL statements only, static SQL statements only, or to all SQL statements.

You specify default resource limit actions for static SQL statements by setting the values of the RLFERRSTC (for local statements) and RLFERRDSTC (for remote statements) subsystem parameters.

**Related tasks**
- Setting limits for system resource usage by using the resource limit facility (Db2 Performance)
- Setting default resource limits for SQL statements (Db2 Performance)

**Related reference**
- Resource limit facility tables (Db2 Performance)
- RLF SCOPE field (RLFENABLE subsystem parameter) (Db2 Installation and Migration)
- STATIC SQL field (RLFERRSTC subsystem parameter) (Db2 Installation and Migration)
- REMOTE STATIC SQL field (RLFERRDSTC subsystem parameter) (Db2 Installation and Migration)

---

**Db2 REST services improve efficiency and security**

The Db2 REST services support, available in Db2 12, unleashes your enterprise data and applications on Db2 for z/OS for the API economy. Mobile and cloud application developers can use efficiently created, consumable, scalable, and RESTful services by using any REST client. They can use these services to securely interact with business-critical data and transactions, without special Db2 for z/OS expertise.

Recent service updates to the Db2 REST services support introduces the BIND SERVICE and FREE SERVICE commands and versioning to REST services. Db2 database administrators can issue these DSN commands to create and drop user-defined REST services the same way as they use BIND PACKAGE and FREE PACKAGE for normal application package creation and deletion. The Db2 commands, START RESTSVC, STOP RESTSVC, and DISPLAY RESTSVC, were also introduced with the recent service updates to the Db2 REST services support. The commands can be used by DBAs to manage the availability of user-defined REST services.

- Db2 for z/OS RESTful API enabling the Mobile Economy
- Db2 for z/OS Native REST services (IBM developerWorks)
- Accessing HTTP and RESTful services from Db2: Introducing the REST user-defined functions for Db2 (IBM developerWorks)

**Related concepts**
- Db2 REST services

**Related reference**
- FREE SERVICE (DSN) (Db2 Commands)
- BIND and REBIND options for packages, plans, and services (Db2 Commands)

---

**Modern language support Db2 for z/OS application development**

Db2 11 and Db2 12 now support application development in many modern programming and scripting languages. Application developers can use languages like Python, Perl, and Ruby on Rails to write Db2 for z/OS applications. Getting business value from your mainframe applications is now more accessible than ever before.

- Take advantage of Db2 for z/OS modern application development language support
  - IBM data server client and driver types
  - IBM Data Server Driver Package (Windows)
  - Programming for Db2 for z/OS (Introduction to Db2 for z/OS)
DevOps with Db2: Automated deployment of applications with IBM UrbanCode Deploy

With UrbanCode® Deploy, you can easily automate the deployment and configuration of database schema changes in Db2 11 and Db2 12. The automation reduces the time, costs, and complexity of deploying and configuring your business-critical apps, getting you to business value faster and more efficiently.

Db2 for z/OS - IBM Urban Code Deploy - Automate Application Deployments
IBM Db2 for z/OS Package (UrbanCode Deploy)
UrbanCode Deploy Tour
Deploy database applications against Db2 for z/OS with IBM UrbanCode deploy

Related concepts
Db2 application provisioning, development, and deployment (DevOps) solutions

Related information
UrbanCode Deploy
Chapter 3. Administrator function in the Db2 12 initial release

The initial release of Db2 12 introduces enhancements and new capabilities for administrators.

**Important:** Db2 12 introduces continuous delivery of new capabilities and enhancements in function levels. Most Db2 12 enhancements and changes become available only after the activation of the function level that delivers them. For more information, see Chapter 29, “Activating Db2 12 function levels,” on page 179. Each new or changed item is labeled below with a link to information about the Db2 12 function level that delivered it.

**Related information**
IBM Db2 12 for z/OS Technical Overview (IBM Redbooks)

**Single-phase catalog migration in Db2 12**

The Db2 12 catalog migration is accomplished in a single phase.

Migration to the Db2 12 catalog is completed by running the DSNTIJTC job. The following migration modes from previous releases are eliminated in Db2 12:

- Conversion Mode (CM)
- Enabling-new-function mode (ENFM)
- New-function mode (NFM)
- * modes (including CM* and ENFM*)

In Db2 12, you can use Db2 function levels and application compatibility levels to control the adoption of new capabilities and enhancements. For details, see Part 4, “Adopting new capabilities in Db2 12 continuous delivery,” on page 167 and Application compatibility levels in Db2 (Db2 Application programming and SQL).

**Important:** Do not issue the ACTIVATE command or run job DSNTIJAF for activation of function level 500 or higher until you are certain that the subsystem or data sharing group can proceed on Db2 12, without the possibility of falling back to or coexistence with Db2 11. In data sharing, the ACTIVATE command has group scope. Fallback and coexistence become impossible with the successful activation of function level 500 or higher.

**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 and examples, see Chapter 27, “Determining the Db2 code level, catalog level, and function level,” on page 175.

**Related concepts**
What's new in the Db2 12 base release
Db2 12 for z/OS takes Db2 to a new level, both extending the core capabilities and empowering the future. Db2 12 extends the core with new enhancements to scalability, reliability, efficiency, security, and availability. Db2 12 also empowers the next wave of applications in the cloud, mobile, and analytics spaces.

**Related tasks**
Migrating your Db2 subsystem to Db2 12 (Db2 Installation and Migration)

**Related reference**
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.

### Installation or migration without requiring SYSADM

In Db2 12, you can use the installation SYSOPR authority to install or migrate a Db2 subsystem. You are not required to use the SYSADM or installation SYSADM authority to perform installation or migration tasks. Using the installation SYSOPR authority prevents unintended access to user objects during installation or migration.

For installing or migrating Db2, the installation SYSOPR authority is enhanced to access catalog and directory objects, create, drop, or alter system objects, execute the -ACTIVATE command, specify the owner for the BIND and REBIND commands, and run the CATMAINT utility. However, you are not allowed any access to non-system objects or user data in the subsystem.

The overall process of installing or migrating Db2 with the installation SYSOPR authority is the same as that with the SYSADM authority. You can use the installation SYSOPR authority to perform all installation or migration tasks if you meet the following additional requirements:

- The primary or secondary authorization ID that performs installation or migration tasks must have the installation SYSOPR authority.
- The current SQLID that processes data definition statements must be set to SYSINSTL which owns all the system objects created by the installation SYSOPR authority.
- The OWNER option for the BIND and REBIND commands must be set to an ID that is authorized to bind and execute SQL statements in a package.
- The new DSNTIAIN plan must be used with the DSNTIAD program to process all dynamic data definition statements for Db2 installation or migration.
- An explicit qualifier must not be specified for the CREATE TABLE or CREATE INDEX statement. The qualifier can be implicitly specified through a leading SET CURRENT SCHEMA statement so that SYSINSTL can own the table or index.

**Related concepts**

Required authorization for installation or migration (Db2 Installation and Migration)

**Related reference**

DSNTIPG: Installation preferences panel (Db2 Installation and Migration)

### Support for preserve mirror options for recovery

Db2 12 supports the preserve mirror options for the RESTORE SYSTEM and RECOVER utilities. These options are useful if you use Peer-to-Peer Remote Copy (PPRC) to mirror your Db2 production volumes to a remote site for high availability.

During recovery, you can use the preserve mirror options to override the DFSMSHsm preserve mirror settings that were in effect when the system-level backup was created. To do so, use the new FLASHCOPY_PPRCP option of RESTORE SYSTEM or RECOVER.

In previous versions of Db2, if you had to override or change the preserve mirror options during recovery from a system-level backup, you had to invoke DFSMSHsm directly.

**Related concepts**

Preserve Mirror FlashCopy (z/OS DFSMSdss Storage Administration)

**Related reference**

Syntax and options of the RECOVER control statement (Db2 Utilities)

Syntax and options of the RESTORE SYSTEM control statement (Db2 Utilities)
**Improved availability when altering index compression**

To improve the availability of indexes in universal table spaces, alterations to index compression are a pending change in Db2 12.

For indexes in universal table spaces, alterations to index compression are a pending change that place the index in advisory REORG-pending (AREOR) status. Applications can continue to access the indexes. The updated value for the COMPRESS attribute in the ALTER INDEX statement is materialized by a subsequent online REORG INDEX or online REORG TABLESPACE at the table space level. With this improvement, database administrators can correct or remove a pending change to index compression without affecting the target index. Also, this improvement reduces the planning and costs that are associated with an application outage caused by the previous behavior.

**Note:** If the index is defined with the DEFINE NO attribute and data sets are not created yet, the alteration is still immediate. Also, for an index that is not in a universal table space, an alteration to index compression can be a pending change if other pending changes exist at the index, table, or table space level when the ALTER INDEX COMPRESS statement runs.

In previous releases, any alteration to use index compression placed the index in REBUILD-pending (RBDP) status. This behavior prevented applications from using the index until the REORG TABLESPACE utility or the REBUILD INDEX utility completed.

**Related tasks**
Compressing indexes (Db2 Performance)

**Related reference**
ALTER INDEX (Db2 SQL)

**Online schema enhancements**

Db2 12 improves system and application availability by extending the scope of recovery to a point in time before materialization of pending definition changes. In addition, in Db2 12, column alterations can be pending definition changes.

**Improvements in recovery to a point in time before materialization of pending definition changes**

Db2 11 introduces the capability to recover the following objects to a point in time before the pending definition changes were materialized:

- partition-by-range table spaces
- LOB or XML table spaces

Db2 12 extends recovery to a point in time before pending definition changes were materialized in the following ways:

- The object that is being recovered can be a partition-by-growth table space.

**Pending column alterations**

In Db2 12, ALTER TABLE ALTER COLUMN statements that change the data type, length, precision, or scale of columns can be pending alterations. By default, column alteration operations are immediate operations. To make column alterations in a subsystem pending operations, set the DDL_MATERIALIZATION subsystem parameter to ALWAYS_PENDING.
Improved availability when altering index compression

To improve the availability of indexes in universal table spaces, alterations to index compression are a pending change in Db2 12. For indexes in universal table spaces, alterations to index compression are a pending change that place the index in advisory REORG-pending (AREOR) status. Applications can continue to access the indexes. The updated value for the COMPRESS attribute in the ALTER INDEX statement is materialized by a subsequent online REORG INDEX or online REORG TABLESPACE at the table space level. With this improvement, database administrators can correct or remove a pending change to index compression without affecting the target index. Also, this improvement reduces the planning and costs that are associated with an application outage caused by the previous behavior.

Note: If the index is defined with the DEFINE NO attribute and data sets are not created yet, the alteration is still immediate. Also, for an index that is not in a universal table space, an alteration to index compression can be a pending change if other pending changes exist at the index, table, or table space level when the ALTER INDEX COMPRESS statement runs.

In previous releases, any alteration to use index compression placed the index in REBUILD-pending (RBDP) status. This behavior prevented applications from using the index until the REORG TABLESPACE utility or the REBUILD INDEX utility completed.

Related concepts
Point-in-time recovery (Db2 Utilities)
Pending column alterations (Db2 Administration Guide)

Related tasks
Recovering to a point in time before pending definition changes were materialized (Db2 Administration Guide)

Related reference
DDL MATERIALIZATION field (DDL_MATERIALIZATION subsystem parameter) (Db2 Installation and Migration)

Improved catalog availability

Db2 12 improves the availability of Db2 catalog objects on which CATMAINT and online REORG run.

When a transaction issues dynamic SQL statements, the Db2 database manager dynamically prepares the SQL statements for execution. During preparation of SQL statements, the database manager acquires read claims on several catalog table spaces and related indexes, and acquires a DBD lock on the catalog. The DBD lock is needed to serialize catalog operations with CATMAINT and other DDL that may execute against the catalog, because CATMAINT might be making structural changes to the catalog.

In previous releases of Db2, the transaction released the DBD lock and the read claims at commit points. If transactions with dynamic SQL statements did not issue commit operations for a long time, CATMAINT and online REORG on the catalog were blocked during that long period.

In Db2 12, the Db2 database manager releases DBD locks on the catalog and read claims against catalog objects as soon as PREPARE statement execution is complete. This change improves availability for CATMAINT and online REORG on catalog objects.

Object ownership transfer

In Db2 12, you can transfer user-defined objects to another user or role by using the new TRANSFER OWNERSHIP statement. The new owner can be an authorization ID or a role.

The privilege set for the TRANSFER OWNERSHIP statement includes the ownership of the object or the SECADM authority. The SEPARATE_SECURITY system parameter does not impact the SECADM authority.
Improved data validation after running DSN1COPY

After the DSN1COPY utility runs, mismatches between the data page format and the catalog description of the format can result. The REPAIR utility is enhanced to identify and fix more of those mismatches.

If DSN1COPY is not used correctly, subsequent attempts to access the data can result in abends, data corruption, and storage overlays. For example, the following scenarios might result in such errors:

- Incorrect DBID, PSID, or OBID values are specified when you run DSN1COPY with the OBIDXLAT option.
- DSN1COPY is used to copy data to a table space of a different type. For example, data is copied from a segmented table space to a partition-by-growth table space.
- DSN1COPY is used to copy data to a table space with a different version number or table definition.

Beginning in Db2 12, the REPAIR CATALOG utility enables the following mismatches to be fixed:

- The column data type or length in the table space differs from the catalog definition for the column. If Db2 supports conversion from the data type and length in the table space to the data type and length in the column, REPAIR CATALOG enables conversion of the data type or length of the data in the table space to match the catalog definition the next time that the data is accessed.

In Db2 12, the REPAIR CATALOG TEST utility is enhanced to detect the following mismatches:

- The table space is a partition-by-range table space with absolute page numbering, but the catalog indicates that the table space has relative page numbering.
- The table space is partition-by-range with relative page numbering, but the catalog indicates that the table space has absolute page numbering.
- The table space is a partition-by-range table space with absolute page numbering, but the catalog indicates that the table space has relative page numbering. REPAIR CATALOG changes the catalog definition to match the table space format.
- The number of columns in the table space is greater than the number of columns in the catalog definition of the table.
- The column data type or length in the table space differs from the catalog definition for the column.

Restriction: None of these data validation enhancements apply to LOB and XML table spaces.

Related reference
DSN1COPY (Db2 Utilities)
Syntax and options of the REPAIR control statement (Db2 Utilities)

Alternate copy pools for system-level backups

In Db2 12, you can create extra system-level backups on alternate backup storage groups without affecting the primary system-level backup.

In previous releases, when you used the BACKUP SYSTEM utility to create system-level backups, you could use up to two copy pools only, one for the database and one for logs. These copy pools define the storage groups to copy and the backup storage groups to store the copies.

Beginning in Db2 12, you can also optionally use additional copy pools called alternate copy pools. An alternate copy pool includes the same defined set of storage groups as the standard copy pool; however different backup storage groups are specified. Therefore, you can make system-level backups on different target volumes. These target volumes can be shared among different Db2 subsystems. That is, a volume can be used by another subsystem if it is not part of an active copy. You can make these extra
system-level backups without overwriting the primary production system-level backup or invalidating any versions in the regular backup storage group.

You can specify that you want to use an alternate copy pool when you make the system-level backup. To do so, specify the ALTERNATE_CP option and the related backup storage group options (DBBSG and LGBSG) on the BACKUP SYSTEM utility control statement.

Alternatively, you can specify that you want BACKUP SYSTEM to regularly alternate between using the standard copy pool and the alternate copy pool. For this behavior, use the ALTERNATE_CP subsystem parameter and the related backup storage group subsystem parameters (UTIL_DBBSG and UTIL_LGBSG).

The system-level backups that are made with the alternate copy pool can be used during recovery by the RESTORE SYSTEM utility or the RECOVER utility.

Related concepts
Copy pools (Db2 Utilities)

Related reference
Syntax and options of the BACKUP SYSTEM control statement (Db2 Utilities)
Syntax and options of the RESTORE SYSTEM control statement (Db2 Utilities)
Syntax and options of the RECOVER control statement (Db2 Utilities)
ALTERNATE COPY POOL field (ALTERNATE_CP subsystem parameter) (Db2 Installation and Migration)
DB BACKUP STG GROUP field (UTIL_DBBSG subsystem parameter) (Db2 Installation and Migration)
LOG BACKUP STG GRP field (UTIL_LGBSG subsystem parameter) (Db2 Installation and Migration)

New DSNUTILV stored procedure

In Db2 12, you can use the new DSNUTILV stored procedure to pass utility statements as large as 2 GB. It is otherwise similar to the existing DSNUTILU stored procedure, which cannot pass utility statements larger than 32 KB.

The DSNUTILV stored procedure has the following restrictions:

- Any keywords, names, or constants in the utility statement cannot be larger than 32,704 bytes.
- If you pass a file reference variable, the specified data set that contains the utility statements must be in variable record format.

For detailed information about the requirements, see the description of the utstmt input parameter in the DSNUTILV information.

The DSNUTILV stored procedure is created after new function is activated in Db2 12.

Related reference
DSNUTILV stored procedure (Db2 SQL)

Automatic start of profiles at Db2 start

In Db2 12, you can specify that profiles take effect immediately when Db2 is started.

The PROFILE_AUTOSTART subsystem parameter specifies whether a START PROFILE command is issued automatically when Db2 is started.

Related concepts
Profiles for monitoring and controlling Db2 for z/OS subsystems (Db2 Performance)

Related reference
PROFILE AUTOSTART field (PROFILE_AUTOSTART subsystem parameter) (Db2 Installation and Migration)
-START PROFILE (Db2) (Db2 Commands)
Increased partition sizes and simplified partition management for partition-by-range table spaces with relative page numbering

Db2 12 introduces a new attribute for partition-by-range table spaces featuring partition sizes up to 1 TB and less-disruptive partition management.

Partition-by-range table spaces with relative page numbering allow the following improvements in space allocation:
- Data and index partition sizes can be up to 1 TB.
- Greater flexibility in growing your partitions. With relative page numbering, you can grow partitions by any number of gigabytes. With absolute page numbering partition growth is restricted to gigabytes in powers of two.
- Different partitions of the same table can have different size limits.
- Improved availability because DSSIZE can be increased for individual partitions as an immediate ALTER, without requiring a REORG.

You create table spaces with the new attribute by specifying the PAGENUM RELATIVE option of the CREATE TABLESPACE statement. Existing table spaces can be converted with the ALTER TABLESPACE statement.

Related concepts
Partition-by-range table spaces (Db2 Administration Guide)

Related tasks
Creating table spaces explicitly (Db2 Administration Guide)

Ability to add partitions between existing logical partitions

Db2 12 introduces the ability to add partitions between existing logical partitions in partition-by-range table spaces.

Adding a partition between existing logical partitions is supported only for partition-by-range (UTS) table spaces. The new partition is the next physical partition not being used until the maximum for the table space has been reached. A partition added between existing logical partitions is a pending definition change if the data sets are already defined. Otherwise, the change is immediate.

Related tasks
Adding partitions (Db2 Administration Guide)

Related reference
ALTER TABLE (Db2 SQL)

More efficient point-in-time recovery

In Db2 12, the default behavior for a RECOVER utility job with the TOLOGPOINT option or the TORBA option is to skip unnecessary recoveries. If an object has not been changed since the recovery point, the utility does not recover that object. Therefore, RECOVER can avoid unnecessary work and use fewer resources. This change can also potentially reduce the time that objects are unavailable and improve the overall recovery time.

This new behavior is controlled by the SCOPE option. The default value is SCOPE UPDATE, which specifies that unchanged objects are to be skipped during recovery. However, you can explicitly specify SCOPE ALL to indicate that all objects are to be processed by the RECOVER utility, regardless of whether they have changed since the recovery point.

Related reference
Syntax and options of the RECOVER control statement (Db2 Utilities)
UNLOAD privilege for the UNLOAD utility

A new UNLOAD privilege is introduced Db2 12 to provide better control for executing the UNLOAD utility. In Db2 12, the SELECT privilege is no longer sufficient to execute the UNLOAD utility.

Before the activation of function level 500 or higher, IFCID 404 trace can be used to identify the usage of the SELECT privilege for UNLOAD utility access. The UNLOAD privilege can be granted prior to activation of, but it does not become effective until new function is activated. When new function is activated, the UNLOAD privilege takes effect by default.

After new function activation, if you want the SELECT privilege to be checked for UNLOAD utility access, set the AUTH_COMPATIBILITY system parameter to SELECT_FOR_UNLOAD.

Related tasks
Run premigration queries (DSNTIJPM) (Db2 Installation and Migration)

Related reference
AUTH_COMPATIBILITY in macro DSN6SPRM (Db2 Installation and Migration)
UNLOAD (Db2 Utilities)
GRANT (Db2 SQL)
REVOKE (Db2 SQL)

Temporal versioning for Db2 catalog tables

Db2 12 can use temporal versioning to record historical information for certain catalog tables. You can use the historical information to analyze, predict, and help prevent specific conditions in a subsystem.

The following table lists the catalog tables that have an associated history table for system-period temporal versioning.

<table>
<thead>
<tr>
<th>Catalog table</th>
<th>History table</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSINDEXSPACESTATS catalog table (Db2 SQL)</td>
<td>SYSIBM.SYSIXSPACESTATS_H</td>
</tr>
<tr>
<td>SYSTABLESPACESTATS catalog table (Db2 SQL)</td>
<td>SYSIBM.SYSTABSPACESTATS_H</td>
</tr>
</tbody>
</table>

The historical information for the real-time statistics catalog tables, SYSINDEXSPACESTATS and SYSTABLESPACESTATS, can be used to analyze the rate of change in size, organization, and activity for table space and index space partitions. This information can then be used to develop an automated response to conditions like table spaces and index spaces running out of space or becoming too disorganized.

Enabling the temporal relationship

The temporal relationship between a history table and its associated catalog table must be enabled before the history table can be used to record historical information. Each member Db2 externalizes rows whenever interval specified by the STATSINT subsystem parameter is reached.

Tip: It is best to develop and test an aggregation and purge strategy before enabling the temporal relationship.

To enable the temporal relationship for SYSIBM.SYSINDEXSPACESTATS and its associated history table, SYSIBM.SYSIXSPACESTATS_H, issue the following statement:

```
ALTER TABLE SYSIBM.SYSINDEXSPACESTATS
ADD VERSIONING
USE HISTORY TABLE SYSIBM.SYSIXSPACESTATS_H;
```
To enable the temporal relationship for SYSIBM.SYSTABLESPACESTATS and its associated history table, SYSIBM.SYSTABLESPACESTATS_H, issue the following statement:

```
ALTER TABLE SYSIBM.SYSTABLESPACESTATS
ADD VERSIONING
USE HISTORY TABLE SYSIBM.SYSTABLESPACESTATS_H;
```

If the temporal relationship between a catalog table and its associated history table later needs to be removed, you can issue the ALTER TABLE statement with the DROP VERSIONING clause on the catalog table.

**Related concepts**
Temporal tables and data versioning (Db2 Administration Guide)
Recovery of temporal tables with system-period data versioning (Db2 Administration Guide)
When Db2 externalizes real-time statistics (Db2 Performance)

**Related tasks**
Creating a system-period temporal table (Db2 Administration Guide)
Querying temporal tables (Db2 Administration Guide)
Collecting history statistics (Db2 Performance)

---

**REORG enhancements**

Db2 12 provides enhancements to the REORG utility.

Db2 12 introduces the following improvements to REORG:

**Improved FlashCopy® management**
A REORG with only a FlashCopy inline image copy causes the REORG to fail if the FlashCopy itself fails.

**Improved partition-level partition-by-growth REORGs**
Supports creation of a new PBG partition for overflow rows during a partition-level REORG.

**Prevention of COPY-pending on a LOB table space during REORG of partition-by-growth table spaces**
Avoids leaving COPY pending on LOB table spaces of new partition-by-growth partitions grown during the log phase of REORG.

**REORG-level management of delete of PBG partitions**
Adds a new keyword, DROP_PART, for REORG empty partition-by-growth partition pruning when ZPARM alteration is not feasible.

**Support for the new COMPRESSIONRATIO catalog column**
Adds to LOAD, REORG, and RUNSTATS the ability to gather average compression ratio at the record level instead of the page saved level.

**Additional offload to zIIP**
The RELOAD phase can now be offloaded to zIIP specialty engines.

**New mapping table format support**
New format supports 7-byte RIDs needed with partition-by-range table spaces with relative page numbering.

**REORG against RO pagesets**
You can run REORG with any SHRLEVEL against any RO table space and indexes.

**Display of claimer information**
You can display claimer information on each drain failure, not just last retry.

**Related concepts**
Statistics collection enhancements for SQL performance
Db2 12 provides several statistics collection enhancements, including several of which improve SQL performance.

Related reference
REORG TABLESPACE (Db2 Utilities)

## LOAD enhancements

Db2 12 introduces enhancements to the LOAD utility.

Db2 12 introduces the following improvements to the LOAD utility:

**LOAD PART REPLACE with dummy input against an empty partition-by-range (PBR) partition**
You can now benefit from reduced elapsed and CPU time.

**LOAD SHRLEVEL CHANGE PARALLEL support for partition-by-growth (PBG)**
Removes the single input parallelism restriction for PBG table spaces for LOAD SHRLEVEL CHANGE.

**Additional zIIP offload**
You can offload the RELOAD phase of the LOAD utility, including the data conversion and loading of the record into the page set.

**LOAD RESUME BACKOUT YES to avoid RECP on failure**
Adds a new option on LOAD RESUME SHRLEVEL NONE to allow LOAD to back out the rows already loaded upon encountering an error (such as conversion, LOB/XML, duplicate key, referential integrity violation) without leaving the page set in RECP.

**PREFORMAT support for auxiliary tables**
Support is extended to LOB table spaces and auxiliary indexes.

**Maintain MAXASSIGNEDVAL for identity columns**
LOAD maintains the MAXASSIGNEDVAL for user-provided input.

**LOAD REPLACE support for COMPRESSRATIO column**
Gathers the average compression ratio at the record level instead of the page saved level.

Related concepts
Statistics collection enhancements for SQL performance

Db2 12 provides several statistics collection enhancements, including several of which improve SQL performance.

Automated statistics profiles
Db2 12 introduces the capability for Db2 to manage which statistics to collect based on the queries running on the system.

Related reference
LOAD (Db2 Utilities)

## Backup and recovery enhancements

Db2 12 introduces enhancements to the BACKUP and RECOVERY utilities.

Db2 12 introduces the following improvements to the BACKUP and RECOVERY utilities:

**Point-in-time (PIT) recovery support for PBG table spaces**
You can recover PBG table spaces to a PIT prior to a REORG that materialized these physical pending alters: page size (buffer pool) DSSIZE, SEGSIZE, member cluster.

**FLASHCOPY_PPRCP keyword**
You can specify the FLASHCOPY_PPRCP keyword for the RESTORE SYSTEM and RECOVER utilities, to specify or override the preserve mirror option for PPRC (sync disk mirroring) on the utility statement.

**New default behavior for the PIT RECOVER utility**
Point-in-time recovery is skipped for non-updated pagesets, which saves resources (CPU) and time.
System Level Backup support for multiple copypools
You can keep extra system-level backups on disk during upgrades.

Improved messaging for BACKUP SYSTEM and RESTORE SYSTEM
DFSMShsm messages included in the utility job output for BACKUP SYSTEM and RESTORE SYSTEM provide improved messaging.

COPY option to specify FASTREPLICATION(REQUIRED)
You can control whether FlashCopy is required during creation of the FlashCopy image copy.

New MODIFY RECOVERY option DELETEDS
You can use this to delete image copy data sets.

New MODIFY RECOVERY option NOCOPYPEND
You can use this to specify that MODIFY RECOVERY not set COPY pending.

More accurate validation of recovery scenarios
When you execute MODIFY RECOVERY at the table space level a SYSCOPY record for each partition is inserted, which allows a more accurate validation of recovery scenarios.

Related tasks
Recovering with the BACKUP SYSTEM and RESTORE SYSTEM utilities (Db2 Administration Guide)

Statistics collection enhancements for SQL performance
Db2 12 provides several statistics collection enhancements, including several of which improve SQL performance.

RUNSTATS utility enhancements
Db2 12 introduces the following improvements to RUNSTATS:

Enhanced CLUSTERRATIO formula
Ensures that CLUSERRATIO better reflects dynamic prefetch behavior.

Specifying FREQVAL without the COUNT n keywords
If you specify FREQVAL without the COUNT n keywords, Db2 automatically determines the appropriate number to collect.

Autonomic statistics maintenance
In Db2 12, missing statistics can be automatically identified and included in the set of recommended statistics in the statistics profile. This enhancement frees you from having to manually manage RUNSTATS.

Column group statistics support
Column group statistics are supported with LOAD PARALLEL.

INVALIDATECACHE option to avoid dynamic cache invalidation
You can specify the INVALIDATECACHE keyword to prevent invalidating the dynamic cache (for example, LOAD, REBUILD INDEX, REORG INDEX, REORG TABLESPACE, and RUNSTATS) when collecting statistics and to avoid the unnecessary work of rebinds.

Inline statistics enhancements
Db2 12 introduces the following improvements to inline statistics. The utilities that provide inline statistics are LOAD, REBUILD INDEX, REORG INDEX, and REORG TABLESPACE:

PROFILE support for inline stats in REORG and LOAD
Profiles are now supported with inline statistics to match RUNSTATS functional support. This means that the latest PROFILE settings will be picked up so the appropriate statistics are gathered during LOAD and REORG (thus eliminating an additional execution of RUNSTATS).

Inline stats support for LOAD PARALLEL
You can now specify LOAD PARALLEL to provide inline statistics functional parity with RUNSTATS.
Keywords added to the REBUILD INDEX and REORG INDEX utilities
The MOST, LEAST, and BOTH keywords are added to the REBUILD INDEX and REORG INDEX utilities.

Keywords supported for indexes
The MOST, LEAST, and BOTH keywords are supported for indexes in the LOAD and REORG TABLESPACE utilities.

COLGROUP performance improvement
COLGROUP performance is improved when COLGROUP column specifications are redundant with INDEX columns. The duplicate COLGROUP is ignored in lieu of the index processing.

Related reference
RUNSTATS TABLESPACE syntax and options (Db2 Utilities)
RUNSTATS (Db2 Utilities)

Data sharing enhancements
Db2 12 introduces various data sharing enhancements including peer recovery, improved automatic LPL an GRECP recovery, and asynchronous coupling facility (CF) lock duplexing.
Db2 12 introduces the following data sharing enhancements:

- Db2 Data sharing members can use peer recovery to automatically recover retained locks for a failed member.
- If automatic LPL and GRECP recovery fails, Db2 automatically retries after three minutes, for a total of three retries.
- Asynchronous lock duplexing reduces the overhead of CF lock duplexing.

Related concepts
Peer recovery for data sharing members (Db2 Data Sharing Planning and Administration)
Db2 for z/OS support for asynchronous CF duplexing (Db2 Installation and Migration)

Related reference
Data sharing status descriptions (Db2 Data Sharing Planning and Administration)

Dynamic plan stability
Db2 12 introduces the capability to achieve access path stability comparable to static SQL statements for repeating cached dynamic SQL statements.

When you enable dynamic SQL plan stability, Db2 stores statement cache structures for specified dynamic SQL statements in the Db2 catalog. Whenever a stabilized dynamic SQL statement is not present in the dynamic statement cache when issued, Db2 can load the statement cache structures from the Db2 catalog and avoid the full prepare operation.

Related concepts
Dynamic SQL plan stability (Db2 Performance)

Related tasks
Stabilizing access paths for dynamic SQL statements (Db2 Performance)

Related reference
-START DYNQUERYCAPTURE (Db2) (Db2 Commands)
Automated statistics profiles

Db2 12 introduces the capability for Db2 to manage which statistics to collect based on the queries running on the system.

The Db2 cost-based optimizer relies on statistics about tables and indexes. Prior to Db2 12, it was often the case that only standard or default statistics were gathered, which meant that queries might not perform as well as possible. It was sometimes difficult to know which statistics should be gathered, due to the number of SQL statements, tables, indexes, columns, and keys.

Db2 12 introduces the automatic updating of statistics profiles based on recommendations generated during query optimization, so that RUNSTATS collects all of the statistics recommended by the optimizer.

Related concepts
Statistics profiles (Db2 Performance)

Related tasks
Applying statistics recommendations to statistics profiles automatically (Db2 Performance)
Identifying missing or conflicting statistics (Db2 Performance)

Related reference
The LOG option of the LOAD or REORG utilities (Db2 Administration Guide)

Real-time statistics improvements

Db2 12 introduces more information to aid users in diagnosing problems with real-time statistics collection, and in collecting more accurate statistics.

Those improvements are:
• New messages DSNT535I and DSNT536I, which identify unavailable resources that prevent externalization of real-time statistics.
• The ability to keep statistics history
• The addition of a column that tracks GETPAGEs

Related concepts
How real-time statistics are used by Db2 utilities (Db2 Utilities)

Support for compressing LOB data

Db2 12 supports compressing LOB data when the zEnterprise® data compression (zEDC) hardware is available to the z/OS environment.

Compressing LOB data by using the zEDC hardware can reduce the size of data in a LOB table space to ensure that the LOB table space and its associated base table space remain available for inserting data. LOB data compression can also be enabled on all LOB table spaces that are associated with the Db2 directory.

LOB compression is enabled for individual table spaces by specifying the COMPRESS YES option on the CREATE TABLESPACE or ALTER TABLESPACE statement.

To enable LOB compression for the Db2 directory, set the COMPRESS_DIRLOB subsystem parameter to YES.

The following conditions are required for LOB compression:
• Db2 must be at function level 500 or higher.
• The zEDC hardware must be installed in the system. In a data sharing system, the zEDC hardware should be installed in all data sharing members. System performance can degrade dramatically if a member of the data sharing system that does not have the zEDC hardware accesses compressed LOB data.

• The LOB table space must be associated with a base table that is in a universal table space.

• The total length of the entire LOB must be larger than the defined data page size, otherwise the LOB is not compressed.

Related reference
CREATE TABLESPACE (Db2 SQL)
COMPRESS DB2 DIR LOBS field (COMPRESS_DIRLOB subsystem parameter) (Db2 Installation and Migration)
Overview and planning of zEnterprise Data Compression (zEDC)
Requirements for zEnterprise Data Compression

Db2 changes for the GDPS Continuous Availability with zero data loss solution

To support the GDPS Continuous Availability with zero data loss (GDPS Continuous Availability with zero data loss) solution, Db2 12 introduces subsystem parameters, commands, use of a new data set to support compressed log records, a new REORG TABLESPACE option, enhanced DISPLAY LOG command output, and new IFI qualification field values.

GDPS Continuous Availability with zero data loss solution terminology: A GDPS Continuous Availability with zero data loss solution that includes Db2 requires three data sharing environments. The solution includes a source Db2 data sharing group, a proxy Db2 data sharing group, and a target Db2 data sharing group. The proxy group uses a capture program to capture changes to tables on the source group. The proxy group then uses a replication program to replicate the changes to the target group. A VSAM key-sequenced data set, called the compression dictionary data set (CDDS), must be defined in the source group. It holds the following items for use in capture and replication:

• The expansion dictionaries for Db2 tables whose changes are captured

  Currently, the CDDS contains a maximum of three versions of the expansion dictionaries.

• System status information that the proxy group uses to find the log data sets and to determine the status of the source group members

Subsystem parameters

The following subsystem parameters are added to support this configuration. These subsystem parameters cannot be changed online.

CDDS_MODE
  Specifies whether a member of a Db2 data sharing group is part of a source data sharing group, a proxy data sharing group, or neither. Possible values are SOURCE, PROXY, and NONE.

CDDS_PREFIX
  Specifies a prefix of up to 39 bytes for the CDDS name. The full CDDS name is this prefix, appended with '.CDDS'.

CDDS creation and population

You need to define the CDDS on the source data sharing group. You run the REORG TABLESPACE utility with the INITCDDS and SEARCHTIME options to populate the CDDS.
**Commands for starting and stopping the CDDS**

Before a CDDS can be recovered, it must be closed and deallocated. The -STOP CDDS command lets you close and deallocate a CDDS without stopping all members of a data sharing group. You can then recover the CDDS, and issue the -START CDDS command to allocate and open the CDDS on all members of the data sharing group. You can issue the -START CDDS and -STOP CDDS commands from the source or proxy data sharing group.

**DISPLAY LOG command output changes**

When a data sharing group is enabled as a source or proxy group, the DISPLAY LOG command includes message DSNJ375I, which shows the name of the CDDS, and whether it is enabled in source or proxy mode.

**DSNJU008 (print CDDS) utility**

The DSNJU008 stand-alone utility prints the CDDS. You can filter the output by DBID, PSID, table space partition, or expansion dictionary version.

**IFI qualification changes**

![PSPI](image)

The IFI READS command for IFCID 0306 supports capture of log records. The following values are added to the WQALLCRI field to indicate that log records are being collected by the proxy data sharing group for the source data sharing group.

**X'01' (WQALLCR1)**

Only log records for changed data capture and unit of recovery control from the proxy data sharing group in a GDPS Continuous Availability with zero data loss environment. Records are returned until the end-of-scope log point is reached.

**X'02' (WQALLCR2)**

All types of log records from the proxy data sharing group in a GDPS Continuous Availability with zero data loss environment. Records are returned until the end-of-scope log point is reached.

**X'03' (WQALLCR3)**

Only log records for changed data capture and unit of recovery control from the proxy data sharing group in a GDPS Continuous Availability with zero data loss environment. Records are returned until the end-of-log point is reached for all members of the data sharing group.

**X'04' (WQALLCR4)**

All types of log records from the proxy data sharing group in a GDPS Continuous Availability with zero data loss environment. Records are returned until the end-of-log point is reached for all members of the data sharing group.

![PSPI](image)

**Related tasks**

- Modifying Db2 for the GDPS Continuous Availability with zero data loss solution (Db2 Administration Guide)

**Related reference**

- CDDS_MODE in macro DSN6LOGP (Db2 Installation and Migration)
- CDDS_PREFIX in macro DSN6LOGP (Db2 Installation and Migration)
- -START CDDS (Db2) (Db2 Commands)
- -STOP CDDS (Db2) (Db2 Commands)
- Qualification fields for READS requests (Db2 Performance)
- Syntax and options of the REORG TABLESPACE control statement (Db2 Utilities)
Chapter 4. Query performance in the Db2 12 initial release

The initial release of Db2 12 delivers enhanced query performance.

**Important:** Db2 12 introduces continuous delivery of new capabilities and enhancements in function levels. Most Db2 12 enhancements and changes become available only after the activation of the function level that delivers them. For more information, see Chapter 29, “Activating Db2 12 function levels,” on page 179. Each new or changed item is labeled below with a link to information about the Db2 12 function level that delivered it.

**Related concepts**
- **Dynamic plan stability**
  Db2 12 introduces the capability to achieve access path stability comparable to static SQL statements for repeating cached dynamic SQL statements.
- **Automated statistics collection**
  Db2 12 introduces several enhancements that help to automate the collection of statistics.

**Related information**
- Db2 12 for z/OS Optimizer
- Db2 12 for z/OS Performance Topics (IBM Redbooks)
- IBM Db2 12 for z/OS Technical Overview (IBM Redbooks)

### Automated statistics collection

Db2 12 introduces several enhancements that help to automate the collection of statistics.

In Db2 12, you can specify that statistics profiles are updated based on the statistics recommendations that are generated during the optimization process, which were introduced by Db2 11. Profiles are created or updated based on these recommendations when the STATFDBK_PROFILE subsystem parameter values is set to YES.

If you then specify the USE PROFILE option when you collect statistics, the recommended statistics are collected. Db2 12 also introduces support for the USE PROFILE option when you collect statistics inline when you run the following utilities:

- **LOAD**
- **REBUILD INDEX**
- **REORG INDEX**
- **REORG TABLESPACE**

Db2 12 also introduces recommendations from the optimization process to collect statistics that might have become stale, before problems arise. The recommendations are issued for column, multi-column cardinality, frequency, and histogram statistics. The recommendations are issued when STATSTIME value for a statistic particular statistic is older (by a threshold) than the STATSTIME values for all objects that might collect the same statistic. Like other recommendations from the optimization process, they can be automatically applied to statistics profiles.

Also new in Db2 12, if you include the FREQVAL keyword and omit the COUNT keyword, Db2 automatically determines the number of frequently occurring values to collect so that the data distribution is no longer skewed.

**Related tasks**
- Applying statistics recommendations to statistics profiles automatically (Db2 Performance)
- Controlling the collection of statistics feedback (Db2 Performance)
Static plan stability enhancements

Db2 12 introduces improvements to the usability of static plan stability features.

Db2 12 introduces the following static plan stability enhancements:

- You can now specify that the FREE PACKAGE command free only the previous or original packages copies by specifying PREVIOUS or ORIGINAL for the PLANMGMTSCOPE option.
- If you specify INACTIVE, PREVIOUS, or ORIGINAL for the PLANMGMTSCOPE option, the FREE PACKAGE command can now complete while applications that use the package are still running, removing the necessity of application outages for maintenance of inactive package copies.
- Improved explain information for APREUSE and ACOMPARE for static plan stability (see columns added to DSN_STATEMNT_TABLE).
- Preservation of LASTUSED for BIND REPLACE of the same version.
- REBIND PACKAGE provides a new option APREUSESOURCE to allow APREUSE to use a PREVIOUS or ORIGINAL copy explicitly.
- Support for statement concentration via a BIND parameter
- Support for statement concentration for CREATE and ALTER

Related concepts
Static SQL (Introduction to Db2 for z/OS)
Improvements for the dynamic statement cache
Db2 12 extends the scope of statement concentration with literals.

Related tasks
Saving and switching to previous access paths (Db2 Performance)

Query performance enhancements

Db2 12 introduces performance enhancements for queries that use any of the following: outer joins, UNION ALL, archive transparency, system-period temporal tables.

Db2 12 introduces the following enhancements to query performance:

- Trimming columns from a materialized view or table expression if those columns are not required by the outer query
- Pruning unique LEFT OUTER JOIN views or table expressions if columns are not required in SELECT list
- Pushing join predicates into UNION ALL or outer join query blocks if it is cost effective for the optimizer to do so
• Push ORDER BY and FETCH FIRST into UNION ALL legs to allow each UNION ALL leg to avoid a sort (if possible) and fetch only the required rows for the outer query (for FETCH FIRST)
• Reorder outer join tables to avoid unnecessary materializations
• Reduce situations where workfile usage is required for materialized outer join query blocks or UNION ALL legs

Related concepts
Ways to merge lists of values (Introduction to Db2 for z/OS)
Outer joins (Db2 Application programming and SQL)

Related information
Db2 12 for z/OS Optimizer

User-defined table function performance improvements
The merge capabilities of user-defined table functions are enhanced in Db2 12 to be similar to the capabilities of views.
Db2 12 also introduces indexes on the join or correlation predicates that are passed in as parameters to user-defined table functions to improve performance.

Related concepts
User-defined functions (Db2 SQL)

Related information
Db2 12 for z/OS Optimizer

Improved performance and reliability of index access with list prefetch

Adaptive index is a Db2 12 enhancement to multi-index and single index list prefetch-based plans that introduces logic at execution time to determine the filtering of each index to ensure the optimal execution sequence of indexes, or quicker reversion to table space scan if no filtering index exists.
This enhancement does not require any usage of REOPT bind parameters and therefore avoids any reoptimization overhead at execution time.

Related concepts
Types of indexes (Introduction to Db2 for z/OS)

Related information
Db2 12 for z/OS Optimizer
Chapter 5. OLTP performance in the Db2 12 initial release

The initial release of Db2 12 delivers enhanced online transaction processing (OLTP) performance.

**Important:** Db2 12 introduces continuous delivery of new capabilities and enhancements in function levels. Most Db2 12 enhancements and changes become available only after the activation of the function level that delivers them. For more information, see Chapter 29, “Activating Db2 12 function levels,” on page 179. Each new or changed item is labeled below with a link to information about the Db2 12 function level that delivered it.

**Related information**
- Db2 12 for z/OS Performance Topics (IBM Redbooks)
- IBM Db2 12 for z/OS Technical Overview (IBM Redbooks)

### Improved INSERT performance for unclustered data

Db2 12 introduces insert algorithm 2, which can improve the performance of INSERT operations for unclustered data.

Using insert algorithm 2 can result in an increase in insert throughput, especially with data that is not indexed. Insert algorithm 2 can also reduce logging and reduce class 2 elapsed time and class 2 CPU time.

Insert algorithm 2 applies only to universal table spaces using MEMBER CLUSTER, and is the default algorithm for this table space type. It is not applicable to other table space types.

To set the system-wide setting of the insert algorithm level, use the DEFAULT_INSERT_ALGORITHM subsystem parameter. To control the insert algorithm setting for individual table spaces, specify the INSERT_ALGORITHM option on the CREATE TABLESPACE or ALTER TABLESPACE statement.

**Recommendation:** To determine the group buffer pool size when insert algorithm 2 is used for concurrent INSERT operations from multiple data sharing members, allocate a minimum of 2000 \( n \) directory and data entries (where \( n \) is the number of members).

**Related concepts**
- Insert algorithm 2 (Db2 Performance)
- Member affinity clustering (Db2 Data Sharing Planning and Administration)

**Related reference**
- ALTER TABLESPACE (Db2 SQL)
- CREATE TABLESPACE (Db2 SQL)

### Improved EDM pool management

Db2 12 introduces improved performance for Db2 database systems that have large EDM pools and run multiple, concurrent threads.

These improvements can lead to somewhat higher real storage usage by the database system.

As part of these changes, how EDM pool storage is allocated for certain types of structures has changed.

Previously, the following subsystem parameters determined the amount of storage that was initially allocated in the EDM pool for each type of structure:

- EDMDBDC for the DBD cache
- EDMSTMTC for the dynamic statement cache
• EDM_SKELETON_POOL for the EDM skeleton pool

After the initial allocation, the values could be increased or decreased, but the values could never be less than the values that were specified at installation time.

In Db2 12, no storage is initially allocated for these structures. Storage is allocated as it is needed. When the values are reached, and more storage is needed, structures that are not currently in use are removed to keep the storage that is in use below the specified values.

**Related reference**

EDM DBD CACHE field (EDMDBDC subsystem parameter) (Db2 Installation and Migration)
EDM STATEMENT CACHE field (EDMSTMTC subsystem parameter) (Db2 Installation and Migration)
EDM SKELETON POOL SIZE field (EDM_SKELETON_POOL subsystem parameter) (Db2 Installation and Migration)

---

**In-memory buffer pools**

In Db2 12, you can assign objects to in-memory buffer pools. The data pages for the object remain in the buffer pool if they can fit within the size of the buffer pool.

**Introductory concepts**

Buffer pools (Introduction to Db2 for z/OS)
The role of buffer pools in caching data (Introduction to Db2 for z/OS)

By assigning frequently accessed objects to in-memory buffer pools, you can reduce the cost of internal processing and I/O for those objects.

In-memory buffer pools are created by the PGSTEAL(NONE) option of the ALTER BUFFERPOOL command. Db2 implicitly creates an overflow area as part of the bufferpool. Objects that cannot fit within the in-memory part of the buffer pool are allocated to the overflow area, where page-stealing is possible. When you specify PGSTEAL(NONE), it is best to specify a buffer pool size that is large enough to contain all assigned objects that might be open at the same time. Pages that are allocated to the overflow area cannot be returned to the main area of the bufferpool until the object is closed.

Db2 issues a console message whenever an object is allocated to the overflow area for in-memory buffer pool.

**Related tasks**
Choosing a page-stealing algorithm (Db2 Performance)

**Related reference**
ALTER BUFFERPOOL (Db2) (Db2 Commands)

**Related information**
DSNB604I (Db2 Messages)

---

**Improvements for the dynamic statement cache**

Db2 12 extends the scope of statement concentration with literals.

Db2 10 added an enhancement to dynamic statement caching that allowed a Db2 database manager to exclude the literal constants that are in dynamic SQL statements when it searched for a statement text match in the dynamic statement cache. This enhancement could be specified at the statement level, by including CONCENTRATE STATEMENTS WITH LITERALS in the ATTRIBUTES clause of the PREPARE SQL statement. For ODBC and JDBC applications, the statement could also be specified at the data source level through connection properties.

Db2 12 expands the scope of literal concentration enablement. You can specify statement concentration at the package level, through the CONCENTRATESTMT bind option, and at the routine or advanced trigger level, through the CONCENTRATE STATEMENTS WITH LITERALS option.
**Improved random index access**

Db2 12 introduces *fast index traversal* (sometimes called "fast traverse blocks" or "FTB"), which can improve overall random index access, especially for indexes used predominantly for read access.

Fast index traversal provides improved performance for indexes with frequent lookups.

In Db2 12, fast index traversal is available in function level 100. However, during coexistence with Db2 11, no storage is allocated for fast index traversal of group buffer pool dependent indexes.

**Related concepts**
Fast index traversal (Db2 Performance)

**Related tasks**
Enabling or disabling fast index traversal at the index level (Db2 Performance)

**Related reference**
INDEX MEMORY CONTROL field (INDEX_MEMORY_CONTROL subsystem parameter) (Db2 Installation and Migration)

---

**Optimized scheduling of dynamic prefab**

Db2 12 reduces the scheduling of unnecessary prefetch for OLTP and query workloads.

With this feature, Db2 12 addresses the situation that arises when all pages are in memory and dynamic prefetch needlessly schedules a prefetch. The result is a saving of CPU cycles.

**Related concepts**
Dynamic prefetch (PREFETCH='D') (Db2 Performance)

---

**Reduced log write latency**

With z/OS HyperWrite, Db2 12 reduces log write latency, resulting in improved transaction latency.

With this feature, Db2 12 offers a significant opportunity for accelerating transaction execution.

**Related concepts**
Dynamic prefetch (PREFETCH='D') (Db2 Performance)
Support for estimating benefits of increasing buffer pool size

Db2 12 provides the buffer pool advisory mode, which simulates the effect on system performance of increasing the size of buffer pools.

With this feature, Db2 Db2 12 enables you to accurately determine how I/O and CPU usage would be affected by increasing buffer pool size.

Related concepts
Buffer pools (Introduction to Db2 for z/OS)

Scalability enhancements

Db2 12 provides several improvements that improve large n-way scaling.

The scalability enhancements introduced in Db2 Db2 12 include:

- Improved efficiency on LPARs with high number of CPs
- Log latch contention reduction
- Bufferpool scaling improvements
- IRLM latch contention reduction
- Increased number of prefetch engines
- Support for > 1 TB buffer pools (sum of VPSIZE and SPSIZE can be up to 16TB)

Related concepts
Db2 data servers and environments (Introduction to Db2 for z/OS)

Increased size of active log data sets

Db2 12 enables support for >4 GB active log data sets.

For Db2 Db2 12, the maximum active log data set is 768 GB.

Adding new active log data set > 4 GB can be done by:

- DSNJU003 stand-alone utility or
- -SET LOG NEWLOG

Related reference
DSNJU003 (change log inventory) (Db2 Utilities)
-SET LOG (Db2) (Db2 Commands)

Instrumentation enhancements

Db2 12 provides several instrumentation enhancements.

These enhancements for Db2 Db2 12 include:

- More granular wait times for IFCIDs 316 (dynamic) and 401 (static)
- Enhanced IFCIDs 53/58 statement level section for PREPARE (similar to INSERT/UPDATE/DELETE)
- Addition of section number in IFCIDs 53/58
- Addition of batch job STEP name in correlation header
- Addition of REFRESH TABLE to counts in DSNDQXST
- Measurement of cache + catalog hit ratio to recognize reduced full prepare due to dynamic plan stability
Related concepts
Instrumentation facilities (IF) (Diagnosing Db2 problems)

Integration with IBM DS8870 Easy Tier Multi-Temperature Management

Db2 12 enhances integration with DS8870 Easy Tier®.

Easy Tier is a “multi-temperature” data placement solution that automatically puts frequently accessed data on SSDs, and less-frequently accessed data on HDDs.

Related concepts
Storage servers (Db2 Performance)
Part 2. What's new in Db2 12 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:** Get the latest news about new capabilities and enhancements in Db2 for z/OS, from the IBM experts who design, build, test, and support Db2. For details, see News about Db2 for z/OS from the IBM lab.

Although most new capabilities and enhancements in Db2 12 are delivered in function levels. Some continue to be delivered in the base release. For highlighted examples of such enhancements, see .

The following function levels are available in Db2 12. They are listed in descending order, beginning with the highest available function level:

**Related concepts**
What's new in the Db2 12 base release
Db2 12 for z/OS takes Db2 to a new level, both extending the core capabilities and empowering the future. Db2 12 extends the core with new enhancements to scalability, reliability, efficiency, security, and availability. Db2 12 also empowers the next wave of applications in the cloud, mobile, and analytics spaces.

**Related tasks**
Adopting new capabilities in Db2 12 continuous delivery
In Db2 12, function levels and application compatibility levels control the adoption of most new capabilities by Db2 subsystems and Db2 applications.

**Related information**
Exploring IBM Db2 for z/OS Continuous Delivery (IBM Redpaper)
Chapter 6. Function level 505 (activation enabled by APAR PH09191—June, 2019)

Function level 505 introduces improved concurrency for rebind of executing packages, automatic page sample by default for RUNSTATS, improved DECIMAL data type support, transparent encryption for data columns, and temporal and archive transparency for WHEN clauses on triggers.

Contents

“Improved Hybrid Transactional Analytical Processing performance” on page 61
“Rebind phase-in for packages that are being used for execution” on page 61
“New built-in functions for encryption and decryption with key labels” on page 61
“Improved support for DECIMAL columns” on page 62
“Improved RUNSTATS performance with automatic page sampling by default” on page 62
“Temporal and archive transparency support for WHEN clause on triggers” on page 62
“Activation details for function level 505” on page 62
“Function level 505 incompatible changes” on page 63

Improved Hybrid Transactional Analytical Processing performance

Performance improvements address the perceived latency between Db2 for z/OS and IBM Db2 Analytics Accelerator in Hybrid Transactional Analytical Processing (HTAP) environments. APAR PI98630 delivered the functional code.

Rebind phase-in for packages that are being used for execution

Function level 505 introduces support for rebind phase-in, an enhancement that allows Db2 to rebind a package concurrently with execution of the package. To allow concurrency, a rebind operation creates a new copy of the package. When the rebind operation finishes, new threads can use the new package copy immediately, and existing threads can continue to use the copy that was in use prior to the rebind (the phased-out copy) without disruption.

This enhancement also enables switching to previous access paths and runtime structures gradually (the switch to the previous access path is phased-in), which allows regression recovery without incurring an application outage.

For more information, see Package copies for plan management (Db2 Performance).

APAR PH12186 delivered the functional code for new rebind phase-in capability.

New built-in functions for encryption and decryption with key labels

Function level 505 introduces new built-in encrypt and decrypt data key functions to simplify data protection. The new built-in function ENCRYPT_DATAKEY (Db2 SQL) converts a block of plain text to a block of encrypted text using a specified algorithm and key label. The new built-in DECRYPT_DATAKEY functions return the block of encrypted text from ENCRYPT_DATAKEY as a block of the datatype specified by the function used.

APAR PH09506 delivered the functional code for the new built-in encryption and decryption functions.
Improved support for DECFLOAT columns

Function level 505 introduces improvements to the support for DECFLOAT columns to improve performance of mission critical applications. These improvements include the ability to specify DECFLOAT columns in an index and as a key in a primary or unique key.

For more information about using DECFLOAT columns in primary and unique keys, see CREATE TABLE (Db2 SQL) and ALTER TABLE (Db2 SQL).

For more information about using DECFLOAT columns in indexes, including changes to the maximum length of an index key, see CREATE INDEX (Db2 SQL) and ALTER INDEX (Db2 SQL).

APAR PH09797 delivered the functional code for the improved support of DECFLOAT columns.

Improved RUNSTATS performance with automatic page sampling by default

Function level 505 changes the meaning of the default value of the STATPGSAMP subsystem parameter to mean the same as YES, which means that RUNSTATS uses page-level sampling by default for universal table spaces. STATPGSAMP was introduced in Db2 12 by APAR PH07220. The default value is SYSTEM.

In function level 504 and lower, SYSTEM has the same meaning as NO. That is, RUNSTATS does not use page sampling by default. Any sampling is determined by the options that the RUNSTATS statement specifies.

For more information, see PAGE-LEVEL SAMPLING field (STATPGSAMP subsystem parameter) (Db2 Installation and Migration) and RUNSTATS TABLESPACE syntax and options (Db2 Utilities).

APAR PH07220 delivered the functional code for the new page default page sampling behavior.

Temporal and archive transparency support for WHEN clause on triggers

Function level 505 allows system-period temporal tables and archive-enabled tables to be referenced in WHEN clauses of both basic and advanced triggers, regardless of the settings of the SYSTIMESENSITIVE and ARCHIVESENSITIVE bind options. Time machine and transparent archive data retrieval are fully supported in the WHEN clause.

APAR PH09794 delivered the functional code for the temporal and archive transparency support for WHEN clause on triggers.

Activation details for function level 505

To learn how to make new capabilities available for use in your Db2 environment, see Part 4, “Adopting new capabilities in Db2 12 continuous delivery,” on page 167.

Table 3. Function level 505 activation details

<table>
<thead>
<tr>
<th>Enabling APAR:</th>
<th>APAR PH09191</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum Db2 catalog level:</td>
<td>V12R1M505</td>
</tr>
<tr>
<td>Catalog level V12R1M505 changes:</td>
<td>A new COPYID column is added in the following tables:</td>
</tr>
<tr>
<td></td>
<td>• SYSIBM.SYSPACKAGE</td>
</tr>
<tr>
<td></td>
<td>• SYSIBM.SYSPACKCOPY</td>
</tr>
<tr>
<td></td>
<td>• SYSIBM.SYSPACKDEP</td>
</tr>
<tr>
<td></td>
<td>• SYSIBM.SYSPACKSTMT</td>
</tr>
</tbody>
</table>
Table 3. Function level 505 activation details (continued)

Application compatibility control: Applications must run at application compatibility level V12R1M505 or higher to use the following new capabilities:

- All built-in functions described in “New built-in functions for encryption and decryption with key labels” on page 61.
- Creating a new index with a DECIMAL column as a key, or adding a DECIMAL column to an existing index.
- Defining a primary or unique constraint with a DECIMAL column.

Function level 505 incompatible changes

Function level 505 introduces no incompatible changes.

If your environment contains existing user-defined functions with the same names and signatures as the new built-in functions, applications with unqualified references to the existing user-defined functions might start invoking the built-in function instead of the user-defined function in certain situations. For information about avoiding such situations, see Ensuring that Db2 executes the intended user-defined function (Db2 Application programming and SQL).
Chapter 7. Function level 504 (activation enabled by APAR PH07672—April, 2019)

Function level 504 introduces new support for data with IBM Z hardware-based Huffman compression of Db2 data, the ability to prevent the creation of certain new deprecated objects types, new support for certain built-in functions by pass-through to IBM Db2 Analytics Accelerator, and new SQL syntax alternatives for certain special registers and NULL predicates.

Contents

“IBM z14 Huffman compression” on page 65
“Prevent creation of new deprecated objects” on page 65
Newly supported built-in functions with IBM Db2 Analytics Accelerator
“New SQL syntax alternatives for special registers and NULL predicates” on page 67
“Activation details for function level 504” on page 68
“Function level 504 incompatible changes” on page 68

IBM z14 Huffman compression

Function level 504 introduces support for compression of Db2 data with IBM Z hardware-based entropy encoding (Huffman) compression with the IBM z14™ Compression Coprocessor (CMPSC).

The TS_COMPRESSION_TYPE subsystem parameter controls the compression method for Db2 subsystems on IBM z14 hardware with Huffman compression enabled. Huffman compression is only available for data in universal table spaces, regardless of the setting of the TS_COMPRESSION_TYPE subsystem parameter.

For more information, see TS_COMPRESSION_TYPE field (TS_COMPRESSION_TYPE subsystem parameter) (Db2 Installation and Migration) and Using Huffman compression to compress your data (Db2 Performance).

APAR PH04424 delivered the functional code that supports IBM z14 Huffman compression.

Prevent creation of new deprecated objects

Certain Db2 capabilities are deprecated. That is, although they remain supported, they have been replaced by other capabilities, their continued use is not recommended, and support for them is likely to be removed in the future. For a full list of such capabilities in Db2 12, see Chapter 24, “Deprecated function in Db2 12,” on page 161.

Function level 504 introduces capability to prevent the creation of certain deprecated objects in your Db2 for z/OS environments. Starting in function level 504, SQL statement in packages that are bound with APPLCOMPAT(V12R1M504) or higher, or with comparable SQL options in effect, are prevented from creating following types of deprecated objects:

Synonyms

The CREATE SYNONYM statement results in an error. Use aliases instead of synonyms. For more information about aliases, see Aliases (Introduction to Db2 for z/OS) and CREATE ALIAS (Db2 SQL).

Segmented (non-UTS) or partitioned (non-UTS) tables spaces

The result of a CREATE TABLESPACE statement that specifies the NUMPARTS clause but no MAXPARTITIONS clause is a partition-by-range table space table space. Otherwise, the result is a partition-by-growth table space.
If no SEGSIZE clause is specified, Db2 uses the DPSEGSZ subsystem parameter value. For more information, see DEFAULT PARTITION SEGSIZE field (DPSEGSZ subsystem parameter) (Db2 Installation and Migration).

Table 4. CREATE TABLESPACE clauses for specifying table space types, by application compatibility level.

<table>
<thead>
<tr>
<th>Table space type</th>
<th>APPLCOMPAT(V12R1M504) and higher</th>
<th>APPLCOMPAT(V12R1M503) and lower</th>
</tr>
</thead>
<tbody>
<tr>
<td>Partition-by-growth</td>
<td>Any of the following combinations:</td>
<td>Any of the following combinations:</td>
</tr>
<tr>
<td></td>
<td>• MAXPARTITIONS and NUMPARTS</td>
<td>• MAXPARTITIONS and NUMPARTS</td>
</tr>
<tr>
<td></td>
<td>• MAXPARTITIONS</td>
<td>• MAXPARTITIONS and SEGSIZE</td>
</tr>
<tr>
<td></td>
<td>• Omit both</td>
<td>n&quot;1&quot; on page 66</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• MAXPARTITIONS</td>
</tr>
<tr>
<td>Partition-by-range</td>
<td>NUMPARTS only</td>
<td>NUMPARTS and SEGSIZE n&quot;1&quot; on page 66</td>
</tr>
<tr>
<td>Segmented (non-UTS)</td>
<td>Not supported n&quot;2&quot; on page 66</td>
<td>One of the following combinations:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SEGSIZE n&quot;1&quot; on page 66</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Omit MAXPARTITIONS, NUMPARTS, and SEGSIZE</td>
</tr>
<tr>
<td>Partitioned (non-UTS)</td>
<td>Not supported n&quot;2&quot; on page 66</td>
<td>NUMPARTS and SEGSIZE 0</td>
</tr>
</tbody>
</table>

Notes:

1. Where n is a non-zero value. The DPSEGSZ subsystem parameter determines the default value. For more information, see DEFAULT PARTITION SEGSIZE field (DPSEGSZ subsystem parameter) (Db2 Installation and Migration).

2. (FL 504) Non-UTS table spaces for base tables are deprecated. In general, for packages bound with APPLCOMPAT(V12R1M504) or higher, the result of a CREATE TABLESPACE statement is always a partition-by-growth or partition-by-range UTS table space, and a CREATE TABLE statement that specifies a non-UTS table space, including existing multi-table segmented table spaces, returns an error. The only exception is a dynamic CREATE TABLESPACE statement that is executed after the CURRENT APPLICATION COMPATIBILITY special register is set to a value lower than V12R1M504. In this case, CREATE TABLESPACE creates a non-UTS table space, and a CREATE TABLE statement can specify a non-UTS table space. Existing tables in non-UTS table spaces remain supported. However, support is likely to be removed in the future.

Also, a CREATE TABLE statement to create a new table in a segmented (non-UTS) table space results in an error.

For more information, see Table space types and characteristics in Db2 for z/OS (Db2 Administration Guide) and CREATE TABLESPACE (Db2 SQL).

Hash-organized tables

With the release of function level 504, hash organized tables are now considered deprecated. CREATE TABLE or ALTER TABLE statement that specifies ORGANIZE BY HASH results in an error. Newer capabilities such as fast index traversal in Db2 12 offer a more effective solution than hash-organized tables. For more information, see Fast index traversal (Db2 Performance).

Tip: Application compatibility operates at the package level. So, any of these deprecated object types can still be created in function level 504 and higher if absolutely necessary (such as for recovery situations). To create the objects, you can bind packages at V12R1M503 or lower, or issue the following statement:

```sql
SET CURRENT APPLICATION COMPATIBILITY = 'V12R1M503'
```
For remote applications, you can avoid application changes by using the DSN_PROFILE_TABLE to set the special register value. For more information see SET CURRENT APPLICATION COMPATIBILITY (Db2 SQL) and Setting special registers with profiles (Db2 Administration Guide).

APAR PH02873 delivered the functional code that supports preventing the creation of the deprecated objects.

**Newly supported built-in functions with IBM Db2 Analytics Accelerator**

Function level 504 introduces support for the following passthrough-only built-in functions, which are passed through from Db2 for z/OS to IBM Db2 Analytics Accelerator:

- CUME_DIST
- CUME_DIST (aggregate) (Db2 SQL)
- FIRST_VALUE
- LAG
- LAST_VALUE
- LEAD
- NTH_VALUE
- NTILE
- PERCENT_RANK
- PERCENT_RANK (aggregate) (Db2 SQL)
- RATIO_TO_REPORT
- REGEXP_COUNT (Db2 SQL)
- REGEXPInstr (Db2 SQL)
- REGEXP_LIKE (Db2 SQL)
- REGEXP_REPLACE (Db2 SQL)
- REGEXP_SUBSTR (Db2 SQL)

Db2 for z/OS only verifies that the data types of the parameters are valid for the functions. The accelerator engine does all other function resolution processing and validation. For more information, see Accelerating queries with passthrough-only expressions.

APAR PH00224 delivered the functional code for the new support of these passthrough-only built-in functions.

**New SQL syntax alternatives for special registers and NULL predicates**

Function level 504 introduces new syntax alternatives in Db2 for z/OS to simplify the porting of database applications from other platforms. For example, support is introduced in Db2 for z/OS for the following alternative syntax for referencing existing special registers. For more information, see Special registers (Db2 SQL).

<table>
<thead>
<tr>
<th>Existing Special Register</th>
<th>New Syntax Alternative</th>
<th>Number of Keywords for New Syntax Alternative</th>
</tr>
</thead>
<tbody>
<tr>
<td>CURRENT CLIENT_ACCTNG</td>
<td>CLIENT ACCTNG</td>
<td>2</td>
</tr>
<tr>
<td>CURRENT CLIENT_APPLNAME</td>
<td>CLIENT APPLNAME</td>
<td>2</td>
</tr>
<tr>
<td>CURRENT CLIENT_USERID</td>
<td>CLIENT USERID</td>
<td>2</td>
</tr>
<tr>
<td>CURRENT CLIENT_WRKSTNNNAME</td>
<td>CLIENT WRKSTNNNAME</td>
<td>2</td>
</tr>
</tbody>
</table>
### Table 5. New syntax alternatives for special registers in function level 504 (continued)

<table>
<thead>
<tr>
<th>Existing Special Register</th>
<th>New Syntax Alternative</th>
<th>Number of Keywords for New Syntax Alternative</th>
</tr>
</thead>
<tbody>
<tr>
<td>CURRENT SERVER</td>
<td>CURRENT_SERVER</td>
<td>1 “1” on page 68</td>
</tr>
<tr>
<td>CURRENT TIME ZONE</td>
<td>CURRENT_TIMEZONE</td>
<td>1 “1” on page 68</td>
</tr>
</tbody>
</table>

**Notes:**

1. The new single-token syntax alternatives might introduce an incompatible change if existing data or applications use a column or variable with the same name as the single token.

The following table shows the new syntax alternatives for the NULL predicate that function level 504 introduces in Db2 for z/OS. For more information, see NULL predicate (Db2 SQL).

### Table 6. New syntax alternatives for the NULL predicate in function level 504

<table>
<thead>
<tr>
<th>Existing Predicate</th>
<th>New Syntax Alternatives</th>
</tr>
</thead>
<tbody>
<tr>
<td>IS NULL</td>
<td>ISNULL</td>
</tr>
<tr>
<td>IS NOT NULL</td>
<td>NOTNULL</td>
</tr>
</tbody>
</table>

APAR PH02646 delivered the functional code to support the new SQL syntax alternatives.

**Activation details for function level 504**

To learn how to make new capabilities available for use in your Db2 environment, see Part 4, “Adopting new capabilities in Db2 12 continuous delivery,” on page 167.

### Table 7. Function level 504 activation details

<table>
<thead>
<tr>
<th>Enabling APAR:</th>
<th>APAR PH07672.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum Db2 catalog level:</td>
<td>V12R1M503</td>
</tr>
<tr>
<td>Application compatibility control:</td>
<td>Applications must run at application compatibility level V12R1M504 or higher to use the following new capabilities:</td>
</tr>
<tr>
<td></td>
<td>• All new SQL syntax described in “New SQL syntax alternatives for special registers and NULL predicates” on page 67.</td>
</tr>
<tr>
<td></td>
<td>• All built-in functions described in “Newly supported built-in functions with IBM Db2 Analytics Accelerator” on page 67.</td>
</tr>
<tr>
<td></td>
<td>• Preventing the creation of deprecated object types, as described in “Prevent creation of new deprecated objects” on page 65.</td>
</tr>
</tbody>
</table>

**Function level 504 incompatible changes**

Function level 504 introduces no incompatible changes. However, be aware of changes that might impact your Db2 environment and applications for packages bound at APPLCOMPAT(V12R1M504) or higher, such as the following changes:

• If your Db2 applications have unqualified references to existing user-defined functions, they might start invoking new built-in functions instead if the names and signatures match, in certain situations. For information about avoiding such situations, see Ensuring that Db2 executes the intended user-defined function (Db2 Application programming and SQL).
• If your Db2 environment contains existing columns or variables with names such as CURRENT_TIMEZONE or CURRENT_SERVER, you must modify your applications to delimit these names. For example, you can issue the following queries to identify columns and variables with these names. For columns, issue:

```sql
SELECT * FROM SYSIBM.SYSCOLUMNS
WHERE NAME IN('CURRENT_SERVER','CURRENT_TIMEZONE');
```

For variables, issue:

```sql
SELECT * FROM SYSIBM.SYSVARIABLES
WHERE NAME IN('CURRENT_SERVER','CURRENT_TIMEZONE');
```

Based on the result, you can then check for dependencies, for example by querying the SYSIBM.SYSPACKDEP catalog table, to identify any static applications that depend on the nondelimited forms of these names.

• CREATE TABLESPACE statements that previously created non-UTS segmented or partitioned table spaces, now always create partition-by-growth or partition-by-range table spaces.

• The following SQL statements now return errors:
  – CREATE SYNONYM statements
  – CREATE TABLE statements that specify existing non-UTS table spaces
  – CREATE TABLE and ALTER TABLE statements that specify ORGANIZE BY HASH
Chapter 8. Function level 503 (activation enabled by APAR PH00506—October, 2018)

Function level 503 introduces support for IBM Db2 AI for z/OS, replication of system-period temporal tables and generated expression columns, a change to temporal auditing support for temporal data, and a new console message that indicates when catalog levels or function levels change.

Contents

“IBM Db2 AI for z/OS support” on page 71
“Enablement for replication of system-period temporal tables and generated expression columns” on page 71
“Temporal auditing” on page 72
“Console message for catalog level or function level change” on page 71
“Activation details for function level 503” on page 72
“Function level 503 incompatible changes” on page 72

IBM Db2 AI for z/OS support

Function level 503 introduces support for IBM Db2 AI for z/OS, which leverages machine learning technology to empower the Db2 for z/OS optimizer to determine the best-performing query access paths, based on your workload characteristics.

Update: With IBM Db2 AI for z/OS Version 1.1.0.1 or higher and APAR PH05323, Db2 for z/OS now supports IBM Db2 AI for z/OS at function level 500 or higher.

For more information, see Overview of IBM Db2 AI for z/OS.

New Db2 commands are added in support of IBM Db2 AI for z/OS. For more information about the new commands, see:

-START ML (Db2) (Db2 Commands)
-STOP ML (Db2) (Db2 Commands)
-DISPLAY ML (Db2) (Db2 Commands)

Enablement for replication of system-period temporal tables and generated expression columns

Function level 503 provides support needed by replication programs to replicate Db2 for z/OS data in system-period temporal tables and in tables that contain generated expression columns.

A new built-in global variable, SYSIBMADM.REPLICATION_OVERRIDE, provides support needed to replicate system-period temporal tables or in tables that contain generated expression columns. The global variable is intended for use on the apply side of the replication process by products that enable replication of Db2 data. Applications that set the SYSIBMADM.REPLICATION_OVERRIDE global variable must be bound using APPLCOMPAT V12R1M503.

See Built-in global variables (Db2 SQL) for more information.

Console message for catalog level or function level change

Function level 503 introduces console message DSN014I. It is issued to the console whenever the function level or catalog level of a Db2 subsystem or data sharing group changes. See DSN014I (Db2 Messages).
Temporal auditing

When a system-period temporal table that contains one or more rows of data is altered to add a DATA CHANGE OPERATION generated expression column (also called an auditing column), Db2 assigns null values to the new column in the existing rows. If the table was also defined with the ON DELETE ADD EXTRA ROW attribute, prior to function level 503, temporal queries that were run on these types of tables did not return any rows that contained a null value that corresponds to the DATA CHANGE OPERATION column. As a result, the data that was returned did not include the historical data that was expected.

With function level 503, rows that contain null values in the history table column that corresponds to the DATA CHANGE OPERATION column will now be considered part of the intermediate result set for a system-period temporal query. However, rows that were added to a history table for the ON DELETE ADD EXTRA ROW attribute will still be excluded.

For information about the auditing capabilities of Db2, see Scenario for tracking auditing information (Db2 Administration Guide).

APAR PI95480 delivered functional code to support the new temporal auditing capability.

Activation details for function level 503

To learn how to make new capabilities available for use in your Db2 environment, see Part 4, “Adopting new capabilities in Db2 12 continuous delivery,” on page 167.

Table 8. Function level 503 activation details

<table>
<thead>
<tr>
<th>Enabling APAR:</th>
<th>PH00506</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum Db2 catalog level:</td>
<td>V12R1M503</td>
</tr>
<tr>
<td>Catalog level V12R1M503 changes:</td>
<td>The REPLICATION_OVERRIDE global variable is added. For a summary of changes in all Db2 12 catalog levels, see Chapter 17, “Catalog changes in Db2 12,” on page 127.</td>
</tr>
<tr>
<td>Application compatibility control:</td>
<td>Applications must run at application compatibility level V12R1M503 or higher to use the following new capabilities:</td>
</tr>
<tr>
<td></td>
<td>• The SYSIBMADM.REPLICATION_OVERRIDE global variable for replication of system-period temporal tables and generated expression columns.</td>
</tr>
<tr>
<td></td>
<td>• The temporal query result change for system-period temporal tables defined with the ON DELETE ADD EXTRA ROW attribute that also contain a DATA CHANGE OPERATION column.</td>
</tr>
</tbody>
</table>

Function level 503 incompatible changes

Function level 503 introduces the following incompatible changes.

Result change for system-period temporal tables defined with ON DELETE ADD EXTRA ROW

A query has incompatible behavior at application compatibility level V12R1M503 if all of the following conditions are met:

• It references a system-period temporal table (STT) with FOR SYSTEM_TIME FROM value1 TO value2 (or BETWEEN value1 AND value2) period specification. The STT can be directly referenced or indirectly referenced through views.
• The STT is defined with the ON DELETE ADD EXTRA ROW clause.
• The STT is defined with the DATA CHANGE OPERATION column.
• During migration to use the temporal auditing feature, the altered table contains existing rows.
The DATA CHANGE OPERATION column is populated with the default NULL value for the existing rows. When the query is executed, the historical table contains records with a NULL value in DATA CHANGE OPERATION column.

**Action required:** Start a trace for IFCID 0376. In the trace output, function code 1215031 or 1202 identifies the affected queries.

**The SET_MAINT_MODE_RECORD_NO_TEMPORALHISTORY stored procedure is not supported when APPLCOMPAT is V12R1M503 or higher**

Users will not be able to invoke the SYSPROC.SET_MAINT_MODE_RECORD_NO_TEMPORALHISTORY stored procedure when running with APPLCOMPAT=V12R1M503 or higher. Existing applications that invoke the SYSPROC.SET_MAINT_MODE_RECORD_NO_TEMPORALHISTORY stored procedure will fail starting with APPLCOMPAT=V12R1M503 or higher.

**Action required:** Start a trace for IFCID 0376. In the trace output, function code 1215032 or 1203 identifies applications that use the SYSPROC.SET_MAINT_MODE_RECORD_NO_TEMPORALHISTORY stored procedure. Modify such applications to set value of the SYSIBMADM.REPLICATION_OVERRIDE built-in global variable to 'Y' instead.

**Related information:**

SYSIBMADM.REPLICATION_OVERRIDE built-in global variable

For all incompatible changes in Db2 12, see Chapter 13, “Incompatible changes in Db2 12,” on page 87
Chapter 9. Function level 502 (activation enabled by APAR PI95511—May, 2018)

Function level 502 introduces management of key labels for z/OS DFSMS data set encryption and explicit casting of numeric values to GRAPHIC or VARGRAPHIC.

Contents

“Key label management for z/OS DFSMS data set encryption” on page 75
“Explicit casting of numeric values to GRAPHIC or VARGRAPHIC” on page 75
“Db2 12 migration process changes” on page 76
“Activation details for function level 502” on page 76
“Function level 502 incompatible changes” on page 77

Key label management for z/OS DFSMS data set encryption

Function level 502 introduces management of key labels for z/OS DFSMS data set encryption to transparently encrypt Db2 data sets. DFSMS can be used to encrypt various types of Db2 data sets including Db2-managed table space and index space data sets, data sets that are used by Db2 utilities, and sequential input and output data sets.

Encrypting log, catalog, and directory data sets with z/OS DFSMS data set encryption (Managing Security)
Encrypting all data sets using a storage group with z/OS DFSMS data set encryption (Managing Security)
Using z/OS DFSMS data set encryption to encrypt the data sets associated with a particular table (Managing Security)
Changing Db2 data sets from encrypted to unencrypted (Managing Security)
Encrypted FlashCopy image copies, copies made with DFSMSdss concurrent copy, and system-level backups (Managing Security)
Sample procedure for setting up z/OS DFSMS data set encryption (Managing Security)

APAR PI90288 delivered the functional code to support key label management for z/OS DFSMS data set encryption.

Explicit casting of numeric values to GRAPHIC or VARGRAPHIC

Function level 502 introduces casting an explicit numeric value to a graphic string value. All of the numeric data types (SMALLINT, INTEGER, BIGINT, DECIMAL, REAL, DOUBLE, FLOAT, and DECFLOAT) are supported. You can use the GRAPHIC or VARGRAPHIC built-in functions and the CAST specification for the explicit cast.

When the GRAPHIC or VARGRAPHIC functions are used to explicitly cast a decimal value to graphic, an optional decimal separator character can be specified as a single-byte character constant for the second argument. The extended support for the GRAPHIC and VARGRAPHIC built-in functions to allow numeric input data is similar to the existing support for numeric input to the CHAR and VARCHAR built-in functions.

Whether the CAST specification or the GRAPHIC or VARGAPGHIC functions are used, the result is Unicode (UTF-16), and the context must support Unicode data.

CAST specification (Db2 SQL)
GRAPHIC (Db2 SQL)
VARGRAPHIC (Db2 SQL)
Casting between data types (Db2 SQL)

APAR PI88790 delivered the functional code to support explicit casting of numeric values to GRAPHIC or VARGRAPHIC.

Db2 12 migration process changes

The Db2 12 migration process is modified to facilitate activation of function level 502 from function level 100, or as part of migration from Db2 11 (with z/OSMF only). Function level 502 requires catalog level 502, and tailoring the catalog for level 502 requires function level 500 or 501. To help manage these requirements, function level 502 introduces the following changes to the Db2 12 migration process:

- Job DSNTIJA1 is added for the intermediate activation of function level 501. DSNTIJA1 consists of a single job step that activates function level 501. It is generated by the installation CLIST only in the following circumstances:
  - In non-z/OSMF ACTIVATE mode, when activating function level 502 or higher from function level 100. In non-z/OSMF MIGRATE mode, DSNTIJA1 is not generated because activating function level 500 or higher is not built into that process. Function level 500 or higher is always activated later, by running the CLIST in ACTIVATE mode.
  - In z/OSMF MIGRATE mode, when migrating a Db2 subsystem or the first member only of a data sharing group to function level 502. In z/OSMF ACTIVATE mode, DSNTIJA1 is not generated because activating function level 500 or higher is built into the z/OSMF migration process.
- In ACTIVATE mode, the CLIST is enhanced to compare the current Db2 function level to the specified target function level, which are specified on panel DSNTIP00, and generates the required job set.
- On panel DSNTIP00:
  - A new field CURRENT FUNCTION LEVEL is added. It is only used in ACTIVATE mode. It is initially set to the input DSNTIDxx FUNCLVL parameter and can be changed. In other modes, this field is always blank and protected.
  - The existing DB2 FUNCTION LEVEL field is renamed to TARGET FUNCTION LEVEL.
- On panel DSNTIPMN, a new step is added to activate function level 501 (DSNTIJA1) before the step for tailoring the Db2 catalog (DSNTIJTC).
- In the z/OSMF workflow DSNTIWMN, a new step is added to activate function level 501 (DSNTIJA1) before the step for tailoring the catalog (DSNTIJTC).

To learn how to make new capabilities available for use in your Db2 environment, see Part 4, “Adopting new capabilities in Db2 12 continuous delivery,” on page 167.

Activation details for function level 502

<table>
<thead>
<tr>
<th>Table 9. Function level 502 activation details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enabling APAR:</td>
</tr>
<tr>
<td>PI95511</td>
</tr>
<tr>
<td>Minimum Db2 catalog level:</td>
</tr>
<tr>
<td>V12R1M502</td>
</tr>
</tbody>
</table>
### Function level 502 activation details (continued)

<table>
<thead>
<tr>
<th>Catalog level V12R1M502 changes:</th>
<th>The KEYLABEL column is added in the following catalog tables:</th>
</tr>
</thead>
<tbody>
<tr>
<td>• SYSIBM.SYSINDEXES</td>
<td></td>
</tr>
<tr>
<td>• SYSIBM.SYSTABLESPACE</td>
<td></td>
</tr>
<tr>
<td>• SYSIBM.SYSSTOGROUP</td>
<td></td>
</tr>
<tr>
<td>• SYSIBM.SYSTABLES</td>
<td></td>
</tr>
</tbody>
</table>

**Restriction:** Before you can run CATMAINT to tailor the Db2 catalog for function level 502 or higher, you must first activate function level 500 or 501. That is, the DISPLAY GROUP command output must indicate HIGHEST ACTIVATED FUNCTION LEVEL (V12R1M500) or higher. This restriction prevents tailoring the Db2 catalog for function levels higher than 500 while fallback to Db2 11 remains possible. However, later activating a lower function level such as function level 100* does not restrict the CATMAINT operation.

For a summary of changes in all Db2 12 catalog levels, see Chapter 17, “Catalog changes in Db2 12,” on page 127.

### Application compatibility control:

Applications must run at application compatibility level V12R1M502 or higher to use the following new capabilities:

- CREATE or ALTER statements that specify the KEYLABEL option. However, applications that already reference encrypted data can continue to run at their current application compatibility levels.
- Explicit casting of numeric data types to fixed or variable length graphic strings, with the GRAPHIC or VARGRAPHIC built-in functions or the CAST specification.

### Function level 502 incompatible changes

Function level 502 introduces no incompatible changes.

However, if your environment contains existing user-defined functions named GRAPHIC or VARGRAPHIC, applications with unqualified references to the existing user-defined functions might start invoking the built-in function instead of the user-defined function in certain situations. For information about avoiding such situations, see Ensuring that Db2 executes the intended user-defined function (Db2 Application programming and SQL).

For all incompatible changes in Db2 12, see Chapter 13, “Incompatible changes in Db2 12,” on page 87.
Function level 501 introduces the LISTAGG built-in function, which produces a list of all values in a group.

Contents

“The LISTAGG built-in function” on page 79
“Function level 501 activation details” on page 79
“Function level 501 incompatible changes” on page 80

The LISTAGG built-in function

The LISTAGG built-in function produces a list of all values in a group. An optional separator argument can delimit items in the result list. For example, specifying a comma as the separator produces a comma-separated list. An optional ordering can also be specified for the items within the group.

Example

Produce an alphabetical list of comma-separated names, grouped by department, from the sample employee table.

```
SELECT WORKDEPT,
  LISTAGG(LASTNAME, ', ') WITHIN GROUP(ORDER BY LASTNAME)
  AS EMPLOYEES
FROM EMP
GROUP BY WORKDEPT;
```

The following result is returned.

<table>
<thead>
<tr>
<th>WORKDEPT</th>
<th>EMPLOYEES</th>
</tr>
</thead>
<tbody>
<tr>
<td>A00</td>
<td>HAAS, HEMMINGER, LUCCHESI, O'CONNELL, ORLANDO</td>
</tr>
<tr>
<td>B01</td>
<td>THOMPSON</td>
</tr>
<tr>
<td>C01</td>
<td>KWAN, NATZ, NICHOLLS, QUINTANA</td>
</tr>
<tr>
<td>D01</td>
<td>ADAMSON, BROWN, JOHN, JONES, LUTZ, PIANKA, SCOUTTEN, STERN, WALKER, YAMAMOTO, YOSHIMURA</td>
</tr>
<tr>
<td>D21</td>
<td>JEFFERSON, JOHNSON, MARINO, MONTEVERDE, PEREZ, PULASKI, SMITH</td>
</tr>
<tr>
<td>E01</td>
<td>GEYER</td>
</tr>
<tr>
<td>E11</td>
<td>HENDERSON, PARKER, SCHNEIDER, SCHWARTZ, SETRIGHT, SMITH, SPRINGER</td>
</tr>
<tr>
<td>E21</td>
<td>ALONZO, GOUNOT, LEE, MEHTA, SPENSER, WONG</td>
</tr>
</tbody>
</table>

APAR PI73929 delivered functional code to support the LISTAGG built-in function.

To learn how to make new capabilities available for use in your Db2 environment, see Part 4, “Adopting new capabilities in Db2 12 continuous delivery,” on page 167.

Function level 501 activation details

| Table 10. Activation details for function level 501. |
|---------------------------------------------|------------------|
| Enabling APAR:                             | PI70535          |
| Minimum required catalog level:            | V12R1M500        |
Function level 501 incompatible changes

Function level 501 introduces no incompatible changes.

For all incompatible changes in Db2 12, see Chapter 13, “Incompatible changes in Db2 12,” on page 87

Related reference

LISTAGG (Db2 SQL)
Chapter 11. Function level 500 (activated at Db2 12 installation or after migration—October, 2016)

Function level function level 500 (V12R1M500) represents the first opportunity for applications to take advantage of most new capabilities in the initial Db2 12 release, including new SQL capabilities and subsystem parameter settings. In many respects, function level 500 is analogous to new-function mode in previous releases.

**Important:** Before you activate function level 500 or higher for the first time, read Migration step 29: Activate function level 500 (Db2 Installation and Migration).

Activation of function level 500 makes most new capabilities and enhancements in the initial Db2 12 available for use, including the following types of enhancements:

**Subsystem parameters**
Most subsystem parameter changes in the initial Db2 12 release take effect in function level 500. For a list of these changes, see Chapter 20, “Subsystem parameter changes in Db2 12,” on page 145.

**SQL capabilities**
New SQL capabilities in the initial Db2 12 release become available in function level 500 for applications that run at the equivalent application compatibility level or higher.

For a summary of new capabilities that are introduced in function level 500, see Part 1, “What's new in the Db2 12 base release,” on page 1.

**Activation details for function level 500**
For information about changes to consider before activating function level 500 and using the equivalent application compatibility level, see Chapter 13, “Incompatible changes in Db2 12,” on page 87.

In Db2 12, function level 500 is activated by default in new installations. However, you can activate a higher function level during the installation process. For migration from Db2 11, you activate function level 500 or a higher function level as part of the migration process.

The catalog level is V12R1M500 for function level 500.

**Important:** Do not issue the ACTIVATE command or run job DSNTIJAF for activation of function level 500 or higher until you are certain that the subsystem or data sharing group can proceed on Db2 12, without the possibility of falling back to or coexistence with Db2 11. In data sharing, the ACTIVATE command has group scope. Fallback and coexistence become impossible with the successful activation of function level 500 or higher.

**Important:** Apply the fallback SPE and stop and restart Db2 11 for every subsystem or data sharing member that you plan to migrate to Db2 12. For data sharing, every member must be started in Db2 11 after the fallback SPE is applied. Inactive members that never started with the fallback SPE applied in Db2 11 cannot start in Db2 12 or Db2 11 after migration to Db2 12 and activation of function level 500 on any other member.

**V12R1M500 application compatibility**
Most new SQL capabilities in function level 500 become available only to applications that use the equivalent application compatibility level or higher. For a list of such SQL capabilities, see Chapter 15, “SQL changes in Db2 12,” on page 113.

For more information about application compatibility levels, see Chapter 32, “Controlling Db2 application compatibility,” on page 187.
Catalog level for function level 500

Db2 12 function levels 100 and 500 require catalog level V12R1M500. Catalog level V12R1M500 is the result when you install a new Db2 12 subsystem, or when you tailor the catalog during migration to Db2 12, as described in Migration step 19: Tailor Db2 12 catalog: DSNTIJTC (Db2 Installation and Migration).

For a summary of changes in all Db2 12 catalog levels, see Chapter 17, “Catalog changes in Db2 12,” on page 127.

Related tasks
Activating Db2 12 function levels
You control the activation and adoption of new features in Db2 12 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.

Migration step 29: Activate function level 500 (Db2 Installation and Migration)

Related reference
-ACTIVATE (Db2) (Db2 Commands)
Chapter 12. Function level 100 (activated by migration to Db2 12)

In function level 100 (V12R1M100), Db2 runs on Db2 12 code, and virtual storage and optimization enhancements in the initial Db2 12 release become available. However, most new function, including and subsystem parameter changes and new SQL capabilities remain disabled. In many respects, function level 100 is analogous to conversion mode in previous Db2 releases.

The following types of Db2 12 enhancements are introduced in function level 100:

**Virtual storage enhancements**

Activation of function level 100 introduces all virtual storage enhancements in the initial Db2 12 release.

**Optimization enhancements**

Activation of function level 100 introduces all optimization enhancements in the initial Db2 12 release. A full prepare is required before any SQL statement can benefit from optimization enhancements. When 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

**Activation details for function level 100**

After migration from Db2 11, Db2 subsystems and data sharing members are at function level 100. The Db2 catalog level is V12R1M500. Subsystems and data sharing members at function level 100 can coexist with Db2 11 new-function mode.

After the activation of function level 500 you can return a subsystem or group member to function level 100*. You can also activate function level 100* in new Db2 12 installations. However, Db2 subsystems and data sharing members at function level 100* cannot coexist with or fall back to Db2 11.

**Catalog level for function level 100**

Db2 12 function levels 100 and 500 require catalog level V12R1M500. Catalog level V12R1M500 is the result when you install a new Db2 12 subsystem, or when you tailor the catalog during migration to Db2 12, as described in Migration step 19: Tailor Db2 12 catalog: DSNTIJTC (Db2 Installation and Migration).

For a summary of changes in all Db2 12 catalog levels, see Chapter 17, “Catalog changes in Db2 12,” on page 127.

**Related tasks**

Migration step 29: Activate function level 500 (Db2 Installation and Migration)

Adopting new capabilities in Db2 12 continuous delivery

In Db2 12, 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 3. What’s changed in Db2 12

Use this information when you are planning migration to Db2 12 and for planning to adopt new capabilities that Db2 12 introduces.

Db2 for z/OS news: Get the latest news about new capabilities and enhancements in Db2 for z/OS, from the IBM experts who design, build, test, and support Db2. For details, see News about Db2 for z/OS from the IBM lab.

This section contains information about incompatible changes that might impact your migration to Db2 12 or the activation of Db2 12 function levels. It also contains summaries of the changes that Db2 12 introduces, including separate sections for changes in the Db2 12 base release and changes in higher function levels. It also summarizes function the Db2 12 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 13. Incompatible changes in Db2 12

Before migrating your Db2 11 subsystem to Db2 12, applying maintenance in Db2 12, 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 12 migration process, or before applying maintenance.

Related concepts
Function that Db2 12 no longer supports
If you are migrating to Db2 12 from Db2 11, be aware of the function that Db2 12 no longer supports.

What’s changed in Db2 12
Use this information when you are planning migration to Db2 12 and for planning to adopt new capabilities that Db2 12 introduces.

Related reference
Deprecated function in Db2 12
Certain capabilities that Db2 12 for z/OS supports are deprecated, meaning that their use is discouraged. Although they remain supported except as noted below in Db2 12, support is likely to be removed eventually.

Incompatible changes summary for function levels 501 and higher

Some Db2 12 function levels introduce incompatible changes to Db2 12 that you must plan to resolve before you activate Db2 12 function levels. Application programming and SQL incompatibilities must be resolved for specific application programs before they run with the corresponding higher application compatibility level.

This section describes incompatible changes that Db2 12 introduces when you activate function levels 501 and higher in Db2 12.

Function level 505 incompatible changes
Function level 505 introduces no incompatible changes.

If your environment contains existing user-defined functions with the same names and signatures as the new built-in functions, applications with unqualified references to the existing user-defined functions might start invoking the built-in function instead of the user-defined function in certain situations. For information about avoiding such situations, see Ensuring that Db2 executes the intended user-defined function (Db2 Application programming and SQL).

Function level 504 incompatible changes
Function level 504 introduces no incompatible changes. However, be aware of changes that might impact your Db2 environment and applications for packages bound at APPLCOMPAT(V12R1M504) or higher, such as the following changes:

• If your Db2 applications have unqualified references to existing user-defined functions, they might start invoking new built-in functions instead if the names and signatures match, in certain situations. For information about avoiding such situations, see Ensuring that Db2 executes the intended user-defined function (Db2 Application programming and SQL).

• If your Db2 environment contains existing columns or variables with names such as CURRENT_TIMEZONE or CURRENT_SERVER, you must modify your applications to delimit these names. For example, you can issue the following queries to identify columns and variables with these names.

For columns, issue:

```
SELECT * FROM SYSIBM.SYSCOLUMNS WHERE NAME IN('CURRENT_SERVER', 'CURRENT_TIMEZONE');
```
For variables, issue:

```sql
SELECT * FROM SYSIBM.SYSVARIABLES
WHERE NAME IN ('CURRENT_SERVER', 'CURRENT_TIMEZONE');
```

Based on the result, you can then check for dependencies, for example by querying the SYSIBM.SYSPACKDEP catalog table, to identify any static applications that depend on the non-delimited forms of these names.

- CREATE TABLESPACE statements that previously created non-UTS segmented or partitioned table spaces, now always create partition-by-growth or partition-by-range table spaces.
- The following SQL statements now return errors:
  - CREATE SYNONYM statements
  - CREATE TABLE statements that specify existing non-UTS table spaces
  - CREATE TABLE and ALTER TABLE statements that specify ORGANIZE BY HASH

### Function level 503 incompatible changes

Function level 503 introduces the following incompatible changes.

**Result change for system-period temporal tables defined with ON DELETE ADD EXTRA ROW**

A query has incompatible behavior at application compatibility level V12R1M503 if all of the following conditions are met:

- It references a system-period temporal table (STT) with FOR SYSTEM_TIME FROM value1 TO value2 (or BETWEEN value1 AND value2) period specification. The STT can be directly referenced or indirectly referenced through views.
- The STT is defined with the ON DELETE ADD EXTRA ROW clause.
- The STT is defined with the DATA CHANGE OPERATION column.
- During migration to use the temporal auditing feature, the altered table contains existing rows.
- The DATA CHANGE OPERATION column is populated with the default NULL value for the existing rows. When the query is executed, the historical table contains records with a NULL value in DATA CHANGE OPERATION column.

**Action required:** Start a trace for IFCID 0376. In the trace output, function code 1215031 or 1202 identifies the affected queries.

**The SET_MAINT_MODE_RECORD_NO_TEMPORALHISTORY stored procedure is not supported when APPLCOMPAT is V12R1M503 or higher**

Users will not be able to invoke the SYSPROC.SET_MAINT_MODE_RECORD_NO_TEMPORALHISTORY stored procedure when running with APPLCOMPAT=V12R1M503 or higher. Existing applications that invoke the SYSPROC.SET_MAINT_MODE_RECORD_NO_TEMPORALHISTORY stored procedure will fail starting with APPLCOMPAT=V12R1M503 or higher.

**Action required:** Start a trace for IFCID 0376. In the trace output, function code 1215032 or 1203 identifies applications that use the SYSPROC.SET_MAINT_MODE_RECORD_NO_TEMPORALHISTORY stored procedure. Modify such applications to set value of the SYSIBMADM.REPLICATION OVERRIDE built-in global variable to ‘Y’ instead.

**Related information:**

SYSIBMADM.REPLICATION OVERRIDE built-in global variable

### Function level 502 incompatible changes

Function level 502 introduces no incompatible changes.

However, if your environment contains existing user-defined functions named GRAPHIC or VARGRAPHIC, applications with unqualified references to the existing user-defined functions might start invoking the
built-in function instead of the user-defined function in certain situations. For information about avoiding such situations, see Ensuring that Db2 executes the intended user-defined function (Db2 Application programming and SQL).

**Function level 501 incompatible changes**
Function level 501 introduces no incompatible changes.

**Related concepts**
What’s changed in Db2 12
Use this information when you are planning migration to Db2 12 and for planning to adopt new capabilities that Db2 12 introduces.

**Related tasks**
Adopting new capabilities in Db2 12 continuous delivery
In Db2 12, function levels and application compatibility levels control the adoption of most new capabilities by Db2 subsystems and Db2 applications.

**Related reference**
What’s new in Db2 12 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.

---

**Application and SQL release incompatibilities**
When you migrate to or apply maintenance in Db2 12, be aware of and plan for application and SQL release incompatibilities that might affect your Db2 environment.

The following incompatible changes apply at any Db2 12 function level, including when you first migrate to Db2 12. For incompatible changes that might impact your Db2 12 environment when you activate function levels 501 and higher, see “Incompatible changes summary for function levels 501 and higher” on page 87.

**SQL capabilities**
New SQL capabilities become available after the activation of the function level that introduces them or higher, for applications that run at the equivalent application compatibility level or higher. New SQL capabilities in the initial Db2 12 release become available in function level 500 for applications that run at the equivalent application compatibility level or higher. You can continue to run SQL statements compatibly with lower function levels, or previous Db2 releases, including Db2 11 and DB2 10. For details, see Application compatibility levels in Db2 (Db2 Application programming and SQL)

Release incompatibilities that were changed or added since the first edition of this Db2 12 publication are indicated by a vertical bar in the left margin. In other areas of this publication, a vertical bar in the margin indicates a change or addition that has occurred since the Db2 11 release of this publication.

**Result change for SQL statement EXPLAIN PACKAGE (APAR PH09796)**
When Db2 processes the SQL statement EXPLAIN PACKAGE, the HINT_USED column in the PLAN_TABLE is populated with EXPLAIN PACKAGE: copy. The copy field in the HINT_USED column will be one of the following values:
- "CURRENT" - the current copy
- "PREVIOUS" - the previous copy
- "ORIGINAL" - the original copy
**Actions to take**

Change the expected output for queries that reference this column.

**Result change for SQL statement EXPLAIN STABILIZED DYNAMIC QUERY (APAR PH09796)**

When Db2 processes the SQL statement EXPLAIN STABILIZED DYNAMIC QUERY, the HINT_USED column in the PLAN_TABLE is populated with EXPLAIN PACKAGE: `copy`. The `copy` field in the HINT_USED column will be one of the following values:

- "CURRENT" - the current copy
- "INVALID" - the invalid copy

**Actions to take**

Change the expected output for queries that reference this column.

**Package accounting block QPAC size increased to reflect the package copy ID (APAR PH05989)**

The package accounting block QPAC size is increased to reflect the package copy ID.

**Actions to take**

Because this adds fields at the end of the trace record, you must change your code to format the new fields.

**SYSCOPY catalog table DSVOLSER column changes (APAR PH03679)**

APAR PH03679 introduces changes to the DSVOLSER column value in the SYSIBM.SYSCOPY catalog table that might impact applications that use this column. Prior to this APAR, the DSVOLSER column was empty for cataloged, sequential, full image copies. Some applications might assume that if the column length of DSVOLSER is zero, the image copy is cataloged. With APAR PH03679, that assumption is no longer correct. For cataloged, sequential, full image copies that are created from a FlashCopy image copy with consistency that also had uncommitted units of work backed out, the DSVOLSER column now contains Db2 checkpoint information.

**Actions to take**

Modify any applications that use the DSVOLSER column in the SYSCOPY catalog table to tolerate the checkpoint information for cataloged, sequential, full image copies. For details, see the description of DSVOLSER in SYSCOPY catalog table.

**Application compatibility levels apply to data definition and data control statements**

After the activation of function level 500 or higher in Db2 12, application compatibility levels also control syntax, semantics, default values, and option validation for most data definition statements and data control statements. Data definition statements (sometimes abbreviated as DDL) include various CREATE and ALTER statements. Data control statements (sometimes abbreviated as DCL) include various GRANT and REVOKE statements. Only application compatibility levels V12R1M505 and higher control the behavior of any data definition or data control statements.

The APPLCOMPAT bind option for a package applies to most static SQL data definition and data control statements. The CURRENT APPLICATION COMPATIBILITY special register applies to most dynamic SQL data definition and data control statements.

For implicit regeneration of an object, the application compatibility level that was in effect for the previous CREATE or ALTER statement for that object is used.

For materialization of pending data definition changes, the application compatibility level of the pending ALTER statement is used.

You can specify the USING APPLICATION COMPATIBILITY clause of certain ALTER statements to regenerate an object with a specific application compatibility level.
Related concepts
Function levels and related levels in Db2 12
Enhancements to Db2 are enabled for use when you activate function levels.
Application compatibility levels in Db2 (Db2 Application programming and SQL)
Related reference
APPLCOMPAT bind option (Db2 Commands)
CURRENT APPLICATION COMPATIBILITY (Db2 SQL)

Automatic rebind of plans and packages created before DB2 10
Migration-related automatic binds occur in Db2 12 because it cannot use runtime structures from a plan or package that was last bound in a release earlier than DB2 10. Plans and packages that were bound in Db2 11 can run in Db2 12, without the risk of migration-related automatic binds. However, plans and packages that are bound in Db2 12 cannot run on Db2 11 members without an automatic bind in Db2 11.

If you specify YES or COEXIST for the ABIND subsystem parameter, Db2 12 automatically rebinds plans and packages that were bound before DB2 10 when Db2 executes the packages. The result of the automatic bind creates a new package and discards the current copy. Db2 does not move the current copy to the previous or original copy because Db2 12 cannot use it. If a regression occurs, REBIND SWITCH PREVIOUS and REBIND SWITCH ORIGINAL are not available.

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 10. SQLCODE -908, SQLSTATE 23510 is returned for packages, and SQLCODE -923, SQLSTATE 57015 is returned for plans until they are rebound in Db2 12.

Actions to take
By preparing for migration to Db2 12 in Db2 11, you can reduce the change and risk for packages that are subject to automatic binds in Db2 12. To do that, you rebind all packages that were last bound before DB2 10 in Db2 11, before you migrate to Db2 12. 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 Avoid automatic binds at migration to Db2 12 (Db2 Installation and Migration).

KEEPDYNAMIC(YES) bind option support for ROLLBACK
In Db2 12, when the APPLCOMPAT value is V12R1M500, the KEEP DYNAMIC(YES) bind option affects both COMMIT and ROLLBACK statements. With KEEP DYNAMIC(YES), the dynamic SQL statements in the package are retained after COMMIT or ROLLBACK, and those statements can run again without another PREPARE.

Prior to Db2 12, the KEEP DYNAMIC(YES) bind option applied only to COMMIT statements. After a ROLLBACK statement, another PREPARE was required so that the dynamic SQL statements could run. This situation is also true in Db2 12 if application compatibility is set to V11R1 or earlier.

In Db2 12, when the APPLCOMPAT value is V12R1M500 or higher, after a ROLLBACK statement is issued, the behavior is different than in prior versions:

- An OPEN statement without a preceding PREPARE statement does not receive an SQLCODE -514.
- An EXECUTE statement without a preceding PREPARE statement does not receive an SQLCODE -518.

An application that was written in Db2 11 and that was bound with KEEP DYNAMIC(YES) was required to prepare dynamic SQL statement again after a ROLLBACK was issued. In Db2 12 when application compatibility is set to V12R1M500 or higher, those extra PREPARE statements are unnecessary.

Actions to take
As you migrate to Db2 12, review packages that use the KEEP DYNAMIC(YES) bind option. You can make dynamic SQL programs that are bound with KEEP DYNAMIC(YES) run more efficiently by removing PREPARE statements that prepare SQL statements again, following execution of ROLLBACK statements. Do not take this action until you are sure that you no longer need to run the programs in Db2 11 or earlier.
After migrating to Db2 12, if you take this action (to remove PREPARE statements after ROLLBACK), programs will not work properly if you subsequently set application compatibility to V11R1 or earlier.

**Alterations to index compression are pending changes for universal table spaces**

When the application compatibility level is V12R1M500 or higher, altering to use index compression for indexes in universal table spaces is a pending change that places the index in advisory REORG-pending (AREOR) status. The LOAD REPLACE and REBUILD INDEX utilities no longer materialize the change. You must use an online REORG to materialize the new value for the COMPRESS attribute in the ALTER INDEX statement.

In releases before Db2 12, any alteration to use index compression placed the index in REBUILD-pending (RBDP) status. You needed to use the REBUILD INDEX utility to rebuild the index, or use the REORG utility to reorganize the table space that corresponds to the index.

**Actions to take**

For indexes in universal table spaces, use an online REORG to materialize the new value for the COMPRESS attribute in the ALTER INDEX statement.

**Data types of output arguments from a stored procedure call in a Java application**

In function level 500 or higher with application compatibility set to V11R1, when a Java application that uses the IBM Data Server Driver for JDBC and SQLJ calls a stored procedure, the data types of stored procedure output arguments match the data types of the parameters in the stored procedure definition.

**Explanation**

Before DB2 10, if a Java client called a Db2 for z/OS stored procedure, the data types of output arguments matched the data types of the corresponding CALL statement arguments. Starting in DB2 10, the data types of the output arguments match the data types of the parameters in the stored procedure definition.

In Db2 12, when application compatibility is set to V10R1, you can set the DDF_COMPATIBILITY subsystem parameter to SP_PARMS_JV to keep the behavior that existed before DB2 10. However, when application compatibility is set to V11R1 or V12R1M100, or to V12R1M500 or higher, SP_PARMS_JV is no longer supported.

In Db2 12 with application compatibility set to V11R1 or V12R1M100, or to V12R1M500 or higher, if the version of the IBM Data Server Driver for JDBC and SQLJ is lower than 3.63 or 4.13, a java.lang.ClassCastException might be thrown when an output argument value is retrieved.

**Actions to take**

Take one of the following actions:

- Upgrade the IBM Data Server Driver for JDBC and SQLJ to version 3.63 or 4.13, or later.
- Modify the data types in CallableStatement.registerOutParameter method calls to match the parameter data types in the stored procedure definitions. You can set application compatibility to V10R1 and run a trace for IFCID 0376 to identify affected applications. Trace records for those applications have a QW0376FN field value of 8.

**Related concepts**

Application compatibility levels in Db2 (Db2 Application programming and SQL)

**SELECT INTO statements with UNION or UNION ALL**

A UNION or UNION ALL is not allowed in the outermost from-clause of a SELECT INTO statement. However, releases before Db2 12 inadvertently tolerate SQL statements that contain this invalid syntax.

The behavior is controlled by the DISALLOW_SEL_INTO_UNION subsystem parameter. In Db2 11, the default setting tolerates the invalid syntax. In Db2 12 the default setting disallows the invalid syntax.
An application that uses the invalid SQL syntax fails at BIND or REBIND with SQLCODE -109.

**Actions to take**

Identify any packages that use UNION or UNION ALL in the from-clause of a SELECT INTO statement and correct them as necessary. You can temporarily specify that Db2 continues to tolerate the invalid syntax NO for the DISALLOW_SEL_INTO_UNION subsystem parameter. However, this subsystem parameter is deprecated and expected to be removed in the future.

You can identify affected packages while DISALLOW_SEL_INTO_UNION is set to NO by binding suspected packages into a dummy collection ID with EXPLAIN(ONLY) and monitoring IFCID 0376 records. Use the following procedure:

1. Issue the following SQL statement to generate a list of BIND commands.

   ```sql
   SELECT 'BIND PACKAGE(DUMMYCOL) COPY(' || COLLID || '.' || NAME || ') ' ||
   CASE WHEN(VERSION <> '')
   THEN 'COPYVER(' || VERSION || ') ' 
   ELSE '' END ||
   'EXPLAIN(ONLY)' 
   FROM SYSIBM.SYSPACKSTMT 
   WHERE STATEMENT LIKE '%SELECT%INTO%UNION%' 
   OR STATEMENT LIKE '%SELECT%UNIONINTO%';
   ```

   The statement generates output similar to the following BIND subcommand:

   ```
   BIND PACKAGE(DUMMYCOL) COPY(DSN_DEFAULT_COLLID_PLAY01.PLAY01) EXPLAIN(ONLY)
   ```

2. Copy the results of the SELECT statement into a bind job. If any BIND subcommands are longer than 72 bytes, formatting is required.
3. Start and collect a trace for IFCID 0376.
4. Run the bind job that you created.
5. Stop the IFCID 0376 trace and analyze the output.

**Related reference**

DISALLOW_SEL_INTO_UNION in macro DSN6SPRM (Db2 Installation and Migration)

**Related information**

-109 (Db2 Codes)

**Change in SQLCODE when the POWER built-in function result is out of range**

After the activation of function level 500 or higher in Db2 12, the SQLCODE that is returned when the result of the POWER® built-in function is out of range is changed in some cases.

Previously, when Db2 executed the POWER built-in function, and the result was a DOUBLE data type that was out of range, Db2 returned SQLCODE -802. In Db2 12 with function level 500 or higher activated, SQLCODE +802 is returned.

For example, the following query returns SQLCODE +802:

```sql
SELECT POWER(DOUBLE(2.0E38), DOUBLE(2.0))
FROM SYSIBM.SYSDUMMY1;
```

Invocations of the POWER function that have DOUBLE arguments and return out-of-range results return SQLCODE +802 instead of SQLCODE -802.

**Actions to take**

In Db2 12, before function level 500 or higher is activated, identify applications with this incompatibility by starting a trace for IFCID 0376, and then running the applications. Review the trace output for incompatible changes with the identifier 1201. Adjust error processing to account for the change in the returned SQLCODE from an error to a warning.
CHAR9 and VARCHAR9 functions for compatibility with pre-DB2 10 string formatting of decimal data

DB2 10 changed the formatting of decimal data by the CHAR and VARCHAR built-in functions and CAST specifications with a CHAR or VARCHAR result type. In Db2 12 you can use alternative functions for compatibility with applications that require decimal to string output in the pre-DB2 10 formats:

- CHAR9 (Db2 SQL)
- VARCHAR9 (Db2 SQL)

Important: For portable applications that might run on platforms other than Db2 for z/OS, do not use the CHAR9 and VARCHAR9 functions. Other Db2 family products do not support these functions.

Actions to take

Review your setting for the BIF _COMPATIBILITY subsystem parameter. If the value is not CURRENT, and you have applications that require decimal to string output in the pre-DB2 10 format, you can rewrite SQL statements to use the CHAR9 and VARCHAR9 functions instead. This approach enables the development of new applications that can accept the current string formatting of decimal data.

To modify your applications to take advantage of the CHAR9, VARCHAR9 functions:

1. Use an IFCID 0376 trace to identify applications that depend on the pre-DB2 10 formats.
2. Rewrite the SQL statements in the identified applications to use the CHAR9 and VARCHAR9 functions instead of the CHAR and VARCHAR functions.
3. Set the BIF _COMPATIBILITY value to CURRENT.

Related reference

BIF COMPATIBILITY field (BIF _COMPATIBILITY subsystem parameter) (Db2 Installation and Migration)

SQL reserved words

Db2 12 introduces several new SQL reserved words, which are listed in Reserved words (Db2 SQL).

In some cases, the use of these reserved words might cause an incompatibility before new function is activated in Db2 12, regardless of the setting of the APPLCOMPAT flag.

Actions to take

Collect IFCID 0376 trace records in Db2 11. Values 4, 5, and 6 for the QW0376FN field indicate instances of reserved words in applications that will cause an incompatibility in Db2 12. Adjust these applications by changing the reserved word to a delimited identifier or by using a word that is not reserved in Db2 12.

Built-in function and existing user-defined functions

For built-in and user-defined functions the combination of the function name and the parameter list form the signature that Db2 uses to identify the function. If the signatures of new or changed built-in functions in Db2 12 match the signatures existing user-defined functions, applications with unqualified references to the existing user-defined functions might start invoking the new or changed built-in functions instead of the user-defined functions. Db2 12 introduces the following built-in function changes:

Important information about existing user-defined functions: When a new application compatibility level introduces a new or changed built-in function that has the same name and signature as an existing user-defined function, unqualified references to the user-defined function might resolve incorrectly.
Applications that have unqualified references to the user-defined function might fail. To avoid this situation, modify applications to explicitly qualify references to user-defined functions with the same name and signature as the new or changed built-in functions.

Db2 12 introduces the following new built-in functions:

- (FL 500) ARRAY_AGG (Db2 SQL) when used for associative array aggregation.
- (FL 504) CUME_DIST "1" on page 95
- (FL 504) CUME_DIST (aggregate) (Db2 SQL) "1" on page 95
- (FL 505) DECRYPT_DATAKEY_INTEGER, DECRYPT_DATAKEY_BIGINT, DECRYPT_DATAKEY_DECIMAL, DECRYPT_DATAKEY_VARCHAR, DECRYPT_DATAKEY_CLOB, DECRYPT_DATAKEY_VARGRAPHIC, DECRYPT_DATAKEY_DBCLOB, and DECRYPT_DATAKEY_BIT
- (FL 505) ENCRYPT_DATAKEY
- (FL 504) FIRST_VALUE "1" on page 95
- (FL 500) GENERATE_UNIQUE_BINARY
- (FL 500) HASH_CRC32, HASH_MD5, HASH_SHA1, and HASH_SHA256
- (FL 504) LAG "1" on page 95
- (FL 504) LAST_VALUE "1" on page 95
- (FL 504) LEAD "1" on page 95
- (FL 501) LISTAGG (Db2 SQL)
- (FL 504) NTH_VALUE "1" on page 95
- (FL 504) NTILE "1" on page 95
- (FL 500) PERCENTILE_CONT
- (FL 500) PERCENTILE_DISC
- (FL 504) PERCENT_RANK "1" on page 95
- (FL 504) PERCENT_RANK (aggregate) (Db2 SQL) "1" on page 95
- (FL 504) RATIO_TO_REPORT "1" on page 95
- (FL 504) REGEXP_COUNT (Db2 SQL) "1" on page 95
- (FL 504) REGEXP_INSTR (Db2 SQL) "1" on page 95
- (FL 504) REGEXP_LIKE (Db2 SQL) "1" on page 95
- (FL 504) REGEXP_REPLACE (Db2 SQL) "1" on page 95
- (FL 504) REGEXP_SUBSTR (Db2 SQL) "1" on page 95
- (FL 500) WRAP (Db2 SQL)

Notes:

1. Supported in Db2 for z/OS only for pass-through to IBM Db2 Analytics Accelerator for z/OS.

Table 11. Changes to built-in functions in Db2 12

<table>
<thead>
<tr>
<th>Built-in function</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>GRAPHIC (Db2 SQL)</td>
<td>(FL 502) The first argument accepts numeric data types, including SMALLINT, INTEGER, BIGINT, DECIMAL, REAL, DOUBLE, FLOAT, and DECFLOAT.</td>
</tr>
<tr>
<td>VARGRAPHIC (Db2 SQL)</td>
<td>(FL 502) The first argument accepts numeric data types, including SMALLINT, INTEGER, BIGINT, DECIMAL, REAL, DOUBLE, FLOAT, and DECFLOAT.</td>
</tr>
</tbody>
</table>
Actions to take

To continue to execute a user-defined function with the same name or signature as a new built-in function or signature, qualify the name of the existing user-defined function in your application. For more information about Db2 resolves qualified and unqualified references to functions, see Function resolution (Db2 SQL).

SQLCODE changes

Some SQLCODE numbers and message text might have changed in Db2 12. Also, the conditions under which some SQLCODEs are issued might have changed. For more information, see New, changed, and deleted codes (Db2 Codes).

V12R1Mnnn application compatibility levels

In Db2 12, you can use the application compatibility level to control the adoption of new SQL capabilities and enhancements of particular function 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 V12R1Mnnn 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 (Db2 Application programming and SQL).

Application compatibility levels are specified in commands and message output by nine-character strings that correspond to the Db2 version, release, and modification value of the corresponding function level. The format is VvvRrMmmm, where vv is the version, r is the release, and mmmm is the modification level. For example, V12R1M505 identifies function level 505. For a list of all available function levels in Db2 12, see Part 2, “What’s new in Db2 12 function levels,” on page 59.

Related concepts
Function levels and related levels in Db2 12
Enhancements to Db2 are enabled for use when you activate function levels.

Related reference
APPLCOMPAT bind option (Db2 Commands)
CURRENT APPLICATION COMPATIBILITY (Db2 SQL)
**V11R1 application compatibility level**

When you set the application compatibility level to V11R1, applications that attempt to use functions and features that are introduced in Db2 12 or later might behave differently or receive an error.

When new function is activated in your Db2 12 environment, you can run individual applications with some of the features and behavior of Db2 11. Your applications can continue to experience V11R1 behavior after new function is activated in Db2 12. Then, you can migrate each application to a new application compatibility value separately until all are migrated. If application compatibility level is set to V11R1 and you attempt to use the new functions of a later version, SQL might behave differently or result in a negative SQLCODE, such as SQLCODE -4743.

You can run package level accounting or monitor traces with IFCID 0239 and review field QPACINCOMPAT, which indicates an SQL incompatible change. If a trace is started for IFCID 0376, and application compatibility is set for a previous version, details about features and functions that have a change in behavior are written in field QW0376FN.

A migrated Db2 12 environment behaves with V11R1 application compatibility until new function is activated. For each Db2 version, application and SQL incompatibilities are described in the migration information for that version.

The following table shows many of the features and functions that are controlled by application compatibility, and the results if you specify V11R1. If a behavior difference is traced, then the IFCID trace function code is shown.

<table>
<thead>
<tr>
<th>Feature or Function</th>
<th>Result with V11R1 application compatibility</th>
<th>IFCID 0376 trace function code</th>
</tr>
</thead>
<tbody>
<tr>
<td>The POWER built-in function returns a result with the DOUBLE data type. The result is out of range.</td>
<td>SQLCODE -802</td>
<td>1201</td>
</tr>
<tr>
<td>CURRENT_SERVER or CURRENT_TIMEZONE is used as a column name or variable name.</td>
<td>SQLCODE -206</td>
<td>1204</td>
</tr>
</tbody>
</table>

**V10R1 application compatibility level**

When you set the application compatibility level to V10R1, applications that attempt to use functions and features that are introduced in Db2 11 or later might behave differently or receive an error.

In Db2 12, you can continue to run individual applications with some of the features and behavior of DB2 10. Your applications can continue to experience V10R1 behavior while in Db2 12, regardless of whether new function is activated. Then, you can migrate each application to a new application compatibility value separately until all are migrated. If application compatibility is set to V10R1 and you attempt to use the new functions of a later version, SQL might behave differently or result in a negative SQLCODE, such as SQLCODE -4743.

You can run package level accounting or monitor traces with IFCID 0239 and review field QPACINCOMPAT, which indicates an SQL incompatible change. If a trace is started for IFCID 0376, and application compatibility is set for a previous version, details about features and functions that have a change in behavior are written in field QW0376FN.
A migrated Db2 12 environment behaves with V11R1 application compatibility until new function is activated. Application and SQL incompatibilities are described in the migration information for each version.

The following table shows many of the features and functions that are controlled by application compatibility, and the results if you specify V10R1. If a behavior difference is traced, then the IFCID trace function code is shown.

<table>
<thead>
<tr>
<th>Feature or Function</th>
<th>Result with V10R1 application compatibility</th>
<th>IFCID 0376 trace function code</th>
</tr>
</thead>
<tbody>
<tr>
<td>An SQL statement in a client application includes an unsupported conversion (from a string type to a numeric type or from a numeric type to a string type), and implicit casting is disabled (DDF_COMPATIBILITY is set to SP_PARMS_NJV or to DISABLE_IMPCAST_NJV).</td>
<td>SQLCODE -301</td>
<td>7 “1” on page 100</td>
</tr>
<tr>
<td>A client application executes an SQL CALL statement to execute a Db2 for z/OS stored procedure. The DDF_COMPATIBILITY subsystem parameter is set to SP_PARMS_NJV for client applications other than Java applications, or SP_PARMS_JV for Java applications.</td>
<td>The data types of the data that is returned from the SQL CALL statement match the data types of the CALL statement arguments. This behavior is compatible with the behavior before Version 10.</td>
<td>8 “1” on page 100</td>
</tr>
<tr>
<td>A client application accesses Db2 11 from an IBM Data Server Driver for JDBC and SQLJ client. The DDF_COMPATIBILITY subsystem parameter is set to IGNORE_TZ for Java applications.</td>
<td>The Db2 server ignores the TIMEZONE portion, appended by the IBM Data Server Driver for JDBC and SQLJ, of the value in the TIMESTAMP WITH TIMEZONE input to a TIMESTAMP target. This behavior is compatible with the behavior before DB2 10.</td>
<td>9</td>
</tr>
<tr>
<td>BIF_COMPATIBILITY is set to V9_TRIM, and input string-expression is EBCDIC mixed data for the RTRIM, LTRIM, or STRIP built-in function.</td>
<td>The DB2 9 version of SYSIBM.LTRIM(string-expression), SYSIBM.RTRIM(string-expression), or SYSIBM.STRIP(string-expression) is executed.</td>
<td>10</td>
</tr>
<tr>
<td>An implicit insert or update of an XML document node</td>
<td>SQLCODE -20345</td>
<td>1101</td>
</tr>
<tr>
<td>A predicate expression with an explicit cast or an operation with an invalid value that does not affect the results of XPath processing</td>
<td>SQLCODE -20345</td>
<td>1102</td>
</tr>
<tr>
<td>How the resource limit facility uses ASUTIME value for nested routines</td>
<td>SQLCODE -905 is issued only when the ASUTIME limit of the top-level calling package is encountered.</td>
<td>1103</td>
</tr>
</tbody>
</table>
### Table 13. Behavior of V10R1 application compatibility (continued)

<table>
<thead>
<tr>
<th>Feature or Function</th>
<th>Result with V10R1 application compatibility</th>
<th>IFCID 0376 trace function code</th>
</tr>
</thead>
<tbody>
<tr>
<td>The lengths of values that are returned from <code>CURRENT_CLIENT_USERID</code>, <code>CURRENT_CLIENT_WRKSTNNNAME</code>, <code>CURRENT_CLIENT_APPLNAME</code>, or <code>CURRENT_CLIENT_ACCTNG</code> special register are longer than the DB2 10 limits.</td>
<td>The special register values are truncated to the DB2 10 maximum lengths and padded with blanks</td>
<td>1104, 1105, 1106, 1107</td>
</tr>
<tr>
<td>A <code>CAST(string as TIMESTAMP)</code> specification with an input string of length of 8 or an input string of length 13</td>
<td>An explicit cast specification from string as TIMESTAMP interprets an 8-byte character string as a Store Clock value and a 13-byte string as a <code>GENERATE_UNIQUE</code> value. CAST result might be incorrect.</td>
<td>1109</td>
</tr>
<tr>
<td>Invocation of the SPACE or VARCHAR built-in function when the result is defined as VARCHAR(32765), VARCHAR(32766), or VARCHAR(32767)</td>
<td>No error</td>
<td>1110, 1111</td>
</tr>
<tr>
<td>Subsystem parameter <code>XML_RESTRICT_EMPTY_TAG</code> is set to YES, and an empty XML element is serialized as <code>&lt;emptyElement&gt;&lt;/emptyElement&gt;</code></td>
<td>No error</td>
<td>1112</td>
</tr>
<tr>
<td>Specification of bind option <code>DBPROTOCOL(DRDACBF)</code></td>
<td>DSNT298I</td>
<td></td>
</tr>
<tr>
<td>A period specification that follows the name of a view in the FROM clause of a query</td>
<td>SQLCODE -4743</td>
<td></td>
</tr>
<tr>
<td>A period clause that follows the name of a target view in an UPDATE or DELETE statement</td>
<td>SQLCODE -4743</td>
<td></td>
</tr>
<tr>
<td>A <code>SET CURRENT TEMPORAL SYSTEM_TIME</code> statement</td>
<td>SQLCODE -4743</td>
<td></td>
</tr>
<tr>
<td>A <code>SET CURRENT TEMPORAL BUSINESS_TIME</code> statement</td>
<td>SQLCODE -4743</td>
<td></td>
</tr>
<tr>
<td>A reference to a global variable</td>
<td>SQLCODE -4743</td>
<td></td>
</tr>
<tr>
<td>Use of array operations and built-in functions such as</td>
<td>SQLCODE -4743</td>
<td></td>
</tr>
<tr>
<td>- Use of the <code>UNNEST</code> collection-derived-table</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Use of the <code>ARRAY_FIRST</code>, <code>ARRAY_LAST</code>, <code>ARRAY_NEXT</code>, <code>ARRAY_PRIOR</code>, <code>ARRAY_AGG</code>, <code>TRIM_ARRAY</code>, <code>CARDINALITY</code>, <code>MAX_CARDINALITY</code> built-in functions</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- A <code>SET</code> assignment-statement of an array element as a target table</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- A <code>CAST</code> specification with a parameter marker as the source and an array as the data type</td>
<td></td>
<td></td>
</tr>
<tr>
<td>An aggregate function that contains the keyword <code>DISTINCT</code> and references a column that is defined with a column mask</td>
<td>SQLCODE -20478</td>
<td></td>
</tr>
</tbody>
</table>
Table 13. Behavior of V10R1 application compatibility (continued)

<table>
<thead>
<tr>
<th>Feature or Function</th>
<th>Result with V10R1 application compatibility</th>
<th>IFCID 0376 trace function code</th>
</tr>
</thead>
<tbody>
<tr>
<td>An SQL statement contains the GROUP BY clause and references a column that is defined with a column mask</td>
<td>SQLCODE -20478</td>
<td></td>
</tr>
<tr>
<td>An SQL statement contains the set operator UNION ALL or UNION DISTINCT and references a column that is defined with a column mask</td>
<td>SQLCODE -20478</td>
<td></td>
</tr>
<tr>
<td>A reference to an alias for a sequence object</td>
<td>SQLCODE -4743</td>
<td></td>
</tr>
<tr>
<td>A reference to an unqualified sequence that is not resolved to a public alias</td>
<td>SQLCODE -204</td>
<td></td>
</tr>
<tr>
<td>A SELECT with a table function reference that includes a typed correlation clause</td>
<td>SQLCODE -4743</td>
<td></td>
</tr>
<tr>
<td>A CALL statement that specifies an autonomous procedure</td>
<td>SQLCODE -4743</td>
<td></td>
</tr>
<tr>
<td>The following datetime assignments:</td>
<td>SQLCODE -180</td>
<td></td>
</tr>
<tr>
<td>• A valid string representation of a timestamp to a date column</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• A valid string representation of a timestamp to a time column</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• A valid string representation of a date to a timestamp column</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Notes:

1. PSPI To find details about the incompatible parameters, examine the contents of fields QW0376SC_Var, QW0376PR_Var, and QW0376INC_Var. See the DSNWMSG file for more information. PSPI

Db2 utility release incompatibilities

When you migrate to or apply maintenance in Db2 12, be aware of the Db2 utility release incompatibilities that might affect your Db2 environment.

The following incompatible changes apply at any Db2 12 function level, including when you first migrate to Db2 12. For incompatible changes that might impact your Db2 12 environment when you activate function levels 501 and higher, see “Incompatible changes summary for function levels 501 and higher” on page 87.

Release incompatibilities that were changed or added since the first edition of this Db2 12 publication are indicated by a vertical bar in the left margin. In other areas of this publication, a vertical bar in the margin indicates a change or addition that has occurred since the Db2 11 release of this publication.

Changes to the mapping table for REORG

In Db2 12, the format of mapping tables has changed to accommodate the 7-byte RIDS that are needed for partition-by-range table spaces with relative page numbering. (Mapping tables are used by the REORG TABLESPACE utility when SHRLEVEL CHANGE is specified.) In the mapping table, the ORID column has changed to CHAR(7) instead of CHAR(5), and the NRID column has changed to CHAR(11) instead of CHAR(9).
After Db2 12 function level 500 or higher is activated, if you use the Db2 11 format for a mapping table, REORG TABLESPACE does not use it. Instead, the utility implicitly creates a mapping table in the Db2 12 format in the same database as the mapping table that you specified.

**Actions to take**

Either create your mapping tables according to the Db2 12 format or allow REORG to implicitly create the mapping tables for you.

**Related concepts**
Before running REORG TABLESPACE (Db2 Utilities)

**Changes to the DSNACCOX stored procedure result set**

In Db2 12, the DSNACCOX result set includes one or more new columns. This expanded result set is returned when the function level is function level 500 or higher. The result set includes the new column RRTPBGSPACEPCT. This column is reserved for future use.

Additionally, the result set always includes two columns that were optional in previous releases. In Db2 12, columns RRIEMPTYLIMIT and RRTHASHOVRFLWRAT are always returned when the function level is function level 500 or higher.

Applications that issue SELECT(*) against the DSNACCOX result set might fail or return unpredictable results.

**Actions to take**

Review any applications that process the results of the DSNACCOX stored procedure. If necessary, update them to handle the new result set.

**Changes to allocation of image copy data sets on tape for RECOVER**

In Db2 12, in RECOVER jobs, you can no longer include DD statements in the JCL to allocate image copy data sets that are on tape. In Db2 12, you must let RECOVER dynamically allocate image copy data sets that are on tape.

RECOVER jobs that contain DD statements for image copy data sets that are on tape fail with error DSNU030I.

**Actions to take**

If a RECOVER job fails with error DSNU030I, find DD statements for image copy data sets that are on tape, and remove the DD statements.

**Changes to the behavior of the RESTORE SYSTEM utility**

In Db2 12, RESTORE SYSTEM fails if either of the following situations occur:

- The most recent system-level backup before the recovery point is not registered in the DFSMSShsm repository.
- An error is returned by DFSMSShsm during the restore of the production volumes.

In previous releases, if RESTORE SYSTEM encountered a problem with the system-level backup, the utility automatically used the previous system-level backup. The new behavior in Db2 12 gives you control over the situation and allows you to determine the best course of action.

RESTORE SYSTEM jobs might fail.

**Actions to take**

Determine how you want to handle these failures. If you want to use an older system-level backup, use the new RESTOREBEFORE option in the RESTORE SYSTEM utility statement.
Changes to the invalidation of cached dynamic SQL statements by utilities that collect statistics

In Db2 12, the RUNSTATS utility and other utilities that collect inline statistics do not invalidate dynamic statements in the dynamic statement cache by default. In previous releases, the collection of statistics for an object always invalidated cached dynamic statements that referred to the objects. In Db2 12, the INVALIDATECACHE option is added to the RUNSTATS utility and to the following utilities that collect inline statistics:

- LOAD
- REBUILD INDEX
- REORG INDEX
- REORG TABLESPACE

The default option INVALIDATECACHE NO means that the utility does not invalidate statements in the dynamic statement cache because statistics are collected. Db2 might continue to use access paths that are based on stale statistics for dynamic SQL statements that remain in the dynamic statement cache.

**Actions to take**

Add the INVALIDATECACHE YES option to any existing utility jobs that collect statistics and must invalidate SQL statements in the dynamic statement cache.

**Related tasks**

Invalidating statements in the dynamic statement cache (Db2 Performance)

**Related reference**

Utility changes in Db2 12

You can use this information to plan for utility changes in Db2 12.

Changes to the behavior of the UNLOAD utility

In Db2 12, the UNLOAD utility includes the REGISTER option, which specifies whether, in a data sharing environment, pages that are read by the UNLOAD utility are registered with the coupling facility. REGISTER NO reduces data sharing overhead. In previous releases, the UNLOAD utility had only REGISTER YES behavior. In Db2 12, the original default for UNLOAD was REGISTER NO, but APAR PI99075 changes the default to REGISTER YES. REGISTER NO causes pages to be read directly from disk, and not from local or group buffer pools.

**Actions to take**

Apply APAR PI99075 to restore Db2 11 default behavior. Then update UNLOAD utility jobs with REGISTER NO, when appropriate to reduce data sharing overhead.

Changes to the authorization check of the UNLOAD utility

After the activation of function level 500 or higher, Db2 checks for the new UNLOAD privilege, instead of the SELECT privilege, for executing the UNLOAD utility and UNLOAD jobs might fail due to authorization errors.

Db2 checks the new UNLOAD privilege for authorization of the UNLOAD utility. You must use a privilege set that includes one of the following to execute the UNLOAD utility:

- Ownership of the tables
- SELECT privilege on the tables (if the AUTH_COMPATIBILITY system parameter is set to the SELECT_FOR_UNLOAD option)
- UNLOAD privilege on the tables (if the AUTH_COMPATIBILITY system parameter is not set to the SELECT_FOR_UNLOAD option)
- DBADM authority for the database. If the object on which the utility operates is in an implicitly created database, DBADM authority on DSNDB04 or the implicitly created database is sufficient.
- DATAACCESS authority
• SYSADM authority
• SYSCtrl authority (catalog tables only)
• SQLADM authority (catalog tables only)
• System DBADM authority (catalog tables only)
• ACCESSCTRL authority (catalog tables only)
• SECADM authority (catalog tables only)

Actions to take

Take one of the following actions:

• In Db2 11 or Db2 12, start a trace for IFCID 404 to identify the users who need the UNLOAD privilege. Then, in Db2 12, grant the UNLOAD privilege to those users.
• As a temporary solution, set the AUTH_COMPATIBILITY subsystem parameter to SELECT_FOR_UNLOAD so that the SELECT privilege continues to be checked when the UNLOAD utility runs.

Db2 command release incompatibilities

When you migrate from Db2 11 to Db2 12, or apply maintenance in Db2 12, be aware of the command release incompatibilities.

Plan for the following changes in Db2 12 that might affect your migration.

The following incompatible changes apply at any Db2 12 function level, including when you first migrate to Db2 12. For incompatible changes that might impact your Db2 12 environment when you activate function levels 501 and higher, see “Incompatible changes summary for function levels 501 and higher” on page 87.

Release incompatibilities that were changed or added since the first edition of this Db2 12 publication are indicated by a vertical bar in the left margin. In other areas of this publication, a vertical bar in the margin indicates a change or addition that has occurred since the Db2 11 release of this publication.

Change to DISPLAY UTILITY output

The output for the DISPLAY UTILITY command now includes message DSNU285I, regardless of whether the SWITCHTIME parameter was specified in the REORG INDEX or REORG TABLESPACE utility control statement. In Db2 11, message DSNU285I was displayed only if SWITCHTIME was specified in the REORG INDEX or REORG TABLESPACE utility control statement.

Applications that parse DISPLAY UTILITY output might need to be updated.

Actions to take

Determine if any of your applications parse output of the DISPLAY UTILITY command and update the applications if needed.

Related information

DSNU100I (Db2 Messages)

Changed REBIND command result for SWITCH option with invalid package copies

If you issue a REBIND PACKAGE or REBIND TRIGGER PACKAGE command with the SWITCH option and you specify an invalid package, the command now fails.
In previous releases, the only productive reason to switch to an invalid package copy was as the first action of a two-step process. The second step was to rebind the packages with the APREUSE option to reuse the access path information from the invalid package as input for access path reuse.

**Actions to take**
If you previously used the two-step process to reuse access paths from invalid copies, modify your process to use a single step. Rebind with the APREUSE option and specify PREVIOUS or ORIGINAL for the APREUSESOURCE bind option to specify whether the previous or original package copy as input for access path reuse.

**Related tasks**
Reverting to saved access paths for static SQL statements (Db2 Performance)

**Related reference**
APREUSESOURCE bind option (Db2 Commands)
APREUSE bind option (Db2 Commands)

**Related information**
DSN730I (Db2 Messages)
-4750 (Db2 Codes)

**Change to START RLIMIT command result**

**Explanation**
Messages DSNT731I and DSNT732I might be issued for START RLIMIT commands if resource limit tables are detected with deprecated or unsupported formats.

DSNRLSTxx table formats and related index formats earlier than the DB2 Version 8 format are not supported in Db2 12 or later releases. When tables with unsupported formats are detected, Db2 issues message DSNT731I. The START RLIMIT command fails with message DSN9023I if no DSNRLMTxx table exists with the specified ID.

DSNRLMTxx table formats and related index formats earlier than the Db2 11 format are deprecated in Db2 12. If tables with deprecated formats are detected, Db2 issues message DSNT732I, processing for the START RLIMIT command continues, and the resource limit facility starts using the deprecated objects.

Dynamic SQL applications that are specified in the obsolete resource limit tables can run without the limits.

**Actions to take**
Update existing resource limit tables to use the supported formats. The DSNTIJSG installation job contains statements that create the database, table space, tables, and indexes for the resource limit facility. You can tailor those statements to create or update the format of resource limit tables.

**Related tasks**
Run premigration queries (DSNTIJPM) (Db2 Installation and Migration)

**Related reference**
-START RLIMIT (Db2) (Db2 Commands)
Resource limit facility tables (Db2 Performance)
Job DSNTIJSG (Db2 Installation and Migration)
Security release incompatibilities

When you migrate to or apply maintenance in Db2 12, be aware of security release incompatibilities that might affect your Db2 environment.

The following incompatible changes apply at any Db2 12 function level, including when you first migrate to Db2 12. For incompatible changes that might impact your Db2 12 environment when you activate function levels 501 and higher, see “Incompatible changes summary for function levels 501 and higher” on page 87.

Release incompatibilities that were changed or added since the first edition of this Db2 12 publication are indicated by a vertical bar in the left margin. In other areas of this publication, a vertical bar in the margin indicates a change or addition that has occurred since the Db2 11 release of this publication.

Change to RACF long object resource names

Explanation
For UPDATE and REFERENCES operations on tables, table-qualifiers and table-names are each truncated at 100 characters. The column-name is truncated at 30 characters.

Possible impact to your Db2 environment
When you use long object names for UPDATE and REFERENCES operations, along with discrete RACF profiles, truncation can cause unintended results. If truncation occurs, a single discrete profile might inadvertently protect multiple resources if the resources are similarly named, the first 100 characters of the schema names are identical, and the qualified object names, such as the table name, subsystem name, or privilege name, are also identical.

Actions to take
Review the RACF profiles for long table names for UPDATE and REFERENCES operations, and update those profiles, as needed.

Change to authorization to run the UNLOAD utility

Explanation
With activation of function level 500 or higher, the SELECT privilege is no longer sufficient to run the UNLOAD utility. The UNLOAD privilege must be used instead.

Possible impact to your Db2 environment
When an UNLOAD job is run by a user with only the SELECT privilege on the target table, the UNLOAD job fails with message DSNU1253I.

Actions to take
Before migration to Db2 12, run job DSNTIJPM, and follow the steps in premigration report 19 to determine which authorization IDs might need the UNLOAD privilege.

Related tasks
Run premigration queries (DSNTIJPM) (Db2 Installation and Migration)
Storage release incompatibilities

When you migrate to or apply maintenance in Db2 12, be aware of and plan for storage release incompatibilities that might affect your Db2 environment.

The following incompatible changes apply at any Db2 12 function level, including when you first migrate to Db2 12. For incompatible changes that might impact your Db2 12 environment when you activate function levels 501 and higher, see “Incompatible changes summary for function levels 501 and higher” on page 87.

Release incompatibilities that were changed or added since the first edition of this Db2 12 publication are indicated by a vertical bar in the left margin. In other areas of this publication, a vertical bar in the margin indicates a change or addition that has occurred since the Db2 11 release of this publication.

Changed behavior for PGSTEAL(NONE) buffer pools
PGSTEAL(NONE) buffer pools are managed differently in Db2 12.

Explanation
In Db2 12, Db2 keeps all pages for open objects that are assigned to PGSTEAL(NONE) buffer pools buffer pool in memory. When the buffer pool is allocated, Db2 creates an overflow area within the VPSIZE of the buffer pool. The overflow area is used if the buffer pool is not large enough to contain all pages for all open assigned objects.

In previous releases, Db2 used the FIFO algorithm to manage pages when assigned objects were too large for PGSTEAL(NONE) buffer pools.

Possible impact to your Db2 environment
If a PGSTEAL(NONE) buffer pool is not large enough to contain all open assigned objects plus the overflow area, the overflow area is used. Page-stealing is possible in the overflow area. Pages that are allocated to the overflow area are not reallocated in the main part of the buffer pool until the data set is closed.

Actions to take
Verify that existing buffer pools that are defined with the PGSTEAL(NONE) option meet the following criteria:

- Assigned objects are frequently accessed, and not likely to grow quickly. For example, you can use the following real-time statistics values to identify candidates:
  - GETPAGES
  - NACTIVE
  - REORGINSERTS
- The VPSIZE of the buffer pool is large enough to contain all pages for all assigned objects, and the overflow area. The size of the overflow is approximately 10 percent of the VPSIZE value for the buffer pool. The minimum size of the overflow area is 50 buffers, and the maximum size is 6400 buffers.

Alter buffer pools that do not meet the criteria to use the PGSTEAL(FIFO) or PGSTEAL(LRU) option.

Related tasks
Choosing a page-stealing algorithm (Db2 Performance)

Related reference
-ALTER BUFFERPOOL (Db2) (Db2 Commands)

Related information
DSNB416I (Db2 Messages)
Change to storage handling for the dynamic statement cache

Db2 12 changes the way in which storage in the EDM pool is allocated for cached dynamic statements.

Explanation

In Db2 12, EDMSTMTC specifies an upper limit on the amount of storage in the EDM pool that is used for cached dynamic statements. EDMSTMTC does not specify the initial amount of storage that is allocated. Storage is allocated only as it is needed. If adding of statements to the dynamic statement cache raises the amount of storage to a value that is greater than EDMSTMTC, the Db2 database manager removes statements that are not in use from the cache to lower the amount of storage to the EDMSTMTC value.

In previous releases, EDMSTMTC specified the initial amount of storage in the EDM pool that was allocated for cached dynamic statements. The initial EDMSTMTC value also established the minimum amount of storage that could be used for cached dynamic statements until the Db2 subsystem was restarted.

Possible impact to your Db2 environment

More real storage might be needed.

Actions to take

If storage shortages occur, increase the amount of real storage that is available.

Related tasks

Calculating the EDM statement cache hit ratio (Db2 Performance)

Related reference

EDM STATEMENT CACHE field (EDMSTMTC subsystem parameter) (Db2 Installation and Migration)

Increased EDM storage usage for DBDs

Starting with Db2 12 function level V12R1M100, you might need more EDM pool storage for DBDs than you needed in Db2 11.

Explanation

Immediately after migration to Db2 12 function level V12R1M100, for tables and table spaces that were created prior to Db2 12, more space is needed for storing their internal data structures in the EDM pool. However, after new function activation (Db2 12 function level V12R1M500 or later), when a data definition statement such as an ALTER is performed on one of those tables or table spaces, the storage usage for their internal structures is reduced again.

Possible impact to your Db2 environment

More EDM pool storage for DBDs might be needed after migration to Db2 12 function level V12R1M100, and before a data definition change to a table or table space in Db2 12 function level V12R1M500 or later.

Actions to take

Issue the following query after you migrate to Db2 12 function level V12R1M100. Then update the EDMDBDC subsystem parameter with a value that is at least twice the query result.

```sql
WITH DBD_LEN_TABLE (DBD_LEN) AS
  (SELECT SUM(LENGTH(A.DBD_DATA)) AS DBD_LEN
   FROM SYSIBM.DBDR A, SYSIBM.SYSDATABASE B
   WHERE A.DBID = B.DBID
   GROUP BY A.DBID
   ORDER BY DBD_LEN DESC)
SELECT SUM(DBD_LEN)/1024 AS EDMDBDC_IN_KBYTES
FROM DBD_LEN_TABLE
WITH UR;
```

Related reference

EDM DBD CACHE field (EDMDBDC subsystem parameter) (Db2 Installation and Migration)
**BSDS record format conversion is required**

The BSDS must be converted to use the 10-byte format RBA and LRSN records before Db2 12 can start.

At migration to Db2 12, you cannot start Db2 12 until the BSDS is converted to use the 10-byte RBA and LRSN formats. You can convert the BSDS before or during the Db2 12 migration process.

Before you can convert the BSDS, the Db2 subsystem or data sharing member must be started in Db2 11 new-function mode.

**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.

In a data sharing environment, page sets contain LRSNs instead of RBAs. After the BSDS has been converted to allow for 10-byte LRSNs, an LRSN that is greater than 6 bytes does not result in failed updates with reason code 00C2026D. It is recommended that after the BSDS has been converted, the catalog and directory table spaces are converted before any other objects. Other objects need to be converted to extended format before the LRSN value reaches the 6-byte LRSN limit.

For instructions, see Convert BSDS records to the extended 10-byte format: DSNTIJCB (Db2 Installation and Migration).

**Related concepts**

The extended 10-byte RBA and LRSN in Db2 12
Beginning in Db2 11, Db2 uses 10-byte RBA and LRSN values.

**Related tasks**

Migration step 18: Start Db2 12 (Db2 Installation and Migration)

**Related reference**

DSNJCNVT (Db2 Utilities)
Chapter 14. Command changes in Db2 12

Db2 12 introduces new commands and changes to existing commands.

Important: Db2 12 introduces continuous delivery of new capabilities and enhancements in function levels. Most Db2 12 enhancements and changes become available only after the activation of the function level that delivers them. For more information, see Chapter 29, “Activating Db2 12 function levels,” on page 179. Each new or changed item is labeled below with a link to information about the Db2 12 function level that delivered it.

New commands

Db2 12 introduces the following new commands:

- (FL 500) -DISPLAY ML (Db2) (Db2 Commands) “1” on page 109
- (FL 500) -START ML (Db2) (Db2 Commands) “1” on page 109
- (FL 500) -STOP ML (Db2) (Db2 Commands)
- (FL 500) -DISPLAY DYNQUERYCAPTURE (Db2) (Db2 Commands)
- (FL 500) -DISPLAY STATS (Db2) (Db2 Commands)
- (FL 500) FREE STABILIZED DYNAMIC QUERY (DSN) (Db2 Commands)
- (FL 500) -START CDDS (Db2) (Db2 Commands)
- (FL 500) -STOP CDDS (Db2) (Db2 Commands)
- (FL 500) -START DYNQUERYCAPTURE (Db2) (Db2 Commands)
- (FL 500) -STOP DYNQUERYCAPTURE (Db2) (Db2 Commands)
- (FL 100) -ACTIVATE (Db2) (Db2 Commands)
- (FL 100) BIND SERVICE (DSN) (Db2 Commands)
- (FL 100) -DISPLAY RESTSVC (Db2) (Db2 Commands)
- (FL 100) FREE SERVICE (DSN) (Db2 Commands)
- (FL 100) -START RESTSVC (Db2) (Db2 Commands)
- (FL 100) -STOP RESTSVC (Db2) (Db2 Commands)

Notes:

1. With IBM Db2 AI for z/OS Version 1.1.01 or higher and APAR PH05323.

Changes to existing commands

The following table shows the existing commands that have new and changed options. For information about an option with a changed default value, see Command release incompatibilities.
<table>
<thead>
<tr>
<th>Command</th>
<th>Description of enhancements and notes</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>-ALTER BUFFERPOOL (Db2) (Db2 Commands)</strong></td>
<td>Changed option: (FL 500) The meaning of the existing PGSTEAL(NONE) option is changed. As in previous versions, Db2 pre-loads the data for assigned objects when they are opened and keeps the data resident until the objects are closed. However, in Db2 12, Db2 creates an overflow area within the VPSIZE of the buffer pool. If all open assigned objects do not fit into the main part of the buffer pool, pages are allocated to the overflow area. Page stealing is possible in the overflow area. After pages are allocated in the overflow area, you must close the object to move the pages back to the overflow area.</td>
</tr>
<tr>
<td><strong>-ALTER GROUPBUFFERPOOL (Db2) (Db2 Commands)</strong></td>
<td>The maximum value that can be specified for the RATIO keyword is increased to 1024.</td>
</tr>
<tr>
<td><strong>BIND PACKAGE (DSN) (Db2 Commands)</strong></td>
<td><strong>New options:</strong></td>
</tr>
<tr>
<td></td>
<td>- (FL 500) CONCENTRATESTMT</td>
</tr>
<tr>
<td><strong>BIND QUERY (DSN) (Db2 Commands)</strong></td>
<td>Support is added for processing the following SQL statement types:</td>
</tr>
<tr>
<td></td>
<td>- (FL 500) INSERT</td>
</tr>
<tr>
<td></td>
<td>- (FL 500) UPDATE WHERE CURRENT OF</td>
</tr>
<tr>
<td></td>
<td>- (FL 500) DELETE WHERE CURRENT OF</td>
</tr>
<tr>
<td><strong>BIND SERVICE (DSN) (Db2 Commands)</strong></td>
<td><strong>New options:</strong></td>
</tr>
<tr>
<td></td>
<td>- COPY</td>
</tr>
<tr>
<td></td>
<td>- COPYVER</td>
</tr>
<tr>
<td></td>
<td>- OPTIONS</td>
</tr>
<tr>
<td><strong>-DISPLAY ARCHIVE (Db2) (Db2 Commands)</strong></td>
<td>(FL 502) The DISPLAY ARCHIVE command displays the current key label information for active logs and archive logs.</td>
</tr>
<tr>
<td><strong>-DISPLAY LOG (Db2) (Db2 Commands)</strong></td>
<td>(FL 502) The DISPLAY LOG command displays the current key label information for active logs and archive logs.</td>
</tr>
<tr>
<td><strong>-DISPLAY GROUP (Db2) (Db2 Commands)</strong></td>
<td>(FL 502) The DISPLAY GROUP command provides the current ENCRYPTION_KEYLABEL system parameter value.</td>
</tr>
<tr>
<td></td>
<td>(FL 100) The DSN71001 message output is modified to indicate information about the code, catalog, and function levels of the Db2 subsystem or data sharing group.</td>
</tr>
<tr>
<td>Command</td>
<td>Description of enhancements and notes</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>FREE PACKAGE (DSN) (Db2 Commands)</strong></td>
<td><strong>New values for existing options:</strong></td>
</tr>
<tr>
<td></td>
<td>• (FL 500) PLANMGMTSCOPE(PREVIOUS</td>
</tr>
<tr>
<td></td>
<td><strong>New options:</strong></td>
</tr>
<tr>
<td></td>
<td>• (FL 500) INVALIDONLY</td>
</tr>
<tr>
<td></td>
<td>If you specify INACTIVE, PREVIOUS, ORIGINAL for the PLANMGMTSCOPE option, FREE PACKAGE processing can now complete while application that use the package are running.</td>
</tr>
<tr>
<td><strong>-MODIFY DDF (Db2) (Db2 Commands)</strong></td>
<td>After the PTF for APAR PH08188 has been applied, the values for PORT and SECPORT can be the same.</td>
</tr>
<tr>
<td></td>
<td><strong>New options:</strong></td>
</tr>
<tr>
<td></td>
<td>(FL 500) SESSIDLE</td>
</tr>
<tr>
<td><strong>REBIND PACKAGE (DSN) (Db2 Commands)</strong></td>
<td>This command can be used to rebind a package for an advanced trigger.</td>
</tr>
<tr>
<td></td>
<td><strong>New options:</strong></td>
</tr>
<tr>
<td></td>
<td>• (FL 500) APREUSESOURCE</td>
</tr>
<tr>
<td></td>
<td>• (FL 500) CONCENTRATESTMT</td>
</tr>
<tr>
<td><strong>REBIND TRIGGER PACKAGE (DSN) (Db2 Commands)</strong></td>
<td>(FL 500) This command can be used to rebind a package for a basic trigger, including any trigger that was created prior to the activation of function level 500 or higher.</td>
</tr>
<tr>
<td></td>
<td><strong>New options:</strong></td>
</tr>
<tr>
<td></td>
<td>• (FL 500) APREUSESOURCE</td>
</tr>
<tr>
<td><strong>-START DB2 (Db2) (Db2 Commands)</strong></td>
<td>(FL 500) You cannot start Db2 at any code level that is lower than the catalog level or the highest activated function level.</td>
</tr>
<tr>
<td><strong>-START PROFILE (Db2) (Db2 Commands)</strong></td>
<td>(FL 500) If the PROFILE_AUTOSTART subsystem parameter is set to YES, this command is issued automatically when Db2 starts.</td>
</tr>
<tr>
<td><strong>-START RLIMIT (Db2) (Db2 Commands)</strong></td>
<td>(FL 500) Certain previously supported resource limit table formats are deprecated or unsupported in Db2 12. Resource limits are not activated for tables with unsupported formats.</td>
</tr>
<tr>
<td></td>
<td>(FL 100) Resource limits that apply to existing threads can be modified by issuing the START RLIMIT command.</td>
</tr>
<tr>
<td></td>
<td><strong>New options:</strong></td>
</tr>
<tr>
<td></td>
<td>• (FL 100) SCOPE</td>
</tr>
<tr>
<td><strong>-TERM UTILITY (Db2) (Db2 Commands)</strong></td>
<td>(FL 500) The TERM UTILITY command can be used for the MODIFY RECOVERY utility.</td>
</tr>
</tbody>
</table>
You can use this information to plan for SQL changes in Db2 12.

**Important:** Db2 12 introduces continuous delivery of new capabilities and enhancements in function levels. Most Db2 12 enhancements and changes become available only after the activation of the function level that delivers them. For more information, see Chapter 29, “Activating Db2 12 function levels,” on page 179. Each new or changed item is labeled below with a link to information about the Db2 12 function level that delivered it.

**Important:** Applications must run with V12R1M500 application compatibility or higher to use most of these new SQL capabilities. For more information, see Application compatibility levels in Db2 (Db2 Application programming and SQL).

### New SQL statements
Db2 12 introduces the following new SQL statements:

- (FL 500) ALTER TRIGGER (advanced) (Db2 SQL)
- (FL 500) CREATE TRIGGER (advanced) (Db2 SQL)
- (FL 500) TRANSFER OWNERSHIP (Db2 SQL)

### SQL statement changes
The following table shows the changes to existing SQL statements. For information about a clause with a changed default value, see Application and SQL release incompatibilities.

<table>
<thead>
<tr>
<th>SQL statement</th>
<th>Description of enhancements and notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALTER FUNCTION (compiled SQL scalar)</td>
<td>New clauses:</td>
</tr>
<tr>
<td></td>
<td>(FL 500) CONCENTRATE STATEMENTS</td>
</tr>
<tr>
<td>ALTER INDEX</td>
<td>New clauses:</td>
</tr>
<tr>
<td></td>
<td>(FL 500) DSSIZE</td>
</tr>
<tr>
<td></td>
<td>Changed clauses:</td>
</tr>
<tr>
<td></td>
<td>(FL 505) ADD COLUMN column-name (can now specify DECFLOAT columns)</td>
</tr>
<tr>
<td></td>
<td>(FL 500) COMPRESS</td>
</tr>
<tr>
<td>ALTER PROCEDURE (SQL native)</td>
<td>New clauses:</td>
</tr>
<tr>
<td></td>
<td>(FL 500) CONCENTRATE STATEMENTS</td>
</tr>
<tr>
<td>ALTER STOGROUP (Db2 SQL)</td>
<td>(FL 502) KEYLABEL</td>
</tr>
</tbody>
</table>
Table 15. Changes to existing SQL statements (continued)

<table>
<thead>
<tr>
<th>SQL statement</th>
<th>Description of enhancements and notes</th>
</tr>
</thead>
</table>
| ALTER TABLE            | **New clauses:** (FL 502) KEYLABEL  
|                        | (FL 500) CCSSID  
|                        | (FL 500) EXCLUSIVE and INCLUSIVE clauses of PERIOD BUSINESS_TIME clause  
|                        | (FL 500) PERIOD BUSINESS_TIME clause for constraints  
|                        | **Changed clauses:** (FL 505) PRIMARY KEY (now supports DECFLOAT columns)  
|                        | (FL 505) UNIQUE (now supports DECFLOAT columns)  
|                        | (FL 503) ON DELETE ADD EXTRA ROW  
|                        | (FL 500) ADD PERIOD FOR  
|                        | (FL 500) ADD PARTITION  |
| ALTER TABLESPACE       | **New clauses:** (FL 500) PAGENUM  
|                        | (FL 500) INSERT ALGORITHM  
|                        | **Changed clauses:** (FL 500) COMPRESS  
|                        | (FL 500) DSSIZE  |
| ALTER TRIGGER (basic)  | Equivalent to ALTER TRIGGER in prior releases.                                                          |
| COMMENT (Db2 SQL)      | **Changed clauses:** (FL 500) TRIGGER trigger-name VERSION trigger-version-id |
| CREATE FUNCTION        | **New clauses:** (FL 500) CONCENTRATE STATEMENTS  
| (compiled SQL scalar)  | (FL 500) WRAPPED  |
| CREATE FUNCTION        | **New clauses:** (FL 500) WRAPPED  
| (inline SQL scalar)    |  
| CREATE FUNCTION        | **New clauses:** (FL 500) WRAPPED  
| (SQL table)            |  
| CREATE INDEX           | **New clauses:** (FL 500) DSSIZE  
|                        | **Changed clauses:** (FL 505) column-name (can now specify DECFLOAT columns)  
| CREATE PROCEDURE (SQL  | **New clauses:** (FL 500) CONCENTRATE STATEMENTS  
<p>| native)                | (FL 500) WRAPPED  |
| CREATE STOGROUP        | <strong>New clauses:</strong> (FL 502) KEYLABEL  |</p>
<table>
<thead>
<tr>
<th>SQL statement</th>
<th>Description of enhancements and notes</th>
</tr>
</thead>
</table>
| CREATE TABLE          | **New clauses:**  
                          (FL 502) KEYLABEL  
                          (FL 500) CCSID on a CHAR, GRAPHIC, CLOB, or DBCLOB column  
                          (FL 500) EXCLUSIVE and INCLUSIVE clauses of the PERIOD BUSINESS_TIME clause  
                          (FL 500) PAGENUM  
                          (FL 500) PERIOD BUSINESS_TIME for constraints  
                          **Changed clauses:**  
                          (FL 505) PRIMARY KEY (now supports DECFLOAT columns)  
                          (FL 505) UNIQUE (now supports DECFLOAT columns)  
                          (FL 500) DSIZE  
                          (FL 500) PERIOD FOR                                                                                                                                                      |
| CREATE TABLESPACE     | **New clauses:**  
                          (FL 500) PAGENUM  
                          (FL 500) INSERT ALGORITHM  
                          **Changed clauses:**  
                          (FL 500) COMPRESS  
                          (FL 500) DSIZE  
                          (FL 500) NUMPARTS                                                                                                                                                        |
| CREATE TRIGGER (basic)| **New clauses:**  
                          (FL 500) WRAPPED  
                          **Changed clauses:**  
                          (FL 505) WHEN search-condition                                                                                                                                           |
| CREATE TRIGGER (advanced) | **Changed clauses:**  
                          (FL 505) WHEN search-condition                                                                                                                                               |
| CREATE VARIABLE       | **Changed clauses:**  
                          (FL 500) data-type                                                                                                                                                          |
| DELETE                | **New clauses:**  
                          (FL 500) BETWEEN value-1 AND value-2 clause of the period-clause  
                          (FL 500) FETCH FIRST n ROWS ONLY                                                                                                                                           |
| EXECUTE               | **Changed clauses:**  
                          (FL 500) USING                                                                                                                                                              |
| EXECUTE IMMEDIATE (Db2 SQL) | **Changed clauses:**  
                          (FL 500) variable                                                                                                                                                           |
| EXPLAIN               | **New clauses:**  
                          (FL 500) STABILIZED DYNAMIC QUERY STMTID                                                                                                                                      |
| FETCH                 | **Changed clauses:**  
                          (FL 500) target-variable                                                                                                                                                      |
<table>
<thead>
<tr>
<th>SQL statement</th>
<th>Description of enhancements and notes</th>
</tr>
</thead>
</table>
| fullselect (Db2 SQL)                  | **New clauses:**  
(FL 500) offset-clause (Db2 SQL)  
**Changed clauses:**  
(FL 500) fetch-clause (Db2 SQL)  
(FL 500) order-by-clause (Db2 SQL) |
| GRANT (table or view privileges) (Db2 SQL) | **New clauses:**  
(FL 500) UNLOAD                                                                                                                                                                                                                   |
| MERGE                                | (FL 500) The MERGE statement now includes the delete operation as part of the merge process.  
**New clauses:**  
(FL 500) signal-statement  
(FL 500) ELSE IGNORE  
**Changed clauses:**  
(FL 500) AS correlation-name  
(FL 500) assignment-clause  
(FL 500) WHEN matching-condition  
(FL 500) THEN modification-operation |
| OPEN                                 | **Changed clauses:**  
(FL 500) USING                                                                                                                                                                                                                   |
| PREPARE (Db2 SQL)                    | **New clauses:**  
(FL 500) offset-clause (Db2 SQL)  
**Changed clauses:**  
(FL 500) fetch-clause (Db2 SQL) |
| REVOKE (table or view privileges) (Db2 SQL) | **New clauses:**  
(FL 100) UNLOAD “1” on page 117                                                                                                                                          |
| SELECT INTO (Db2 SQL)                | **New clauses:**  
(FL 500) offset-clause (Db2 SQL)  
**Changed clauses:**  
(FL 500) INTO  
(FL 500) fetch-clause (Db2 SQL) |
| SET assignment-statement (Db2 SQL)   | **Changed clauses:**  
(FL 500) DEFAULT                                                                                                                                                                                                                 |
| subselect (Db2 SQL)                  | **New clauses:**  
(FL 500) offset-clause (Db2 SQL)  
**Changed clauses:**  
(FL 500) correlation-clause of collection-derived-table (Db2 SQL) in a table-reference of a FROM clause  
(FL 500) fetch-clause (Db2 SQL)  
(FL 100) A user-defined function that is defined with MODIFIES SQL DATA can be invoked. |
Table 15. Changes to existing SQL statements (continued)

<table>
<thead>
<tr>
<th>SQL statement</th>
<th>Description of enhancements and notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>UPDATE (Db2 SQL)</td>
<td>New clauses:</td>
</tr>
<tr>
<td></td>
<td>(FL 500) BETWEEN value-1 AND value-2 clause of the period-clause</td>
</tr>
<tr>
<td>VALUES INTO</td>
<td>Changed clauses:</td>
</tr>
<tr>
<td></td>
<td>(FL 500) Assignment clause source</td>
</tr>
<tr>
<td></td>
<td>(FL 500) INTO</td>
</tr>
<tr>
<td>compound-statement (Db2 SQL) for SQL routines</td>
<td>New clauses:</td>
</tr>
<tr>
<td></td>
<td>(FL 500) ATOMIC</td>
</tr>
<tr>
<td></td>
<td>Changed clauses:</td>
</tr>
<tr>
<td></td>
<td>(FL 500) DEFAULT or CONSTANT</td>
</tr>
</tbody>
</table>

Notes:

1. This SQL statement or clause is supported in application compatibility V12R1M100.

General SQL changes

Db2 12 introduces the following general SQL changes:

- Predicates involving row-value-expressions can use the following additional comparison operators: <, >, <=, and >=. See Basic predicate (Db2 SQL) for details.

New and changed built-in functions

Important information about existing user-defined functions: When a new application compatibility level introduces a new or changed built-in function that has the same name and signature as an existing user-defined function, unqualified references to the user-defined function might resolve incorrectly. Applications that have unqualified references to the user-defined function might fail. To avoid this situation, modify applications to explicitly qualify references to user-defined functions with the same name and signature as the new or changed built-in functions.

Db2 12 introduces the following new built-in functions:

- (FL 500) ARRAY_AGG (Db2 SQL) when used for associative array aggregation.
- (FL 504) CUME_DIST”1” on page 118
- (FL 504) CUME_DIST (aggregate) (Db2 SQL)”1” on page 118
- (FL 505) DECRYPT_DATAKEY_INTEGER, DECRYPT_DATAKEY_BIGINT, DECRYPT_DATAKEY_DECIMAL, DECRYPT_DATAKEY_VARCHAR, DECRYPT_DATAKEY_CLOB, DECRYPT_DATAKEY_VARGRAPHIC, DECRYPT_DATAKEY_DBCLOB, and DECRYPT_DATAKEY_BIT
- (FL 505) ENCRYPT_DATAKEY
- (FL 504) FIRST_VALUE”1” on page 118
- (FL 500) GENERATE_UNIQUE_BINARY
- (FL 500) HASH_CRC32, HASH_MD5, HASH_SHA1, and HASH_SHA256
- (FL 504) LAG”1” on page 118
- (FL 504) LAST_VALUE”1” on page 118
Notes:

1. Supported in Db2 for z/OS only for pass-through to IBM Db2 Analytics Accelerator for z/OS.

Table 16. Changes to built-in functions in Db2 12

<table>
<thead>
<tr>
<th>Built-in function</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>GRAPHIC (Db2 SQL)</td>
<td>(FL 502) The first argument accepts numeric data types, including SMALLINT, INTEGER, BIGINT, DECIMAL, REAL, DOUBLE, FLOAT, and DECFLOAT.</td>
</tr>
<tr>
<td>VARGRAPHIC (Db2 SQL)</td>
<td>(FL 502) The first argument accepts numeric data types, including SMALLINT, INTEGER, BIGINT, DECIMAL, REAL, DOUBLE, FLOAT, and DECFLOAT.</td>
</tr>
</tbody>
</table>

New and changed Db2-supplied stored procedures

- Db2 12 introduces or changes the following Db2-supplied stored procedures:
  - (FL 500) CREATE_WRAPPED stored procedure (Db2 SQL) is new
  - (FL 500) DSNUTILV stored procedure (Db2 SQL) is new
  - (FL 500) DSNACCOX stored procedure (Db2 SQL): Several of the default values have changed

Built-in global variables

- Db2 12 introduces the following built-in global variables:
  - (FL 500) TEMPORAL_LOGICAL_TRANSACTION_TIME
  - (FL 500) TEMPORAL_LOGICAL_TRANSACTIONS

See Built-in global variables (Db2 SQL) for details.
New reserved words in Db2 12

Db2 12 introduces the following SQL reserved words:

- (FL 504) CURRENT_SERVER
- (FL 504) CURRENT_TIMEZONE
- (FL 500) LIMIT
- (FL 500) OFFSET

For the complete list, see Reserved words (Db2 SQL).

Related reference

Statements (Db2 SQL)
Reserved words (Db2 SQL)
Chapter 16. Utility changes in Db2 12

You can use this information to plan for utility changes in Db2 12.

Important: Db2 12 introduces continuous delivery of new capabilities and enhancements in function levels. Most Db2 12 enhancements and changes become available only after the activation of the function level that delivers them. For more information, see Chapter 29, “Activating Db2 12 function levels,” on page 179. Each new or changed item is labeled below with a link to information about the Db2 12 function level that delivered it.

New utilities
The initial Db2 12 release introduces the following utilities:
• DSNJU008

Changes to existing utilities
The following table lists and describes these new and changed options. For information about a new option, see the information for the utility. For information about an option with a changed default value, see Utility release incompatibilities.

<table>
<thead>
<tr>
<th>Utility name</th>
<th>Summary of changes</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>BACKUP SYSTEM</strong></td>
<td><strong>New options:</strong></td>
</tr>
<tr>
<td></td>
<td>• (FL 100) ALTERNATE_CP</td>
</tr>
<tr>
<td></td>
<td>• (FL 100) DBBSG</td>
</tr>
<tr>
<td></td>
<td>• (FL 100) LGBSG</td>
</tr>
<tr>
<td><strong>CATMAINT</strong></td>
<td><strong>New options:</strong></td>
</tr>
<tr>
<td></td>
<td>• (FL 100) LEVEL (Specify a function level or catalog level. If you specify a function level Db2 determines the appropriate catalog level.)</td>
</tr>
<tr>
<td><strong>CATENFM (Db2 Utilities)</strong></td>
<td>With the introduction of single-phase migration in Db2 12, the CATENFM utility is not needed for the migration process and is removed. <strong>Related information:</strong> “Single-phase catalog migration in Db2 12” on page 33</td>
</tr>
<tr>
<td><strong>DSN1COMP</strong></td>
<td><strong>New options:</strong></td>
</tr>
<tr>
<td></td>
<td>(FL 100) LOB</td>
</tr>
<tr>
<td><strong>DSN1LOGP</strong></td>
<td><strong>New options:</strong></td>
</tr>
<tr>
<td></td>
<td>(FL 100) PART</td>
</tr>
<tr>
<td><strong>DSN1PRNT</strong></td>
<td><strong>New options:</strong></td>
</tr>
<tr>
<td></td>
<td>(FL 100) PART</td>
</tr>
</tbody>
</table>
### Table 17. New and changed utility options (continued)

<table>
<thead>
<tr>
<th>Utility name</th>
<th>Summary of changes</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>LOAD</strong></td>
<td><strong>Changed options:</strong></td>
</tr>
<tr>
<td></td>
<td>• (FL 100) You can specify multiple SYSREC data sets by providing multiple <code>ddname</code> values in a list. The data sets represented by those <code>ddname</code> values are dynamically concatenated to a single <code>ddname</code>, which becomes the input to LOAD.</td>
</tr>
<tr>
<td></td>
<td>• (FL 100) A restriction against use of the COLGROUP keyword in conjunction with PARALLEL keyword is removed. In Db2 11, you can collect cardinality, frequency, and histogram statics for column groups when the PARALLEL keyword is specified.</td>
</tr>
<tr>
<td></td>
<td>• (FL 100) FREQVAL with MOST, LEAST, or BOTH is supported for indexes.</td>
</tr>
<tr>
<td></td>
<td>• (FL 100) Support is added for new IGNORE options: PART, CONV, VALPROC, IDERROR, DUPKEY</td>
</tr>
<tr>
<td></td>
<td>• (FL 100) Support is added for concurrent access to target tables during LOAD utility processing, with REPLACE SHRLEVEL REFERENCE.</td>
</tr>
<tr>
<td></td>
<td>• (FL 100) Support for a wider range of external date formats is added with <code>DATE EXTERNAL (date-format)</code></td>
</tr>
<tr>
<td></td>
<td>• (FL 100) Support for a wider range of external time formats is added with <code>TIME EXTERNAL (time-format)</code></td>
</tr>
<tr>
<td></td>
<td>• (FL 100) For inline statistics, if you specify the FREQVAL keyword and omit the COUNT keyword, Db2 automatically determines the number of frequently occurring values to collect so that the data distribution is no longer skewed.</td>
</tr>
<tr>
<td></td>
<td><strong>New options:</strong></td>
</tr>
<tr>
<td></td>
<td>• (FL 100) CCSID in a field specification</td>
</tr>
<tr>
<td></td>
<td>• (FL 100) DRAIN_WAIT</td>
</tr>
<tr>
<td></td>
<td>• (FL 100) IGNORE</td>
</tr>
<tr>
<td></td>
<td>• (FL 100) INVALIDATECACHE</td>
</tr>
<tr>
<td></td>
<td>• (FL 100) NOCHECKPEND</td>
</tr>
<tr>
<td></td>
<td>• (FL 100) RETRY</td>
</tr>
<tr>
<td></td>
<td>• (FL 100) RETRY_DELAY</td>
</tr>
<tr>
<td></td>
<td>• (FL 100) SWITCHTIME</td>
</tr>
<tr>
<td></td>
<td>• (FL 100) STATCLGMEMSRT</td>
</tr>
<tr>
<td><strong>MODIFY RECOVERY</strong></td>
<td><strong>New options:</strong></td>
</tr>
<tr>
<td></td>
<td>• (FL 500) DELETEDS</td>
</tr>
<tr>
<td></td>
<td>• (FL 500) FLASHCOPY ONLY</td>
</tr>
<tr>
<td></td>
<td>• (FL 100) NOCOPYPEND</td>
</tr>
<tr>
<td></td>
<td><strong>Other changes:</strong></td>
</tr>
<tr>
<td></td>
<td>(FL 100) When MODIFY RECOVERY is run at the table space level, a SYSIBM.SYSCOPY row with ICTYPE='M' and STYPE='R' is inserted for each partition of the table space and any partitioned indexes with the COPY YES attribute that the MODIFY utility processes. This differs from previous versions where only one SYSCOPY record was inserted for the entire table space, regardless of partitions.</td>
</tr>
</tbody>
</table>
### Table 17. New and changed utility options (continued)

<table>
<thead>
<tr>
<th>Utility name</th>
<th>Summary of changes</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>REBUILD INDEX</strong></td>
<td><strong>Changed options</strong></td>
</tr>
<tr>
<td></td>
<td>• (FL 100) For inline statistics, if you specify the FREQVAL keyword and omit the COUNT keyword, Db2 automatically determines the number of frequently occurring values to collect so that the data distribution is no longer skewed.</td>
</tr>
<tr>
<td></td>
<td><strong>New options:</strong></td>
</tr>
<tr>
<td></td>
<td>• (FL 100) BOTH</td>
</tr>
<tr>
<td></td>
<td>• (FL 100) LEAST</td>
</tr>
<tr>
<td></td>
<td>• (FL 100) MOST</td>
</tr>
<tr>
<td></td>
<td>• (FL 100) INVALIDATECACHE</td>
</tr>
<tr>
<td><strong>RECOVER</strong></td>
<td><strong>New options:</strong></td>
</tr>
<tr>
<td></td>
<td>• (FL 100) ALTERNATE_CP</td>
</tr>
<tr>
<td></td>
<td>• (FL 100) FLASHCOPY_PPRCP</td>
</tr>
<tr>
<td></td>
<td>• (FL 100) INLCOPY</td>
</tr>
<tr>
<td></td>
<td>• (FL 100) NOSYSCOPY</td>
</tr>
<tr>
<td></td>
<td>• (FL 100) SCOPE</td>
</tr>
<tr>
<td><strong>REORG INDEX</strong></td>
<td><strong>Changed options</strong></td>
</tr>
<tr>
<td></td>
<td>• (FL 100) For inline statistics, if you specify the FREQVAL keyword and omit the COUNT keyword, Db2 automatically determines the number of frequently occurring values to collect so that the data distribution is no longer skewed.</td>
</tr>
<tr>
<td></td>
<td><strong>New options:</strong></td>
</tr>
<tr>
<td></td>
<td>• (FL 100) BOTH</td>
</tr>
<tr>
<td></td>
<td>• (FL 100) LEAST</td>
</tr>
<tr>
<td></td>
<td>• (FL 100) MOST</td>
</tr>
<tr>
<td></td>
<td>• (FL 100) INVALIDATECACHE</td>
</tr>
</tbody>
</table>
### Table 17. New and changed utility options (continued)

<table>
<thead>
<tr>
<th>Utility name</th>
<th>Summary of changes</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>REORG TABLESPACE</strong></td>
<td><strong>Changed options</strong></td>
</tr>
<tr>
<td></td>
<td>• (FL 100) FREQVAL with MOST, LEAST, or BOTH is supported for indexes.</td>
</tr>
<tr>
<td></td>
<td>• (FL 100) For inline statistics, if you specify the FREQVAL keyword and omit the COUNT keyword, Db2 automatically determines the number of frequently occurring values to collect so that the data distribution is no longer skewed.</td>
</tr>
<tr>
<td></td>
<td><strong>New options</strong></td>
</tr>
<tr>
<td></td>
<td>• (FL 500) DROP_PART NO</td>
</tr>
<tr>
<td></td>
<td>• (FL 500) DROP_PART YES</td>
</tr>
<tr>
<td></td>
<td>• INITCDDS</td>
</tr>
<tr>
<td></td>
<td>• (FL 100) INVALIDATECACHE</td>
</tr>
<tr>
<td></td>
<td>• SEARCHTIME</td>
</tr>
<tr>
<td></td>
<td>• (FL 100) STATCLGMEMSRT</td>
</tr>
<tr>
<td></td>
<td><strong>Other changes:</strong></td>
</tr>
<tr>
<td></td>
<td>• (FL 100) In Db2 12, the REORG utility processes partition-by-growth table spaces differently than in prior versions:</td>
</tr>
<tr>
<td></td>
<td>• REORG no longer sets COPY-pending status on newly created LOB table spaces during the LOG phase. Instead, REORG will create inline image copy data sets for those LOB table spaces.</td>
</tr>
<tr>
<td></td>
<td>• REORG will no longer fail during a reorganization of a partition-by-growth table space when it is unable to extend the partition range. Instead, REORG will now create a new partition for the partition-by-growth table space as long as the maximum number of partitions have not been reached.</td>
</tr>
<tr>
<td></td>
<td>• REORG TABLESPACE has a new keyword, DROP_PART NO or DROP_PART YES, that can be specified to delete the highest numbered partitions if they are empty when REORG successfully completes. In prior versions, this behavior could only be accomplished using the REORG_DROPP_PBG_PARTS subsystem parameter was set to ENABLE.</td>
</tr>
<tr>
<td></td>
<td>In addition to the REORG changes for processing partition-by-growth table spaces, REORG in Db2 12 will fail if it is using a FlashCopy and the creation of the online FlashCopy image fails.</td>
</tr>
<tr>
<td></td>
<td>Also, for REORG TABLESPACE, when SHRLEVEL REFERENCE or SHRLEVEL CHANGE is specified and blocking claimers are encountered, the DISPLAY DATABASE command output will now identify the blocking claimers on each and every drain attempt failure. In prior versions, only the most recent drain attempt failure was identified.</td>
</tr>
<tr>
<td><strong>REPAIR</strong></td>
<td><strong>New options:</strong></td>
</tr>
<tr>
<td></td>
<td>• (FL 100) WRITELOG</td>
</tr>
<tr>
<td></td>
<td>• (FL 100) INSERTVERSIONPAGES</td>
</tr>
<tr>
<td></td>
<td>• (FL 100) SETCURRENTVERSION</td>
</tr>
<tr>
<td></td>
<td><strong>Changed options:</strong></td>
</tr>
<tr>
<td></td>
<td>• (FL 100) The VERSIONS option is no longer supported, and is replaced by the CATALOG option.</td>
</tr>
</tbody>
</table>
Table 17. New and changed utility options (continued)

<table>
<thead>
<tr>
<th>Utility name</th>
<th>Summary of changes</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>RESTORE SYSTEM</strong></td>
<td>New options:&lt;br&gt;  - (FL 100) ALTERNATE_CP&lt;br&gt;  - (FL 100) FLASHCOPY_PPRCP&lt;br&gt;  - (FL 100) RESTOREBEFORE</td>
</tr>
<tr>
<td><strong>RUNSTATS</strong></td>
<td>New options:&lt;br&gt;  - (FL 100) INVALIDATECACHE&lt;br&gt;  - (FL 100) REGISTER&lt;br&gt;  - (FL 100) STATCLGMEMSRT&lt;br&gt; Other changes:&lt;br&gt;  - (FL 100) RUNSTATS collects frequency and histogram statistics for XML indexes in the following catalog table columns:&lt;br&gt;     - KEYVALUE, FREQUENCYF, and TYPE in SYIBM.SYSINDEXES&lt;br&gt;     - CARDF, FREQUENCYF, HIGHVALUE, LOWVALUE, TYPE, QUANTILENO in SYIBM.SYSKEYTGTDIST&lt;br&gt; Db2 can use frequency and histogram statistics for XML indexes to estimate the filter factor of XMLEXISTS predicates when the following conditions are true:&lt;br&gt;     - The predicate is one of the following types:&lt;br&gt;       - Equality predicates: =&lt;br&gt;       - Range predicates: &gt;, &gt;=, &gt;_, &lt;=&lt;br&gt;     - The right side of the predicate is a literal value.&lt;br&gt;     - Frequency or histogram statistics are collected for the XML index that matches the XMLEXISTS predicate. Related information:&lt;br&gt;     - Improving filter factors by collecting cardinality and frequency statistics (Db2 Performance)&lt;br&gt;     - Histogram statistics (Db2 Performance)&lt;br&gt;     - Collection of statistics on XML objects (Db2 Utilities)&lt;br&gt;  - (FL 100) If you specify the FREQVAL keyword and omit the COUNT keyword, Db2 automatically determines the number of frequently occurring values to collect so that the data distribution is no longer skewed.</td>
</tr>
<tr>
<td><strong>UNLOAD</strong></td>
<td>New options:&lt;br&gt;  - (FL 100) CCSID in a field specifications&lt;br&gt;  - (FL 100) REGISTER</td>
</tr>
</tbody>
</table>

**Related concepts**<br>Db2 utilities (Db2 Utilities)
Chapter 17. Catalog changes in Db2 12

Use this information to plan for catalog changes in Db2 12.

**Important:** Db2 12 introduces continuous delivery of new capabilities and enhancements in function levels. Most Db2 12 enhancements and changes become available only after the activation of the function level that delivers them. For more information, see Chapter 29, “Activating Db2 12 function levels,” on page 179. Each new or changed item is labeled below with a link to information about the Db2 12 function level that delivered it.

### New catalog tables in Db2 12

**GUPI** (FL 100) Db2 12 introduces the following new catalog tables in catalog level V12R1M500, which is the catalog level for both function level 100 and function level 500:

- SYSDYNQRY catalog table (Db2 SQL)
- SYSDYNQRYDEP catalog table (Db2 SQL)
- SYSDYNQRY_EXPL catalog table (Db2 SQL)
- SYSDYNQRY_OPL catalog table (Db2 SQL)
- SYSDYNQRY_SHTEL catalog table (Db2 SQL)
- SYSDYNQRY_SPAL catalog table (Db2 SQL)
- SYSDYNQRY_TXTL catalog table (Db2 SQL)
- SYSINDEXCONTROL catalog table (Db2 SQL)
- SYSIBM.SYSIXSPACESTATS_H, which is described in Temporal versioning for Db2 catalog tables (Db2 SQL)
- SYSLEVELUPDATES catalog table (Db2 SQL)
- SYSSESSION catalog table (Db2 SQL)
- SYSSESSION_EX catalog table (Db2 SQL)
- SYSSESSION_STATUS catalog table (Db2 SQL)
- SYSIBM.SYSTABSPACESTATS_H, which is described in Temporal versioning for Db2 catalog tables (Db2 SQL)

The following catalog tables are added but not used:

- SYSAUDITPOLICIES_H
- SYSCOLAUTH_H
- SYSCONTEXT_H
- SYSCONTEXTAUTHID_H
- SYSCONTROLS_H
- SYSCTXTTRUSTATTR_H
- SYSDBAUTH_H
- SYSPACKAUTH_H
- SYSPLANAUTH_H
- SYSRESAUTH_H
- SYSROUTINEAUTH_H
- SYSSCHEMAAUTH_H
- SYSSEQUENCEAUTH_H
- SYSTABAUTH_H
### Changed catalog tables in Db2 12

The following table summarizes new and changed catalog table columns in Db2 12. Changes to columns include new column descriptions, new values, and changed data types and lengths.

**Table 18. Summary of new and revised catalog table columns**

<table>
<thead>
<tr>
<th>Catalog table name</th>
<th>New column</th>
<th>Revised column</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSAUDITPOLICIES catalog table (Db2 SQL)</td>
<td>(FL 100)</td>
<td>The following columns:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SYS_START</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SYS_END</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TRANS_START</td>
</tr>
<tr>
<td>SYSCOLAUTH catalog table (Db2 SQL)</td>
<td>(FL 100)</td>
<td>The following columns:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SYS_START</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SYS_END</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TRANS_START</td>
</tr>
<tr>
<td>SYSCOLUMNS catalog table (Db2 SQL)</td>
<td></td>
<td>ENCODING_SCHEME</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(FL 100) PERIOD</td>
</tr>
<tr>
<td>SYSCONTEXTAUTHIDS catalog table (Db2 SQL)</td>
<td>(FL 100)</td>
<td>The following columns:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SYS_START</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SYS_END</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TRANS_START</td>
</tr>
<tr>
<td>SYSCONTEXT catalog table (Db2 SQL)</td>
<td>(FL 100)</td>
<td>The following columns:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SYS_START</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SYS_END</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TRANS_START</td>
</tr>
<tr>
<td>SYSCONTROLS catalog table (Db2 SQL)</td>
<td>(FL 100)</td>
<td>The following columns:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SYS_START</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SYS_END</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TRANS_START</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• REGENERATETS</td>
</tr>
<tr>
<td>SYSCOPY catalog table (Db2 SQL)</td>
<td></td>
<td>(FL 100) The following columns:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• DSVOLSER</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ICTYPE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• STYPE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TTYPE</td>
</tr>
<tr>
<td>Catalog table name</td>
<td>New column</td>
<td>Revised column</td>
</tr>
<tr>
<td>--------------------</td>
<td>------------</td>
<td>----------------</td>
</tr>
<tr>
<td>SYSCTXTRUSTATTRS catalog table (Db2 SQL)</td>
<td>(FL 100) The following columns: • SYS_START • SYS_END • TRANS_START</td>
<td></td>
</tr>
<tr>
<td>SYSDBAUTH catalog table (Db2 SQL)</td>
<td>(FL 100) The following columns: • SYS_START • SYS_END • TRANS_START</td>
<td></td>
</tr>
<tr>
<td>SYSDEPENDENCIES catalog table (Db2 SQL)</td>
<td>• (FL 100) DTYPE</td>
<td></td>
</tr>
<tr>
<td>SYSENVIRONMENT catalog table (Db2 SQL)</td>
<td>(FL 100) The following columns: • CREATEDTS • APPLCOMPAT</td>
<td></td>
</tr>
<tr>
<td>SYSINDEXSPACESTATS catalog table (Db2 SQL)</td>
<td>(FL 100) The following columns: • SYS_START • SYS_END • TRANS_START</td>
<td></td>
</tr>
<tr>
<td>SYSINDEXPART catalog table (Db2 SQL)</td>
<td>(FL 100) The following columns: • DSSIZE • PAGENUM • LIMITKEY_EXTERNAL</td>
<td></td>
</tr>
<tr>
<td>SYSINDEXES catalog table (Db2 SQL)</td>
<td>• (FL 502) KEYLABEL • (FL 100) The following columns: • PQTY • STORTYPE • STORNAME • VCATNAME • FREEPAGE • PCTFREE • GBPCACHE • SEQHTY • ENFORCED_CONS • IMPLICIT • REGENERATETS</td>
<td></td>
</tr>
<tr>
<td>SYSINDEXPART catalog table (Db2 SQL)</td>
<td>• SPACE</td>
<td></td>
</tr>
<tr>
<td>Catalog table name</td>
<td>New column</td>
<td>Revised column</td>
</tr>
<tr>
<td>----------------------------------------</td>
<td>----------------------------------------------------------------------------</td>
<td>---------------------------------------</td>
</tr>
<tr>
<td>SYSKEYCOLUSE catalog table (Db2 SQL)</td>
<td>• (FL 100) PERIOD</td>
<td></td>
</tr>
<tr>
<td>SYSKEYS catalog table (Db2 SQL)</td>
<td>• (FL 100) CREATEDTS</td>
<td>• (FL 100) PERIOD</td>
</tr>
<tr>
<td>SYSKEYTGTDIST catalog table (Db2 SQL)</td>
<td>• (FL 100) NUMKEYS</td>
<td></td>
</tr>
<tr>
<td>SYSPENDINGDDL catalog table (Db2 SQL)</td>
<td>(FL 100) The following columns:</td>
<td>• (FL 100) PARTITION</td>
</tr>
<tr>
<td></td>
<td>• REORG_SCOPE_LOWPART</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• REORG_SCOPE_HIGHPART</td>
<td></td>
</tr>
<tr>
<td>SYSPACKAGE catalog table (Db2 SQL)</td>
<td>(FL 505)The following column:</td>
<td>(FL 100) The following columns:</td>
</tr>
<tr>
<td></td>
<td>• COPYID</td>
<td>• TYPE</td>
</tr>
<tr>
<td></td>
<td>(FL 100) The following columns:</td>
<td>• HOSTLANG</td>
</tr>
<tr>
<td></td>
<td>• APREUSE_NO_FL</td>
<td>• VERSION</td>
</tr>
<tr>
<td></td>
<td>• APREUSE_NO_TS</td>
<td>• DEFERPREPARE</td>
</tr>
<tr>
<td></td>
<td>• CONC_STMT</td>
<td>• APPLCOMPAT</td>
</tr>
<tr>
<td></td>
<td>• FUNCTION_LVL</td>
<td>• DTYPE</td>
</tr>
<tr>
<td></td>
<td>• ORIGIN</td>
<td>• KEEPDYNAMIC</td>
</tr>
<tr>
<td>SYSPACKAUTH catalog table (Db2 SQL)</td>
<td>(FL 100) The following columns:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• SYS_START</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• SYS_END</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• TRANS_START</td>
<td></td>
</tr>
<tr>
<td>SYSPACKCOPY catalog table (Db2 SQL)</td>
<td>(FL 505)The following column:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• COPYID</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(FL 100) The following columns:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• ORIGIN</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• APREUSE_NO_FL</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• APREUSE_NO_TS</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• CONC_STMT</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• FUNCTION_LVL</td>
<td></td>
</tr>
<tr>
<td>SYSPACKDEP catalog table (Db2 SQL)</td>
<td>(FL 505)The following column:</td>
<td>• (FL 100) DTYPE</td>
</tr>
<tr>
<td></td>
<td>• COPYID</td>
<td></td>
</tr>
<tr>
<td>SYSPACKSTMT catalog table (Db2 SQL)</td>
<td>(FL 505)The following column:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• COPYID</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(FL 100) The following columns:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• QUERYID</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• QUERY_HASH</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• QUERY_HASH_VERSION</td>
<td></td>
</tr>
</tbody>
</table>
Table 18. Summary of new and revised catalog table columns (continued)

<table>
<thead>
<tr>
<th>Catalog table name</th>
<th>New column</th>
<th>Revised column</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSPLANAUTH catalog table (Db2 SQL)</td>
<td>(FL 100)</td>
<td>The following columns:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SYS_START</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SYS_END</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TRANS_START</td>
</tr>
<tr>
<td>SYSRELS catalog table (Db2 SQL)</td>
<td>(FL 100)</td>
<td>The following columns:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• FUNCTION_LVL</td>
</tr>
<tr>
<td>SYSRELS catalog table (Db2 SQL)</td>
<td></td>
<td>(FL 100) CHECKEXISTING-DATA</td>
</tr>
<tr>
<td>SYSRESAUTH catalog table (Db2 SQL)</td>
<td>(FL 100)</td>
<td>The following columns:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SYS_START</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SYS_END</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TRANS_START</td>
</tr>
<tr>
<td>SYSROUTINEAUTH catalog table (Db2 SQL)</td>
<td>(FL 100)</td>
<td>The following columns:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SYS_START</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SYS_END</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TRANS_START</td>
</tr>
<tr>
<td>SYSRoutines catalog table (Db2 SQL)</td>
<td></td>
<td>• WRAPPED</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• REGENERATETS</td>
</tr>
<tr>
<td>SYSSEQUENCES catalog table (Db2 SQL)</td>
<td></td>
<td>(FL 100) MAXASSIGNEDVAL</td>
</tr>
<tr>
<td>SYSSCHEMAUTH catalog table (Db2 SQL)</td>
<td>(FL 100)</td>
<td>The following columns:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SYS_START</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SYS_END</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TRANS_START</td>
</tr>
<tr>
<td>SYSSTOGROUP catalog table (Db2 SQL)</td>
<td></td>
<td>(FL 502) KEYLABEL</td>
</tr>
<tr>
<td>SYSTABAUTH catalog table (Db2 SQL)</td>
<td>(FL 100)</td>
<td>The following columns:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• UNLOADAUTH</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SYS_START</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SYS_END</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TRANS_START</td>
</tr>
<tr>
<td>Catalog table name</td>
<td>New column</td>
<td>Revised column</td>
</tr>
<tr>
<td>----------------------------------------</td>
<td>----------------</td>
<td>-------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>SYSTABCONST catalog table (Db2 SQL)</td>
<td></td>
<td>• (FL 100) TYPE</td>
</tr>
<tr>
<td>SYSTABLEPART catalog table (Db2 SQL)</td>
<td>(FL 100) The following columns: • SPACE</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TYPE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• PAGENUM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• BPOOL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• PGSIZE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• DSSIZE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• MEMBER_CLUSTER</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• COMPRESSRATIO</td>
</tr>
<tr>
<td>SYSTABLESPACE catalog table (Db2 SQL)</td>
<td>(FL 502) KEYLABEL</td>
<td>• (FL 100) The following columns:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>– PAGENUM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>– PQTY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>– STORTYPE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>– STORNAME</td>
</tr>
<tr>
<td></td>
<td></td>
<td>– VCATNAME</td>
</tr>
<tr>
<td></td>
<td></td>
<td>– FREEPAGE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>– PCTFREE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>– COMPRESS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>– COMPRESSRATIO</td>
</tr>
<tr>
<td></td>
<td></td>
<td>– GBPCACHE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>– TRACKMOD</td>
</tr>
<tr>
<td></td>
<td></td>
<td>– SECQTYI</td>
</tr>
<tr>
<td></td>
<td></td>
<td>– PCTFREE_UPD</td>
</tr>
<tr>
<td></td>
<td></td>
<td>– PCTFREE_UPD_CALC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>– INSERTALG</td>
</tr>
<tr>
<td>SYSTABLES catalog table (Db2 SQL)</td>
<td>(FL 502) KEYLABEL</td>
<td>• (FL 100) TABLESTATUS</td>
</tr>
<tr>
<td></td>
<td>(FL 100) REGENERATETS</td>
<td>• (FL 100) TABLESTATUS</td>
</tr>
<tr>
<td>SYSTABLESPACESTATS catalog table (Db2 SQL)</td>
<td>(FL 100) The following columns:</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• GETPAGES</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SYS_START</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SYS_END</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TRANS_START</td>
</tr>
<tr>
<td>SYSTATFEEDBACK catalog table (Db2 SQL)</td>
<td></td>
<td>• (FL 100) REASON</td>
</tr>
</tbody>
</table>
### Table 18. Summary of new and revised catalog table columns (continued)

<table>
<thead>
<tr>
<th>Catalog table name</th>
<th>New column</th>
<th>Revised column</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTRIGGERS catalog table (Db2 SQL)</td>
<td>(FL 100) The following columns:</td>
<td>• ACTIVE&lt;br&gt;• ALTEREDTS&lt;br&gt;• ASUTIME&lt;br&gt;• DEBUG_MODE&lt;br&gt;• ORIGINAL_CONTOK&lt;br&gt;• REGENERATETS&lt;br&gt;• SQLPL&lt;br&gt;• VERSION&lt;br&gt;• WLM_ENVIRONMENT&lt;br&gt;• WRAPPED</td>
</tr>
<tr>
<td>SYSUSERAUTH catalog table (Db2 SQL)</td>
<td>(FL 100) The following columns:</td>
<td>• SYS_START&lt;br&gt;• SYS_END&lt;br&gt;• TRANS_START</td>
</tr>
<tr>
<td>SYSVARIABLES catalog table (Db2 SQL)</td>
<td>(FL 100) The following columns:</td>
<td>• DATATYPEID&lt;br&gt;• SOURCETYPEID&lt;br&gt;• DEFAULT&lt;br&gt;• LENGTH&lt;br&gt;• CCSID</td>
</tr>
<tr>
<td>SYSVIEWDEP catalog table (Db2 SQL)</td>
<td>(FL 100) The following columns:</td>
<td>• SYS_START&lt;br&gt;• SYS_END&lt;br&gt;• TRANS_START</td>
</tr>
</tbody>
</table>

### Catalog index changes in Db2 12

The following catalog indexes are new in Db2 12:

<table>
<thead>
<tr>
<th>Catalog table</th>
<th>Index</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSCOLUMNS</td>
<td>(FL 100) DSNDCX06</td>
</tr>
<tr>
<td>SYSCONTROLS_DESC_H</td>
<td>(FL 100) DSNTRX04</td>
</tr>
<tr>
<td>SYSCONTROLS_RTXT_H</td>
<td>(FL 100) DSNTRX03</td>
</tr>
</tbody>
</table>
Table 19. New catalog indexes in Db2 12 (continued)

<table>
<thead>
<tr>
<th>Catalog table</th>
<th>Index</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSDYNQRY</td>
<td>(FL 100) The following indexes:</td>
</tr>
<tr>
<td></td>
<td>• DSNDQX01</td>
</tr>
<tr>
<td></td>
<td>• DSNDQX02</td>
</tr>
<tr>
<td></td>
<td>• DSNDQX11</td>
</tr>
<tr>
<td>SYSDYNQRYDEP</td>
<td>(FL 100) The following indexes:</td>
</tr>
<tr>
<td></td>
<td>• DSNDQX03</td>
</tr>
<tr>
<td></td>
<td>• DSNDQX04</td>
</tr>
<tr>
<td></td>
<td>• DSNDQX05</td>
</tr>
<tr>
<td></td>
<td>• DSNDQX12</td>
</tr>
<tr>
<td>SYSDYNQRY_EXPL</td>
<td>(FL 100) DSNDQX08</td>
</tr>
<tr>
<td>SYSDYNQRY_OPL</td>
<td>(FL 100) DSNDQX10</td>
</tr>
<tr>
<td>SYSDYNQRY_SHTEL</td>
<td>(FL 100) DSNDQX09</td>
</tr>
<tr>
<td>SYSDYNQRY_SPAL</td>
<td>(FL 100) DSNDQX07</td>
</tr>
<tr>
<td>SYSDYNQRY_TXTL</td>
<td>(FL 100) DSNDQX06</td>
</tr>
<tr>
<td>SYSLEVELSUPDATED</td>
<td>(FL 100) The following indexes:</td>
</tr>
<tr>
<td></td>
<td>• DSNLVX01</td>
</tr>
<tr>
<td></td>
<td>• DSNLVX02</td>
</tr>
<tr>
<td>SYSQUERY</td>
<td>(FL 100) DSNQYX04</td>
</tr>
<tr>
<td>SYSSESSION</td>
<td>(FL 100) DSNSNX02</td>
</tr>
<tr>
<td>SYSSESSION_DATA</td>
<td>(FL 100) DSNSNX03</td>
</tr>
<tr>
<td>SYSSESSION_EX</td>
<td>(FL 100) The following indexes:</td>
</tr>
<tr>
<td></td>
<td>• DSNSNX04</td>
</tr>
<tr>
<td></td>
<td>• DSNSNX05</td>
</tr>
<tr>
<td>SYSSESSION_GV</td>
<td>(FL 100) SYSSESSION_GV</td>
</tr>
<tr>
<td>SYSSESSION_STATUS</td>
<td>(FL 100) DSNSNX06</td>
</tr>
<tr>
<td>SYSVARIABLES</td>
<td>(FL 100) DSNOVX01</td>
</tr>
</tbody>
</table>

Global variable changes in Db2 12

Db2 12 introduces the following new built-in global variables:

- (FL 100) CATALOG_LEVEL
- (FL 100) DEFAULT_SQLLEVEL
- (FL 100) PRODUCTID_EXT
- (FL 503) REPLICATION_OVERRIDE
- (FL 100) TEMPORAL_LOGICAL_TRANSACTION_TIME
- (FL 100) TEMPORAL_LOGICAL_TRANSACTIONS

For descriptions, see Built-in global variables (Db2 SQL).
Chapter 18. EXPLAIN table changes

The initial Db2 12 release introduces changes to the formats of certain EXPLAIN tables.

**Important:** Db2 12 introduces continuous delivery of new capabilities and enhancements in function levels. Most Db2 12 enhancements and changes become available only after the activation of the function level that delivers them. For more information, see Chapter 29, “Activating Db2 12 function levels,” on page 179. Each new or changed item is labeled below with a link to information about the Db2 12 function level that delivered it.

For the current formats, and sample CREATE TABLE statements for PLAN_TABLE and the other EXPLAIN tables, see member DSNTESC of the prefix.SDSNSAMP library. For the complete set of column descriptions for each table, see EXPLAIN tables (Db2 Performance) and Input tables (Db2 Performance).

**Important:** It is best to convert EXPLAIN tables to Db2 12 format during migration, or soon after migration. In Db2 12, the EXPLAIN function supports tables in Db2 12 or Db2 11 formats only. However, Db2 11 format EXPLAIN tables are deprecated. If you invoke EXPLAIN and Db2 11 tables are used, Db2 issues SQL code +20520. If tables of an unsupported format are found, Db2 issues SQL code -20008 and the EXPLAIN operation fails. You can call the ADMIN_EXPLAIN_MAINT stored procedure to create EXPLAIN tables, upgrade them to the format for the current Db2 release, or complete other maintenance tasks. See ADMIN_EXPLAIN_MAINT stored procedure (Db2 SQL) for information about using the action input parameter to request each of these tasks.

**Result change for SQL statement EXPLAIN PACKAGE**

When Db2 processes the SQL statement EXPLAIN PACKAGE, the HINT_USED column in the PLAN_TABLE is populated with EXPLAIN PACKAGE: copy. The copy field in the HINT_USED column will be one of the following values:

- "CURRENT" - the current copy
- "PREVIOUS" - the previous copy
- "ORIGINAL" - the original copy

**DSNTIJG creates no EXPLAIN tables with SYSIBM qualifier**

The DSNTIJSG installation job no long contains CREATE statements for EXPLAIN tables with the SYSIBM qualifier. Db2 no longer uses these tables.

**New EXPLAIN tables**

The initial Db2 12 release introduces no new EXPLAIN tables.

**Changed EXPLAIN tables**

The initial Db2 12 release introduces the following new and changed EXPLAIN table columns:
### Table 20. List of new and changed EXPLAIN tables

<table>
<thead>
<tr>
<th>EXPLAIN table</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>PLAN_TABLE (Db2 Performance)</td>
<td><strong>New columns</strong></td>
</tr>
<tr>
<td></td>
<td>• PER_STMT_ID</td>
</tr>
<tr>
<td></td>
<td><strong>Changed columns</strong></td>
</tr>
<tr>
<td></td>
<td>• COLUMN_FN_EVAL</td>
</tr>
<tr>
<td></td>
<td>• PRIMARY_ACCESSTYPE</td>
</tr>
<tr>
<td></td>
<td>• TNAME</td>
</tr>
<tr>
<td>DSN_COLDIST_TABLE (Db2 Performance)</td>
<td><strong>New columns</strong></td>
</tr>
<tr>
<td></td>
<td>• PER_STMT_ID</td>
</tr>
<tr>
<td>DSN_KEYTGTDIST_TABLE (Db2 Performance)</td>
<td><strong>New columns</strong></td>
</tr>
<tr>
<td></td>
<td>• PER_STMT_ID</td>
</tr>
<tr>
<td>DSN_DETCOST_TABLE (Db2 Performance)</td>
<td><strong>New columns</strong></td>
</tr>
<tr>
<td></td>
<td>• BLOCK_FETCH</td>
</tr>
<tr>
<td></td>
<td>• PER_STMT_ID</td>
</tr>
<tr>
<td>DSN_FUNCTION_TABLE (Db2 Performance)</td>
<td><strong>New columns</strong></td>
</tr>
<tr>
<td></td>
<td>• PER_STMT_ID</td>
</tr>
<tr>
<td>DSN_PGRANGE_TABLE (Db2 Performance)</td>
<td><strong>New columns</strong></td>
</tr>
<tr>
<td></td>
<td>• PER_STMT_ID</td>
</tr>
<tr>
<td>DSN_PGROUP_TABLE (Db2 Performance)</td>
<td><strong>New columns</strong></td>
</tr>
<tr>
<td></td>
<td>• PER_STMT_ID</td>
</tr>
<tr>
<td>DSN_PREDICAT_TABLE (Db2 Performance)</td>
<td><strong>New columns</strong></td>
</tr>
<tr>
<td></td>
<td>• PER_STMT_ID</td>
</tr>
<tr>
<td>DSN_PREDICATE_SELECTIVITY table (Db2 Performance)</td>
<td><strong>New columns</strong></td>
</tr>
<tr>
<td></td>
<td>• PER_STMT_ID</td>
</tr>
<tr>
<td>DSN_PTASK_TABLE (Db2 Performance)</td>
<td><strong>New columns</strong></td>
</tr>
<tr>
<td></td>
<td>• PER_STMT_ID</td>
</tr>
<tr>
<td>DSN_SORT_TABLE (Db2 Performance)</td>
<td><strong>New columns</strong></td>
</tr>
<tr>
<td></td>
<td>• PER_STMT_ID</td>
</tr>
<tr>
<td>DSN_SORTKEY_TABLE (Db2 Performance)</td>
<td><strong>New columns</strong></td>
</tr>
<tr>
<td></td>
<td>• PER_STMT_ID</td>
</tr>
<tr>
<td>DSN_STAT_FEEDBACK (Db2 Performance)</td>
<td><strong>Changed columns</strong></td>
</tr>
<tr>
<td></td>
<td>• REASON</td>
</tr>
<tr>
<td></td>
<td>• STALE</td>
</tr>
<tr>
<td>EXPLAIN table</td>
<td>Changes</td>
</tr>
<tr>
<td>----------------</td>
<td>---------</td>
</tr>
<tr>
<td>DSN_STATEMENT_CACHE_TABLE (Db2 Performance)</td>
<td>New columns</td>
</tr>
<tr>
<td></td>
<td>• APPLCOMPAT</td>
</tr>
<tr>
<td></td>
<td>• CNO</td>
</tr>
<tr>
<td></td>
<td>• PER_STMT_ID</td>
</tr>
<tr>
<td></td>
<td>• QUERY_HASH</td>
</tr>
<tr>
<td></td>
<td>• QUERY_HASH_VERSION</td>
</tr>
<tr>
<td></td>
<td>• STABILIZED</td>
</tr>
<tr>
<td></td>
<td>• STAT_SUS_CHILDLOCKS</td>
</tr>
<tr>
<td></td>
<td>• STAT_SUS_OTHERLOCKS</td>
</tr>
<tr>
<td></td>
<td>• STAT_SUS_PPLLOCKS</td>
</tr>
<tr>
<td></td>
<td>• STAT_SUS_PAGELOCKS</td>
</tr>
<tr>
<td></td>
<td>• STAT_SUS_OTHERPLOCKS</td>
</tr>
<tr>
<td></td>
<td>• STBLGRP</td>
</tr>
<tr>
<td></td>
<td>• STAT_SUS_PIPE</td>
</tr>
<tr>
<td></td>
<td>• STAT_SUS_PQSYNC</td>
</tr>
<tr>
<td>DSN_STRUCT_TABLE (Db2 Performance)</td>
<td>New columns</td>
</tr>
<tr>
<td></td>
<td>• PER_STMT_ID</td>
</tr>
</tbody>
</table>

Related concepts
Investigating SQL performance by using EXPLAIN (Db2 Performance)
Input tables (Db2 Performance)

Related tasks
Migration step 25: Convert EXPLAIN tables to the current format (Db2 Installation and Migration)

Related reference
EXPLAIN (Db2 SQL)
EXPLAIN tables (Db2 Performance)
Chapter 19. IFCID changes

Db2 12 introduces IFCID changes.

**Important:** Db2 12 introduces continuous delivery of new capabilities and enhancements in function levels. Most Db2 12 enhancements and changes become available only after the activation of the function level that delivers them. For more information, see Chapter 29, “Activating Db2 12 function levels,” on page 179. Each new or changed item is labeled below with a link to information about the Db2 12 function level that delivered it.

### New trace records

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

<table>
<thead>
<tr>
<th>IFCID</th>
<th>Trace type and class</th>
<th>Mapping macro</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0382</td>
<td>Accounting class 3</td>
<td>DSNDQW05</td>
<td>Records the beginning of a suspend for parallel task synchronization</td>
</tr>
<tr>
<td></td>
<td>Accounting class 8</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Monitor class 3</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Monitor class 8</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Accounting, Monitor</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0383</td>
<td>Accounting class 3</td>
<td>DSNDQW05</td>
<td>Records the end of a suspend for parallel task synchronization</td>
</tr>
<tr>
<td></td>
<td>Accounting class 8</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Monitor class 3</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Monitor class 8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0389</td>
<td>Statistics class 8</td>
<td>DSNDQW05</td>
<td>Records information about indexes that have structures allocated for fast index traversal</td>
</tr>
<tr>
<td>0413</td>
<td>Accounting class 3</td>
<td>DSNDQW05</td>
<td>Records the beginning of a wait for a pipe for an INSERT statement that is using insert algorithm 2</td>
</tr>
<tr>
<td></td>
<td>Accounting class 8</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Monitor class 3</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Monitor class 8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0414</td>
<td>Accounting class 3</td>
<td>DSNDQW05</td>
<td>Records the end of a wait for a pipe for an INSERT statement that is using insert algorithm 2</td>
</tr>
<tr>
<td></td>
<td>Accounting class 8</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Monitor class 3</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Monitor class 8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0477</td>
<td>Performance class 4</td>
<td>DSNDQW05</td>
<td>Records allocation and deallocation of structures for fast index traversal</td>
</tr>
</tbody>
</table>

### Changes to selected trace records

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.
<table>
<thead>
<tr>
<th>Enhancement</th>
<th>IFCID</th>
<th>Description of changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adaptive index processing</td>
<td>0002, 0125</td>
<td>Fields are added to record:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• RID list processing statistics for adaptive index processing</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Details about adaptive index processing that was performed</td>
</tr>
<tr>
<td>Autonomous procedures</td>
<td>0003</td>
<td>Fields are added to record accounting information for autonomous procedures.</td>
</tr>
<tr>
<td>Db2 exploitation of asynchronous cross-invalidation for coupling facility cache structures (APAR PH05193)</td>
<td>0002</td>
<td>Fields are added to record:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The number of IXLCACHE (cache services) requests with asynchronous cross-invalidation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The number of IXLAXISN (coupling facility asynchronous cross-invalidation service) calls</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The number of suspensions of the IXLAXISN service that occurred during waits for asynchronous cross-invalidation operations to complete.</td>
</tr>
<tr>
<td></td>
<td>0003</td>
<td>A field is added to record the number of suspensions of the IXLAXISN service that occurred during waits for cross-invalidation operations to complete.</td>
</tr>
<tr>
<td>Dynamic SQL plan stability</td>
<td>0002, 0021, 0029, 0030, 0316</td>
<td>Fields are added to record:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The number of requests and satisfied requests for stabilized dynamic SQL statements</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Locking for dynamic plan stability operations</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Start and end of an EDM request for the SYSIBM.SYSDYNQRY table</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Information on stabilized dynamic SQL statements in the dynamic statement cache</td>
</tr>
<tr>
<td>Fast index traversal</td>
<td>0002</td>
<td>Fields are added to record information about when fast index traversal is used.</td>
</tr>
<tr>
<td>Insert algorithm 2</td>
<td>0002, 0003, 0018, 0058, 0239, 0316, 0401</td>
<td>Fields are added to record:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Wait times for pipes for insert algorithm 2 INSERT statements</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Information about the use of pipes for insert algorithm 2 INSERT statements</td>
</tr>
<tr>
<td>Enhanced monitoring support for insert algorithm 2 (APAR PI81731)</td>
<td>0002, 0003, 0148</td>
<td>Fields are added to record:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The number times that insert algorithm 2 was used.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The number of times that the basic insert algorithm (insert algorithm 1) is used.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The number attempts and successful attempts to re-enable pipes for insert algorithm 2 since Db2 restart</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Whether query parallelism was disabled by the resource limit facility for at least one dynamic SELECT statement in a thread.</td>
</tr>
</tbody>
</table>
### Table 22. Changed IFCIDs (continued)

<table>
<thead>
<tr>
<th>Enhancement</th>
<th>IFCID</th>
<th>Description of changes</th>
</tr>
</thead>
</table>
| IBM zHyperLink exploitation for database synchronous read I/O (APAR PI87072) and active log write I/O (APAR PH05030) | 0001 | Fields are added to record:  
  - The number of eligible active log write I/O requests with successful IBM zHyperLink processing.  
  - The number of eligible active log write I/O requests with unsuccessful IBM zHyperLink processing. |
| | 0002 | Fields are added to record:  
  - The number of successful synchronous read I/O operations that were performed using IBM zHyperLink processing.  
  - The number of synchronous read I/O operations that resulted in disk cache hits, and did not use IBM zHyperLink processing. |
| | 0003, 0148 | Fields are added to record:  
  - Wait time for synchronous read I/O operations that resulted in disk cache hits  
  - The number of synchronous read I/O operations that resulted in disk cache hits. |
| | 0003, 0148, 0239 | Fields are added to record:  
  - The number of synchronous read I/O operations that used IBM zHyperLink processing.  
  - The number of synchronous read I/O operations that resulted in disk cache hits.  
  - The amount of CPU time for successful synchronous read I/O operations that used IBM zHyperLink processing. |
| | 0006 | A flag is added to indicate whether a synchronous read I/O request occurred with or without IBM zHyperLink processing. |
| | 0007 | A flag is added to indicate whether a synchronous read I/O request with IBM zHyperLink processing was successful. |
| | 0199 | Fields are added to record the average and maximum delay and total number of pages read with synchronous read I/O and IBM zHyperLink processing. |
| Lift partition limits | 0006, 0007, 0021, 0044, 0124, 0127, 0128, 0150, 0172, 0196, 0198, 0223, 0226, 0227, 0255, 0259, 0305 | Fields are added to:  
  - Indicate whether pages in range-partitioned table spaces use absolute or relative page numbering  
  - Hold relative page numbers |
### Table 22. Changed IFCIDs (continued)

<table>
<thead>
<tr>
<th>Enhancement</th>
<th>IFCID</th>
<th>Description of changes</th>
</tr>
</thead>
</table>
| Statement-level monitoring for accelerated Hybrid Transactional and Analytical Processing (HTAP) support (APAR PH00574, PH15534) | 0003, 0316, 0401 | Fields are added to record:  
  - The amount of time in milliseconds that statements waited for execution on an accelerator because the statements needed changes that were not yet applied to a referenced table.  
  - The number of times that statement execution did not complete because changes were not applied to a referenced table before the WAITFORDATA delay protocol time period expired. |
| Transfer ownership                                                           | 0002, 0062, 0140, 0361 | Fields are added to record use of the TRANSFER OWNERSHIP statement.                                                                                                                                                                                                                                                                                      |
| Miscellaneous changes                                                        | 0002        | Fields are added to record:  
  - Details about buffer pool overflow  
  - The number of times that RID list processing was not used.  
  Fields that are related to the EDM statement pool are deleted.                                                                                                                                                                                                                        |
|                                                                             | 0002, 0003   | Fields are added to record:  
  - Additional IRLM system information                                                                                                                                                                                                                                                                                                                   |
|                                                                             | 0053        | A field is added to record the section number of the SQL statement.                                                                                                                                                                                                                                                                                      |
|                                                                             | 0058        | A field is added to record the section number of the SQL statement.                                                                                                                                                                                                                                                                                      |
|                                                                             | 0199        | Fields are added to record I/O for read and castout, in microseconds.                                                                                                                                                                                                                                                                                     |
|                                                                             | 0316        | Fields are added to record:  
  - Accumulated wait time due to global contention for L-locks and P-locks  
  - Wait times that are at the same level of granularity as the wait times in class 3 accounting data                                                                                                                                                                                          |
|                                                                             | 0401        | Fields are added to record accumulated wait time due to global contention for L-locks and P-locks.                                                                                                                                                                                                                                                      |
|                                                                             | All         | A field is added to the correlation header with the batch job step name, to correlate an accounting record with a given job step.                                                                                                                                                                                                                       |

#### Deleted trace records

IFCID 0366 is no longer supported.

---

144 Db2 12 for z/OS: What’s New?
You can use this information to plan for subsystem parameter changes in Db2 12.

**Important:** Db2 12 introduces continuous delivery of new capabilities and enhancements in function levels. Most Db2 12 enhancements and changes become available only after the activation of the function level that delivers them. For more information, see Chapter 29, “Activating Db2 12 function levels,” on page 179. Each new or changed item is labeled below with a link to information about the Db2 12 function level that delivered it.

### New subsystem parameters

- **(FL 504)** \(TS\_COMPRESSION\_TYPE\) field \((TS\_COMPRESSION\_TYPE\) subsystem parameter\) (Db2 Installation and Migration)
- **(FL 502)** \(ENCRYPTION\_KEYLABEL\) field in macro DSN6SPRM (Db2 Installation and Migration)
- **(FL 500)** The following subsystem parameters:
  - \(ALTERNATE\_COPY\_POOL\) field \((ALTERNATE\_CP\) subsystem parameter\) (Db2 Installation and Migration)
  - \(AUTH\_COMPATIBILITY\) in macro DSN6SPRM (Db2 Installation and Migration)
  - \(CACHE\_DYN\_STABILITY\) field \((CACHEDYN\_STABILIZATION\) subsystem parameter\) (Db2 Installation and Migration)
  - \(COMPRESS\_DB2\_DIR\_LOBs\) field \((COMPRESS\_DIRLOB\) subsystem parameter\) (Db2 Installation and Migration)
  - \(COPY\_FAST\_REPLICATION\) field \((COPY\_FASTREPLICATION\) subsystem parameter\) (Db2 Installation and Migration)
  - \(DDL\_MATERIALIZATION\) field \((DDL\_MATERIALIZATION\) subsystem parameter\) (Db2 Installation and Migration)
  - \(DEFAULT\_INSERT\_ALGORITHM\) field \((DEFAULT\_INSERT\_ALGORITHM\) subsystem parameter\) (Db2 Installation and Migration)
  - \(FLASHCOPY\_XRCP\) field \((FLASHCOPY\_XRCP\) subsystem parameter\) (Db2 Installation and Migration)
  - \(PAGE\_SET\_PAGE\_NUMBERING\) field \((PAGESET\_PAGENUM\) subsystem parameter\) (Db2 Installation and Migration)
  - \(PEER\_RECOVERY\) field \((PEER\_RECOVERY\) subsystem parameter\) (Db2 Installation and Migration)
  - \(STATISTICS\_PROFILE\_FEEDBACK\) field \((STATFDBK\_PROFILE\) subsystem parameter\) (Db2 Installation and Migration)
  - \(PAGE\_LEVEL\_SAMPLING\) field \((STATPGSAMP\) subsystem parameter\) (Db2 Installation and Migration)
  - \(RLF\_SCOPE\) field \((RLFENABLE\) subsystem parameter\) (Db2 Installation and Migration)
  - \(REMOTE\_STATIC\_SQL\) field \((RLFERRDSTC\) subsystem parameter\) (Db2 Installation and Migration)
  - \(STATIC\_SQL\) field \((RLFERRSTC\) subsystem parameter\) (Db2 Installation and Migration)
  - \(DB\_BACKUP\_STG\_GROUP\) field \((UTIL\_DBBSG\) subsystem parameter\) (Db2 Installation and Migration)
  - \(LOG\_BACKUP\_STG\_GRP\) field \((UTIL\_LGBSG\) subsystem parameter\) (Db2 Installation and Migration)
  - \(HSM\_MESSAGE\_DS\_HLQ\) field \((UTILS\_HSM\_MSGDS\_HLQ\) subsystem parameter\) (Db2 Installation and Migration)
- **(FL 100)** The following subsystem parameters:
  - \(CDDS\_MODE\) in macro DSN6LOGP (Db2 Installation and Migration)
- CDDS_PREFIX in macro DSN6LOGP (Db2 Installation and Migration)
- DISALLOW_SSARAUTH in macro DSN6SPRM (Db2 Installation and Migration)
- INDEX MEMORY CONTROL field (INDEX MEMORY CONTROL subsystem parameter) (Db2 Installation and Migration)
- PROFILE AUTOSTART field (PROFILE_AUTOSTART subsystem parameter) (Db2 Installation and Migration)
- REORG TS NOPAD DEFAULT (REORG_TS_NOPAD_DEFAULT subsystem parameter) (Db2 Installation and Migration)
- STATISTICS COLGROUP DATA SORT STG LIMIT field (STATCLGSRT subsystem parameter) (Db2 Installation and Migration)
- DB2 zHyperLinks SCOPE field (ZHYPERLINK subsystem parameter) (Db2 Installation and Migration)

### Changed subsystem parameters

The following table summarizes existing subsystem parameters that have new and changed options.

<table>
<thead>
<tr>
<th>Subsystem parameter</th>
<th>Description of enhancements and notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUTO BIND field (ABIND subsystem parameter) (Db2 Installation and Migration)</td>
<td>COEXIST behavior is used even when YES is specified. That is, no automatic bind occurs in release coexistence or fallback scenarios when a plan or package that was automatically bound in Db2 11 runs again in Db2 12. This change prevents repeating automatic remigration binds each time that a plan or package runs again on a different release during coexistence or fallback scenarios. For more information, see Automatic binds in coexistence (Db2 Installation and Migration).</td>
</tr>
<tr>
<td>APPL COMPAT LEVEL field (APPLCOMPAT subsystem parameter) (Db2 Installation and Migration)</td>
<td>The default value for new installations is changed to 'V12R1M500'. Support is also added for values that correspond to Db2 12 function levels.</td>
</tr>
<tr>
<td>DISALLOW_SEL INTO UNION in macro DSN6SPRM (Db2 Installation and Migration)</td>
<td>The default value is changed to YES, which means that UNION or UNION ALL as the outermost FROM clause of a SELECT INTO statement, which is invalid SQL syntax, is not allowed.</td>
</tr>
<tr>
<td>DEFAULT INSERT ALGORITHM field (DEFAULT_INSERT_ALGORITHM subsystem parameter) (Db2 Installation and Migration)</td>
<td>A new value is added, 0, which means that the basic insert algorithm (insert algorithm 1) is used regardless of the specification at the object level.</td>
</tr>
<tr>
<td>EDM SKELETON POOL SIZE field (EDM_SKELETON_POOL subsystem parameter) (Db2 Installation and Migration)</td>
<td>The default value is changed to 51200.</td>
</tr>
<tr>
<td>PREVENT_NEW_IXCTRL_PART in macro DSN6SPRM (Db2 Installation and Migration)</td>
<td>The default value is changed to YES.</td>
</tr>
<tr>
<td>PRIMARY QUANTITY field (PRIQTY subsystem parameter) (Db2 Installation and Migration)</td>
<td>The default value is changed to 125 (in cylinders—ACLUNIT is removed).</td>
</tr>
</tbody>
</table>
### Table 23. Changes to existing subsystem parameters (continued)

<table>
<thead>
<tr>
<th>Subsystem parameter</th>
<th>Description of enhancements and notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>SECONDARY QTY field (SECQTY subsystem parameter) (Db2 Installation and Migration)</td>
<td>The default value is changed to 15 (in cylinders—ACLUNIT is removed).</td>
</tr>
<tr>
<td>SMF ACCOUNTING field (SMFACCT subsystem parameter) (Db2 Installation and Migration)</td>
<td>The default value is changed to '1,2,3,7,8'.</td>
</tr>
<tr>
<td>PAGE-LEVEL SAMPLING field (STATPGSAMP subsystem parameter) (Db2 Installation and Migration)</td>
<td>(FL 505) The meaning of the default value SYSTEM is changed to mean the same as YES.</td>
</tr>
</tbody>
</table>

### Removed subsystem parameters in Db2 12

The following tables 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.

### Table 24. Removed subsystem parameters in Db2 12

<table>
<thead>
<tr>
<th>Subsystem parameter</th>
<th>Setting used in Db2 12</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALCUNIT</td>
<td>CYL, which means that primary and secondary space allocations are always obtained in cylinder units.</td>
</tr>
<tr>
<td>CACHE_DEP_TRACK_STOR_LIM</td>
<td>None, because the storage is allocated from the SCB pool.</td>
</tr>
<tr>
<td>CACHEDYN_FREELocal</td>
<td>0, which means that Db2 does not free cached dynamic statements to relieve high use of storage by dynamic SQL caching.</td>
</tr>
<tr>
<td>CATALOG (in DSN6ARVP)</td>
<td>YES, which means that Db2 requires all archive log data sets to be cataloged, regardless of whether they are allocated on disk or tape.</td>
</tr>
<tr>
<td>CHECKSETCHKP</td>
<td>NO, which means that the CHECK DATA and CHECK LOB utilities do not place objects in CHECK PENDING status if an inconsistency is detected.</td>
</tr>
<tr>
<td>CONTSTOR</td>
<td>NO, which means that Db2 does not periodically contract the working storage area for each thread.</td>
</tr>
<tr>
<td>DB2SORT</td>
<td>ENABLE, which means that Db2 utilities use the Db2 Sort for z/OS product when it is installed. If Db2 Sort for z/OS is not installed, DFSORT is used instead.</td>
</tr>
<tr>
<td>EDMPOOL</td>
<td>None, because all EDM storage is above the bar.</td>
</tr>
<tr>
<td>INDEX_IO_PARALLELISM</td>
<td>YES, which means that I/O parallelism is enabled.</td>
</tr>
<tr>
<td>LEMAX</td>
<td>None. Db2 does not require an LE token limit for any package that was bound in DB2 10 or later. Db2 12 supports only packages that are bound in DB2 10 or later.</td>
</tr>
<tr>
<td>LOBVALA</td>
<td>Db2 automatically determines the amount of storage needed for processing LOB values.</td>
</tr>
<tr>
<td>LOBVALS</td>
<td>Db2 automatically determines the amount of storage needed for processing LOB values.</td>
</tr>
<tr>
<td>MINSTOR</td>
<td>NO, which means that Db2 does not use storage management algorithms that minimize the amount of working storage that is consumed by individual threads</td>
</tr>
</tbody>
</table>
Table 24. Removed subsystem parameters in Db2 12 (continued)

<table>
<thead>
<tr>
<th>Subsystem parameter</th>
<th>Setting used in Db2 12</th>
</tr>
</thead>
<tbody>
<tr>
<td>NEWFUN dsnhdecp parameter</td>
<td>None, because the Db2 precompiler and coprocessor no longer use the NEWFUN dsnhdecp parameter.</td>
</tr>
<tr>
<td>REORG_IGNORE_FREESPACE</td>
<td>NO, which means that the REORG TABLESPACE utility uses the specified PCTFREE and FREEPAGE values.</td>
</tr>
<tr>
<td>RRF</td>
<td>ENABLE, which means that newly created table spaces or newly added partitions on partition-by-growth table spaces are created in the reordered row format. Existing basic row format table spaces are converted to reordered row format by the LOAD REPLACE or REORG TABLESPACE utilities.</td>
</tr>
<tr>
<td>SQWIDSC</td>
<td>YES, which means that the database manager invalidates but does not quiesce SQL statements in the dynamic statement cache that depend on an object that is the target of a RENAME TABLE or DROP ALIAS statement.</td>
</tr>
<tr>
<td>UTSORTAL</td>
<td>YES, which means that the utility attempts to allocate the sort work data sets. To increase the possible degree of parallelism, the utility minimizes the number of sort work data sets that are used. If the SORTNUM clause is specified in the utility control statement and the IGNSORTN subsystem parameter is set to NO, the sort program is used to allocate the sort work data sets. The utility also does not allocate the work data sets if they are allocated in the JCL.</td>
</tr>
<tr>
<td>XMLVALA</td>
<td>Db2 automatically determines the amount of storage for processing XML values.</td>
</tr>
<tr>
<td>XMLVALS</td>
<td>Db2 automatically determines the amount of storage for processing XML values.</td>
</tr>
</tbody>
</table>

**Related reference**

Directory of subsystem parameters, panel fields, and application default values (Db2 Installation and Migration)
Chapter 21. The extended 10-byte RBA and LRSN in Db2 12

Beginning in Db2 11, Db2 uses 10-byte RBA and LRSN values.

**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.

<table>
<thead>
<tr>
<th>6-byte RBA value:</th>
<th>10-byte RBA value:</th>
</tr>
</thead>
<tbody>
<tr>
<td>112233445566</td>
<td>00000000112233445566</td>
</tr>
</tbody>
</table>

**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.

<table>
<thead>
<tr>
<th>6-byte LRSN value:</th>
<th>10-byte LRSN value:</th>
</tr>
</thead>
<tbody>
<tr>
<td>112233445566</td>
<td>00112233445566000000</td>
</tr>
</tbody>
</table>

The 3 bytes on the right side might be 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**

In Db2 12, the extended RBA and LRSN values might be externalized before objects are converted in certain contexts:

**Messages**

In Db2 12, 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.

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, before you convert the BSDS.

Before you can convert the BSDS, the Db2 subsystem or data sharing member must be started in Db2 11 new-function mode.

In data sharing, you must convert the BSDS for each Db2 member separately.

At migration to Db2 12, you cannot start Db2 12 until the BSDS is converted to use the 10-byte RBA and LRSN formats. You can convert the BSDS before or during the Db2 12 migration process.

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.

For instructions, see Convert BSDS records to the extended 10-byte format: DSNTIJCB (Db2 Installation and Migration).

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 12, 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 (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, before you convert the BSDS.

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.

In data sharing, you do not need to convert all BSDSs in a data sharing group before the conversion of database objects begins. When an object is converted to the extended format, the change means that a 10-byte LRSN or RBA value is stored on the page, instead of a 6-byte value. That change does not mean that the converted object must have nonzero bytes in the extended area.

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.

<table>
<thead>
<tr>
<th>Time</th>
<th>Update</th>
<th>Content of PGLOGRBA or PGBigRBA</th>
</tr>
</thead>
<tbody>
<tr>
<td>001000000001000002</td>
<td>M10 updates TBasic</td>
<td>100000000001 A</td>
</tr>
<tr>
<td>001000000001000003</td>
<td>M10 updates TExt</td>
<td>01000000000001000003 B</td>
</tr>
<tr>
<td>001000000001000004</td>
<td>M6 updates TExt</td>
<td>01000000000020000000 C</td>
</tr>
<tr>
<td>001000000001000005</td>
<td>M6 updates TBasic</td>
<td>100000000002 D</td>
</tr>
</tbody>
</table>
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 PGBigRBA value.
- For the fifth update E, M10 must generate a larger value. A value greater than 00100000002FFFFFF 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 22. Activation details summary for function levels

Use this information to plan activation of Db2 12 function levels. Activation of a function level implies activation of all lower function levels.

Before you activate function levels, you must also resolve any incompatible changes that are introduced by the activated function levels. For a summary of such changes, see “Incompatible changes summary for function levels 501 and higher” on page 87.

Activation details for function level 505

To learn how to make new capabilities available for use in your Db2 environment, see Part 4, “Adopting new capabilities in Db2 12 continuous delivery,” on page 167.

Table 25. Function level 505 activation details

<table>
<thead>
<tr>
<th>Enabling APAR:</th>
<th>APAR PH09191.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum Db2 catalog level:</td>
<td>V12R1M505</td>
</tr>
<tr>
<td>Catalog level V12R1M505 changes:</td>
<td>A new COPYID column is added in the following tables:</td>
</tr>
<tr>
<td></td>
<td>• SYSIBM.SYSPACKAGE</td>
</tr>
<tr>
<td></td>
<td>• SYSIBM.SYSPACKCOPY</td>
</tr>
<tr>
<td></td>
<td>• SYSIBM.SYSPACKDEP</td>
</tr>
<tr>
<td></td>
<td>• SYSIBM.SYSPACKSTMT</td>
</tr>
<tr>
<td>Application compatibility control:</td>
<td>Applications must run at application compatibility level V12R1M505 or higher to use the following new capabilities:</td>
</tr>
<tr>
<td></td>
<td>• All built-in functions described in “New built-in functions for encryption and decryption with key labels” on page 61.</td>
</tr>
<tr>
<td></td>
<td>• Creating a new index with a DECFLOAT column as a key, or adding a DECFLOAT column to an existing index.</td>
</tr>
<tr>
<td></td>
<td>• Defining a primary or unique constraint with a DECFLOAT column.</td>
</tr>
</tbody>
</table>

Activation details for function level 504

To learn how to make new capabilities available for use in your Db2 environment, see Part 4, “Adopting new capabilities in Db2 12 continuous delivery,” on page 167.

Table 26. Function level 504 activation details

<table>
<thead>
<tr>
<th>Enabling APAR:</th>
<th>APAR PH07672.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum Db2 catalog level:</td>
<td>V12R1M503</td>
</tr>
</tbody>
</table>
Table 26. Function level 504 activation details (continued)

Application compatibility control: Applications must run at application compatibility level V12R1M504 or higher to use the following new capabilities:

- All new SQL syntax described in “New SQL syntax alternatives for special registers and NULL predicates” on page 67.
- All built-in functions described in “Newly supported built-in functions with IBM Db2 Analytics Accelerator” on page 67.
- Preventing the creation of deprecated object types, as described in “Prevent creation of new deprecated objects” on page 65.

Activation details for function level 503

To learn how to make new capabilities available for use in your Db2 environment, see Part 4, “Adopting new capabilities in Db2 12 continuous delivery,” on page 167.

Table 27. Function level 503 activation details

<table>
<thead>
<tr>
<th>Enabling APAR:</th>
<th>PH00506</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum Db2 catalog level:</td>
<td>V12R1M503</td>
</tr>
<tr>
<td>Catalog level V12R1M503 changes:</td>
<td>The REPLICATION OVERRIDE global variable is added. For a summary of changes in all Db2 12 catalog levels, see Chapter 17, “Catalog changes in Db2 12,” on page 127.</td>
</tr>
<tr>
<td>Application compatibility control:</td>
<td>Applications must run at application compatibility level V12R1M503 or higher to use the following new capabilities:</td>
</tr>
<tr>
<td></td>
<td>• The SYSIBMADM.REPLICATION OVERRIDE global variable for replication of system-period temporal tables and generated expression columns.</td>
</tr>
<tr>
<td></td>
<td>• The temporal query result change for system-period temporal tables defined with the ON DELETE ADD EXTRA ROW attribute that also contain a DATA CHANGE OPERATION column.</td>
</tr>
</tbody>
</table>

Activation details for function level 502

Table 28. Function level 502 activation details

<table>
<thead>
<tr>
<th>Enabling APAR:</th>
<th>PI95511</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum Db2 catalog level:</td>
<td>V12R1M502</td>
</tr>
</tbody>
</table>
Table 28. Function level 502 activation details (continued)

<table>
<thead>
<tr>
<th>Catalog level V12R1M502 changes:</th>
</tr>
</thead>
<tbody>
<tr>
<td>The KEYLABEL column is added in the following catalog tables:</td>
</tr>
<tr>
<td>• SYSIBM.SYSINDEXES</td>
</tr>
<tr>
<td>• SYSIBM.SYSSTOGROUP</td>
</tr>
<tr>
<td>• SYSIBM.SYSTABLESPACE</td>
</tr>
<tr>
<td>• SYSIBM.SYSTABLES</td>
</tr>
</tbody>
</table>

**Restriction:** Before you can run CATMAINT to tailor the Db2 catalog for function level 502 or higher, you must first activate function level 500 or 501. That is, the DISPLAY GROUP command output must indicate HIGHEST ACTIVATED FUNCTION LEVEL (V12R1M500) or higher. This restriction prevents tailoring the Db2 catalog for function levels higher than 500 while fallback to Db2 11 remains possible. However, later activating a lower function level such as function level 100* does not restrict the CATMAINT operation.

For a summary of changes in all Db2 12 catalog levels, see Chapter 17, “Catalog changes in Db2 12,” on page 127.

Application compatibility control:

Applications must run at application compatibility level V12R1M502 or higher to use the following new capabilities:

• CREATE or ALTER statements that specify the KEYLABEL option. However, applications that already reference encrypted data can continue to run at their current application compatibility levels.

• Explicit casting of numeric data types to fixed or variable length graphic strings, with the GRAPHIC or VARGRAPHIC built-in functions or the CAST specification.

Function level 501 activation details

Table 29. Activation details for function level 501.

<table>
<thead>
<tr>
<th>Enabling APAR:</th>
<th>PI70535</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum required catalog level:</td>
<td>V12R1M500</td>
</tr>
</tbody>
</table>

Application compatibility control:

Use of the LISTAGG built-in function level requires application compatibility level V12R1M501 or higher.

IBM data server clients and drivers that use the capabilities in function level 501 or are bound with application compatibility level V12R1M501 or higher, require extra program preparation steps. For instructions, see V12R1Mnnn application compatibility levels for data server clients and drivers (Db2 Application programming and SQL).

Related tasks

Adopting new capabilities in Db2 12 continuous delivery

In Db2 12, function levels and application compatibility levels control the adoption of most new capabilities by Db2 subsystems and Db2 applications.
Chapter 23. Function that Db2 12 no longer supports

If you are migrating to Db2 12 from Db2 11, be aware of the function that Db2 12 no longer supports.
The following functions are not supported in Db2 12.

Creating non-UTS table spaces

(FL 504) Non-UTS table spaces for base tables are deprecated. In general, for packages bound with
APPLCOMPAT(V12R1M504) or higher, the result of a CREATE TABLESPACE statement is always a
partition-by-growth or partition-by-range UTS table space, and a CREATE TABLE statement that
specifies a non-UTS table space, including existing multi-table segmented table spaces, returns an error. The only
exception is a dynamic CREATE TABLESPACE statement that is executed after the CURRENT
APPLICATION COMPATIBILITY special register is set to a value lower than V12R1M504. In this case,
CREATE TABLESPACE creates a non-UTS table space, and a CREATE TABLE statement can specify a non-
UTS table space. Existing tables in non-UTS table spaces remain supported. However, support is likely to
be removed in the future.

Creating hash-organized tables

(FL 504) Hash-organized tables are deprecated. Beginning in Db2 12, packages that are bound with
APPLCOMPAT(V12R1M504) or higher cannot create hash-organized tables or alter existing tables to use
hash-organization. Existing hash-organized tables remain supported, but they are likely to be
unsupported in the future.

CREATE SYNONYM statements

(FL 504) In V12R1M504 application compatibility and higher, Db2 no longer supports CREATE SYNONYM
statements. Create aliases with CREATE ALIAS statements instead of synonyms.

The DSNACCOR stored procedure

(FL 504) The DSNACCOR stored procedure is no longer created or configured during the Db2 installation
and migration process. Use the DSNACCOX stored procedure instead. See DSNACCOX stored procedure
(Db2 SQL).

Subsystem parameters that are removed in Db2 12

The following table lists the subsystem parameters that are removed and the resulting behavior in Db2
12. For a summary of all subsystem parameter changes in Db2 12 including new and changed subsystem
parameters, see Chapter 20, “Subsystem parameter changes in Db2 12,” on page 145.

<table>
<thead>
<tr>
<th>Subsystem parameter</th>
<th>Setting used in Db2 12</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALCUNIT</td>
<td>CYL, which means that primary and secondary space allocations are always obtained in cylinder units.</td>
</tr>
<tr>
<td>CACHE_DEP_TRACK_STOR_LIM</td>
<td>None, because the storage is allocated from the SCB pool.</td>
</tr>
<tr>
<td>CACHEDYN_FREELOCAL</td>
<td>0, which means that Db2 does not free cached dynamic statements to relieve high use of storage by dynamic SQL caching.</td>
</tr>
<tr>
<td>CATALOG (in DSN6ARVP)</td>
<td>YES, which means that Db2 requires all archive log data sets to be cataloged, regardless of whether they are allocated on disk or tape.</td>
</tr>
<tr>
<td>CHECKSETCHKP</td>
<td>NO, which means that he CHECK DATA and CHECK LOB utilities do not place objects in CHECK PENDING status if an inconsistency is detected.</td>
</tr>
</tbody>
</table>
### Table 30. Removed subsystem parameters in Db2 12 (continued)

<table>
<thead>
<tr>
<th>Subsystem parameter</th>
<th>Setting used in Db2 12</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONTSTOR</td>
<td>NO, which means that Db2 does not periodically contract the working storage area for each thread.</td>
</tr>
<tr>
<td>DB2SORT</td>
<td>ENABLE, which means that Db2 utilities use the Db2 Sort for z/OS product when it is installed. If Db2 Sort for z/OS is not installed, DFSORT is used instead.</td>
</tr>
<tr>
<td>EDMPOOL</td>
<td>None, because all EDM storage is above the bar.</td>
</tr>
<tr>
<td>INDEX_IO_PARALLELISM</td>
<td>YES, which means that I/O parallelism is enabled.</td>
</tr>
<tr>
<td>LEMAX</td>
<td>None. Db2 does not require an LE token limit for any package that was bound in DB2 10 or later. Db2 12 supports only packages that are bound in DB2 10 or later.</td>
</tr>
<tr>
<td>LOBVALA</td>
<td>Db2 automatically determines the amount of storage needed for processing LOB values.</td>
</tr>
<tr>
<td>LOBVALS</td>
<td>Db2 automatically determines the amount of storage needed for processing LOB values.</td>
</tr>
<tr>
<td>MINSTOR</td>
<td>NO, which means that Db2 does not use storage management algorithms that minimize the amount of working storage that is consumed by individual threads</td>
</tr>
<tr>
<td>NEWFUN dsnhdecp parameter</td>
<td>None, because the Db2 precompiler and coprocessor no longer use the NEWFUN dsnhdecp parameter.</td>
</tr>
<tr>
<td>REORG_IGNORE_FREESPACE</td>
<td>NO, which means that the REORG TABLESPACE utility uses the specified PCTFREE and FREEPAGE values.</td>
</tr>
<tr>
<td>RRF</td>
<td>ENABLE, which means that newly created table spaces or newly added partitions on partition-by-growth table spaces are created in the reordered row format. Existing basic row format table spaces are converted to reordered row format by the LOAD REPLACE or REORG TABLESPACE utilities.</td>
</tr>
<tr>
<td>SQWIDSC</td>
<td>YES, which means that the database manager invalidates but does not quiesce SQL statements in the dynamic statement cache that depend on an object that is the target of a RENAME TABLE or DROP ALIAS statement.</td>
</tr>
<tr>
<td>UTSORTAL</td>
<td>YES, which means that the utility attempts to allocate the sort work data sets. To increase the possible degree of parallelism, the utility minimizes the number of sort work data sets that are used. If the SORTNUM clause is specified in the utility control statement and the IGNSORTN subsystem parameter is set to NO, the sort program is used to allocate the sort work data sets. The utility also does not allocate the work data sets if they are allocated in the JCL.</td>
</tr>
<tr>
<td>XMLVALA</td>
<td>Db2 automatically determines the amount of storage for processing XML values.</td>
</tr>
<tr>
<td>XMLVALS</td>
<td>Db2 automatically determines the amount of storage for processing XML values.</td>
</tr>
</tbody>
</table>

**Query I/O parallelism**

Query I/O parallelism is removed in Db2 12. Applications that used query I/O parallelism in Db2 11 are downgraded to use sequential access at run time. You can rebind the packages to enable Db2 to consider the use of CP parallelism instead in Db2 12.
Resource limit table formats
DSNRLSTxx table formats and related index formats earlier than the DB2 Version 8 format are not supported in Db2 12 or later releases. When tables with unsupported formats are detected, Db2 issues message DSNT731I. The START RLIMIT command fails with message DSN9023I if no DSNRLMTxx table exists with the specified ID.

Trace types
IFCID 0366 is no longer supported. Run a trace for IFCID 0376 to track information about SQL statements that have potential incompatible changes when you switch to a new application behavior.

Deprecated function in Db2 12
Certain capabilities that Db2 12 for z/OS supports are deprecated, meaning that their use is discouraged. Although they remain supported except as noted below in Db2 12, 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 a detailed list of deprecated function in Db2 12, see Chapter 24, “ Deprecated function in Db2 12,” on page 161.
Chapter 24. Deprecated function in Db2 12

Certain capabilities that Db2 12 for z/OS supports are deprecated, meaning that their use is discouraged. Although they remain supported except as noted below in Db2 12, 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 31. Deprecated functions in Db2 12

<table>
<thead>
<tr>
<th>Deprecated function</th>
<th>Recommended alternative</th>
<th>Support removed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic row format table spaces</td>
<td>Use reordered row format.</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td>Starting in Db2 12, any table space that uses basic row format is automatically converted to reordered row format when you run the LOAD REPLACE or REORG TABLESPACE utilities.</td>
<td></td>
</tr>
<tr>
<td>BIND PLAN command MEMBER option</td>
<td>Use BIND PACKAGE commands to bind DBRMs into packages explicitly.</td>
<td>—</td>
</tr>
<tr>
<td>COPY utility CHANGELIMIT option</td>
<td>Use the DSNACCOX stored procedure to determine if the object needs to be copied.</td>
<td>—</td>
</tr>
<tr>
<td>DISALLOW_SEL_INTO_UNION subsystem parameter.</td>
<td>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.</td>
<td>—</td>
</tr>
<tr>
<td>DSNRLMT and DSNRLST in older formats</td>
<td>Use the latest Db2 12 format resource limit tables.</td>
<td>—</td>
</tr>
<tr>
<td>SYSPROC.DSNTBIND stored procedure</td>
<td>Use the SYSPROC.ADMIN_COMMAND_DSN stored procedure.</td>
<td>—</td>
</tr>
<tr>
<td>SYSPROC.DSNUTILS stored procedure</td>
<td>Use the SYSPROC.DSNUTILU stored procedure.</td>
<td>—</td>
</tr>
<tr>
<td>SYSPROC.DSNWZP</td>
<td>Use the SYSPROC.ADMIN_INFO_SYSPARM stored procedure.</td>
<td>—</td>
</tr>
<tr>
<td>Hash-organized tables</td>
<td>Alter tables to drop hash organization, and, create indexes to support fast index traversal in Db2 12. For more information, see Fast index traversal (Db2 Performance).</td>
<td>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.</td>
</tr>
</tbody>
</table>


<table>
<thead>
<tr>
<th>Deprecated function</th>
<th>Recommended alternative</th>
<th>Support removed</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOAD utility IDENTITYOVERRIDE option</td>
<td>Use the OVERRIDE(IDENTITY) option.</td>
<td>—</td>
</tr>
<tr>
<td>LOAD utility PERIODOVERRIDE option</td>
<td>Use the OVERRIDE(SYSTEMPERIOD) option.</td>
<td>—</td>
</tr>
<tr>
<td>LOAD utility TRANSIDOVERRIDE option</td>
<td>Use the OVERRIDE(TRANSID) option.</td>
<td>—</td>
</tr>
<tr>
<td>MATERIALIZE_NODET_SQLTUDF subsystem parameter</td>
<td>Set MATERIALIZE_NODET_SQLTUDF to YES. Later Db2 releases always use the behavior of this setting.</td>
<td>—</td>
</tr>
<tr>
<td>NEWFUN SQL processing option</td>
<td>Use SQLLEVEL. NEWFUN is ignored if SQLLEVEL is specified.</td>
<td>—</td>
</tr>
<tr>
<td>ODBC 2.0 function</td>
<td>See Deprecated ODBC functions (Db2 Programming for ODBC).</td>
<td>—</td>
</tr>
<tr>
<td>REORG INDEX utility LAEFDISTLIMIT and REPORT only options</td>
<td>Use the DSNACCOX stored procedure to determine whether the object needs to be reorganized.</td>
<td>—</td>
</tr>
<tr>
<td>REORG INDEX utility UNLOAD ONLY option</td>
<td>Use the UNLOAD utility.</td>
<td>—</td>
</tr>
<tr>
<td>REORG INDEX utility UNLOAD PAUSE option</td>
<td>Use the DIAGNOSE utility to stop the process.</td>
<td>—</td>
</tr>
<tr>
<td>REORG TABLESPACE utility UNLOAD EXTERNAL option</td>
<td>Use the UNLOAD utility.</td>
<td>—</td>
</tr>
<tr>
<td>REORG TABLESPACE utility INDREFLIMIT and REPORTONLY options</td>
<td>Use the DSNACCOX stored procedure to determine whether the object needs to be reorganized.</td>
<td>—</td>
</tr>
<tr>
<td>REORG TABLESPACE utility OFFPOSLIMIT and REPORTONLY options</td>
<td>Use the DSNACCOX stored procedure to determine whether the object needs to be reorganized.</td>
<td>—</td>
</tr>
<tr>
<td>REORG TABLESPACE utility PARALLEL option</td>
<td>Use the LISTPARTS option instead.</td>
<td>—</td>
</tr>
<tr>
<td>REORG TABLESPACE utility UNLOAD ONLY option</td>
<td>Use the UNLOAD utility.</td>
<td>—</td>
</tr>
<tr>
<td>REORG TABLESPACE utility UNLOAD PAUSE option</td>
<td>Use the UNLOAD utility FORMAT INTERNAL option.</td>
<td>—</td>
</tr>
<tr>
<td>REPAIR VERSIONS utility</td>
<td>Use the REPAIR CATALOG utility.</td>
<td>—</td>
</tr>
<tr>
<td>DB2XML.SOAPHTTPC supplied user-defined function</td>
<td>Use the DB2XML.SOAPHTTPNC user-defined function.</td>
<td>—</td>
</tr>
<tr>
<td>DB2XML.SOAPHTTPV supplied user-defined function</td>
<td>Use the DB2XML.SOAPHTTPNV user-defined function.</td>
<td>—</td>
</tr>
<tr>
<td>Deprecated function</td>
<td>Recommended alternative</td>
<td>Support removed</td>
</tr>
<tr>
<td>---------------------</td>
<td>-------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>Synonyms</td>
<td>Use aliases when writing new SQL statements or creating portable applications. Aliases behave the same for the Db2 family of products.</td>
<td>—</td>
</tr>
<tr>
<td>Non-UTS table spaces for base tables, including segmented (non-UTS), partitioned (non-UTS), and simple table spaces.</td>
<td>Use partition-by-growth or partition-by-range universal table spaces instead. (FL 504) In Db2 12, Packages bound with APPLCOPMAT(V12R1M504) or higher cannot create objects of the following types:   • 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 You cannot create new simple table spaces in any supported Db2 release.</td>
<td>—</td>
</tr>
<tr>
<td>TEMP database</td>
<td>Db2 uses the work file database instead.</td>
<td>—</td>
</tr>
</tbody>
</table>

**Related tasks**
Preparing your system to install or migrate to Db2 12 (Db2 Installation and Migration)
Db2 12 function levels introduce changes to the operational requisites for Db2 12 for z/OS. *Operational requisites* are products that are required and must be present on the system or products that are not required but should be present on the system for Db2 to operate all or part of its functions.

Db2 function levels up to and including function level 505 introduce the following changes to the operational requisites for Db2 that are described in the Program Directory for Db2 12 for z/OS: No changes at this time.
Part 4. Adopting new capabilities in Db2 12 continuous delivery

In Db2 12, 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 \texttt{VvvRrMmmm}, where \texttt{vv} is the version, \texttt{r} is the release, and \texttt{mmm} is the modification level. For example, \texttt{V12R1M505} identifies function level 505. For a list of all available function levels in Db2 12, see Part 2, “What's new in Db2 12 function levels,” on page 59. Often function level identifiers are abbreviated. For example, "function level 505" refers to \texttt{V12R1M505}.

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 \texttt{DISPLAY GROUP} commands. For more information and examples, see Chapter 27, “Determining the Db2 code level, catalog level, and function level,” on page 175.

Overview of Db2 12 new function availability

The availability of new function depends on the type of enhancement, the activated function level, and the application compatibility levels of applications. In the initial Db2 12 release, most new capabilities are enabled only after the activation of function level 500 or higher.

Virtual storage enhancements

Virtual storage enhancements become available at the activation of the function level that introduces them or higher. Activation of function level 100 introduces all virtual storage enhancements in the initial Db2 12 release. That is, activation of function level 500 introduces no virtual storage enhancements.

Subsystem parameters

New subsystem parameter settings are in effect only when the function level that introduced them or a higher function level is activated. Most subsystem parameter changes in the initial Db2 12 release take effect in function level 500. For a list of these changes, see Chapter 20, “Subsystem parameter changes in Db2 12,” on page 145.

Optimization enhancements

Optimization enhancements become available after the activation of the function level that introduces them or higher, and full prepare of the SQL statements. When 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

Activation of function level 100 introduces all optimization enhancements in the initial Db2 12 release. That is, function level 500 introduces no optimization enhancements.

SQL capabilities

New SQL capabilities become available after the activation of the function level that introduces them or higher, for applications that run at the equivalent application compatibility level or higher. New SQL capabilities in the initial Db2 12 release become available in function level 500 for applications that run at the equivalent application compatibility level or higher. You can continue to run SQL statements compatibly with lower function levels, or previous Db2 releases, including Db2 11 and DB2 10. For details, see Application compatibility levels in Db2 (Db2 Application programming and SQL).
Procedure

To manage the adoption of new capabilities in Db2 12, 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 in advance, before tailoring the catalog level or activating 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.

   When you update the Db2 catalog to a higher catalog level, the single CATMAINT job applies the updates for the target catalog level and all lower catalog levels above the current catalog level. 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 required 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 12 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 (Db2 Application programming and SQL).

Related concepts

What’s new in the Db2 12 base release
Db2 12 for z/OS takes Db2 to a new level, both extending the core capabilities and empowering the future. Db2 12 extends the core with new enhancements to scalability, reliability, efficiency, security, and availability. Db2 12 also empowers the next wave of applications in the cloud, mobile, and analytics spaces.

Application compatibility levels in Db2 (Db2 Application programming and SQL)

Related tasks

Managing application incompatibilities (Db2 Application programming and SQL)

Related reference

What’s new in Db2 12 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

Exploring IBM Db2 for z/OSContinuous Delivery (IBM Redpaper)
Db2 12 for z/OS Technical Overview (IBM Redbooks)
Video: Db2 for z/OS—Delivering New Capabilities Faster (YouTube: 1:33:25)
PI70406: Add ACTIVATE mode to the Db2 installation CLIST
Chapter 26. Function levels and related levels in Db2 12

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 and examples, see Chapter 27, “Determining the Db2 code level, catalog level, and function level,” on page 175.

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 vvv is the version, r is the release, and mmmm is the modification level. For example, V12R1M505 identifies function level 505. For a list of all available function levels in Db2 12, see Part 2, “What's new in Db2 12 function levels,” on page 59.

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 in advance, before tailoring the catalog level or activating 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 vvv is the version, r is the release, and mmmm 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 mmm 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.

When you update the Db2 catalog to a higher catalog level, the single CATMAINT job applies the updates for the target catalog level and all lower catalog levels above the current catalog level.

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 required 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 VvRrMmmm, where vv is the version, r is the release, and mmm is the modification level. For example, function level 505 requires catalog level V12R1M505.

An exception is function level 100, which requires catalog level V12R1M500. However, catalog level V12R1M500 is the result when you tailor the catalog for migration to Db2 12, as described in Installation step 16: Tailor the Db2 catalog: DSNTIJTC (Db2 Installation and Migration).

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.

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.

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

Activation of a function level implies activation of the capabilities that are introduced by all lower function levels.

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.

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
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 V12R1Mnnn 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 (Db2 Application programming and SQL).

Application compatibility levels for Db2 12 are identified by the same identifier as the corresponding function level. The format is VvvRrMmm, where vv is the version, r is the release, and mmm is the modification level.

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. In some ways function level 100* in Db2 12 is analogous to the CM* and ENFM* migration modes in previous releases. 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. V12R1M500* 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**

What's new in Db2 12 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)

Video: Db2 for z/OS—Delivering New Capabilities Faster (YouTube: 1:33:25)
Chapter 27. 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

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 example illustrates DISPLAY GROUP output in a data sharing group with two members migrated to Db2 12 and ready for the activation of function level 500. However, the code level of member DB2B indicates that it is not migrated to Db2 12 and not ready for the activation of function level 500. The DB2 LVL value is 111500, which indicates the code level of Db2 11 in new-function mode.

```
-DISPLAY GROUP DETAIL
DSN7100I -DB2C DSN7GCMD
*** BEGIN DISPLAY OF GROUP(DSNCAT ) CATALOG LEVEL(V12R1M500)
  CURRENT FUNCTION LEVEL(V12R1M100)
  HIGHEST ACTIVATED FUNCTION LEVEL(V12R1M100)
  HIGHEST POSSIBLE FUNCTION LEVEL(V12R1M100)
  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   121500 MVSA      DJ2A   DB2AIRLM
  DB2B       2 DB2B -DB2B     ACTIVE   111500 MVSB      DJ2B   DB2BIRLM
  DB2C       3 DB2C -DB2C     ACTIVE   121500 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
```

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

```
-DISPLAY GROUP DETAIL
DSN7100I -DB2B DSN7GCMD
*** BEGIN DISPLAY OF GROUP(DSNCAT ) CATALOG LEVEL(V12R1M500)
  CURRENT FUNCTION LEVEL(V12R1M100)
  HIGHEST ACTIVATED FUNCTION LEVEL(V12R1M100)
  HIGHEST POSSIBLE FUNCTION LEVEL(V12R1M500)

DSN9022I -DB2C DSN7GCMD 'DISPLAY GROUP ' NORMAL COMPLETION
```
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 29, “Activating Db2 12 function levels,” on page 179.

Related concepts
Function levels and related levels in Db2 12
Enhancements to Db2 are enabled for use when you activate function levels.

Related reference
- DISPLAY GROUP (Db2) (Db2 Commands)

Related information
DSN7100I (Db2 Messages)
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 and examples, see Chapter 27, “Determining the Db2 code level, catalog level, and function level,” on page 175.

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 (V12R1M500) 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 (V12R1M500)
   GROUP ELIGIBLE FOR FUNCTION LEVEL (V12R1M500)
   CATALOG LEVEL(V12R1M500)
   CURRENT FUNCTION LEVEL(V12R1M100)
   PREVIOUS HIGHEST FUNCTION LEVEL (V12R1M100)
   HIGHEST POSSIBLE FUNCTION LEVEL (V12R1M500)
   ```

   ```
   DB2 MEMBER ID CODE-LEVEL LOWEST HIGHEST STATUS
   -------- --- ---------- ---------- ---------- -------------------
   DB2A 1 V12R1M500 V12R1M100 V12R1M500 ELIGIBLE
   DB2B 2 V12R1M500 V12R1M100 V12R1M500 ELIGIBLE
   DB2C 3 V12R1M500 V12R1M100 V12R1M500 ELIGIBLE
   ```

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

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 29, “Activating Db2 12 function levels,” on page 179.

Related concepts

Function levels and related levels in Db2 12
Enhancements to Db2 are enabled for use when you activate function levels.

**Related reference**
- ACTIVATE (Db2) (Db2 Commands)

**Related information**
- DSNU757I (Db2 Messages)
Chapter 29. Activating Db2 12 function levels

You control the activation and adoption of new features in Db2 12 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

Restriction: Before you can run CATMAINT to tailor the Db2 catalog for function level 502 or higher, you must first activate function level 500 or 501. That is, the DISPLAY GROUP command output must indicate HIGHEST ACTIVATED FUNCTION LEVEL (V12R1M500) or higher. This restriction prevents tailoring the Db2 catalog for function levels higher than 500 while fallback to Db2 11 remains possible. However, later activating a lower function level such as function level 100* does not restrict the CATMAINT operation.

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:

- “Incompatible changes summary for function levels 501 and higher” on page 87
- Part 2, “What’s new in Db2 12 function levels,” on page 59
- Part 3, “What’s changed in Db2 12,” on page 85

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 Activating Db2 function levels by using z/OSMF.

Overview of Db2 12 new function availability

The availability of new function depends on the type of enhancement, the activated function level, and the application compatibility levels of applications. In the initial Db2 12 release, most new capabilities are enabled only after the activation of function level 500 or higher.

Virtual storage enhancements

Virtual storage enhancements become available at the activation of the function level that introduces them or higher. Activation of function level 100 introduces all virtual storage enhancements in the initial Db2 12 release. That is, activation of function level 500 introduces no virtual storage enhancements.

Subsystem parameters

New subsystem parameter settings are in effect only when the function level that introduced them or a higher function level is activated. Most subsystem parameter changes in the initial Db2 12 release take effect in function level 500. For a list of these changes, see Chapter 20, “Subsystem parameter changes in Db2 12,” on page 145.

Optimization enhancements

Optimization enhancements become available after the activation of the function level that introduces them or higher, and full prepare of the SQL statements. When 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

Activation of function level 100 introduces all optimization enhancements in the initial Db2 12 release. That is, function level 500 introduces no optimization enhancements.

**SQL capabilities**
New SQL capabilities become available after the activation of the function level that introduces them or higher, for applications that run at the equivalent application compatibility level or higher. New SQL capabilities in the initial Db2 12 release become available in function level 500 for applications that run at the equivalent application compatibility level or higher. You can continue to run SQL statements compatibly with lower function levels, or previous Db2 releases, including Db2 11 and Db2 10. For details, see Application compatibility levels in Db2 (Db2 Application programming and SQL)

**Procedure**
To activate capabilities and enhancements that are introduced by Db2 function levels, 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 27, “Determining the Db2 code level, catalog level, and function level,” on page 175.

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 12 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 12, or most recently activated a Db2 12 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 *v* 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.

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

5. Run the DSNTIJTC job to run the CATMAINT utility to tailor the Db2 catalog for the target function level.
When you update the Db2 catalog to a higher catalog level, the single CATMAINT job applies the updates for the target catalog level and all lower catalog levels above the current 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 required code level. The code to tolerate catalog changes is contained in the code level that delivers the CATMAINT job.

**Restriction:** Before you can run CATMAINT to tailor the Db2 catalog for function level 502 or higher, you must first activate function level 500 or 501. That is, the DISPLAY GROUP command output must indicate HIGHEST ACTIVATED FUNCTION LEVEL (V12R1M500) or higher. This restriction prevents tailoring the Db2 catalog for function levels higher than 500 while fallback to Db2 11 remains possible. However, later activating a lower function level such as function level 100* does not restrict the CATMAINT operation.

6. Optional: Test the Db2 code level and catalog level for readiness to activate of the function level. For more information and examples, see Chapter 28, “Testing Db2 function level activation,” on page 177.

7. 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.

```
DSNU757I  -DB2A DSNUGCCA
*** BEGIN ACTIVATE FUNCTION LEVEL (V12R1M502)
   FUNCTION LEVEL (V12R1M502) SUCCESSFULLY ACTIVATED
   CATALOG LEVEL(V12R1M502)
   CURRENT FUNCTION LEVEL(V12R1M500)
   PREVIOUS HIGHEST FUNCTION LEVEL (V12R1M500)
   HIGHEST POSSIBLE FUNCTION LEVEL(V12R1M502)
```

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

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.

**Tip:** Proceed with the following steps only if all Db2 applications can be bound and run at the target function level. For more information, see Chapter 32, “Controlling Db2 application compatibility,” on page 187.

8. Optional: Run the DSNTIJUZ job to modify the subsystem parameter module with the APPLCOMPAT value that was specified on panel DSNTP00.

9. Optional: Run the DSNTIJQZ job to run a SET SYSPARM command to bring the APPLCOMPAT subsystem parameter change online.

10. Optional: Run the DSNTIJUA job to modify the Db2 data-only application defaults module with the SQLLEVEL value that was specified on panel DSNTP00.

**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. When 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 32, “Controlling Db2 application compatibility,” on page 187.

• If you encounter regressions or other problems when you activate a Db2 12 function level, minimize the impact to your applications while you resolve the problems by following the general approaches described in Chapter 33, “Responding to problems after function level activation,” on page 189.

Related concepts
Function levels and related levels in Db2 12
Enhancements to Db2 are enabled for use when you activate function levels.

Application compatibility levels in Db2 (Db2 Application programming and SQL)

Related reference
What’s new in Db2 12 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)
Chapter 30. 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:** If you are activating function level 500 or higher for the first time, do not start this task before you read Migration step 29: Activate function level 500 (Db2 Installation and Migration).

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:

- “Incompatible changes summary for function levels 501 and higher” on page 87
- Part 2, “What’s new in Db2 12 function levels,” on page 59
- Part 3, “What’s changed in Db2 12,” on page 85

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 27, “Determining the Db2 code level, catalog level, and function level,” on page 175.

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 12 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 DSNITP00, specify the target function level in the **TARGET FUNCTION LEVEL** field.
   
   The format is **VvRrMmmm**, where **v** 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)
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 12 and activating Db2 function levels use this job.

Related tasks
- Activating Db2 function levels by using z/OSMF
- Automating Db2 migration by using z/OS Management Facility (Db2 Installation and Migration)

Related reference
- SET SYSPARM (Db2) (Db2 Commands)
Chapter 31. 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 (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 HDBCC10, 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, DSNSTSPRM, DSNDIR1, 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 HDBCC10, 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 he 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 32. 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 12 migration process, and the activation of Db2 12 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 4, “Adopting new capabilities in Db2 12 continuous delivery,” on page 167.

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 12 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 12), continue to run your applications at the same application compatibility level until you are satisfied that your Db2 12 environment is stable at the new function level.
2. Rebind packages such as SPUFI and DSNTEP2 to enable database administrators to begin using the new SQL data definition capabilities.
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 (Db2 Application programming and SQL).
Related concepts
Application compatibility levels in Db2 (Db2 Application programming and SQL)
Function levels and related levels in Db2 12
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 12 function level, you can take certain actions to minimize the impact to your applications while you resolve the underlying problems.

V12R1Mnnn application compatibility levels for data server clients and drivers (Db2 Application programming and SQL)
Chapter 33. Responding to problems after function level activation

If you encounter regression or other problems after the activation of a new Db2 12 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. In some ways function level 100* in Db2 12 is analogous to the CM* and ENFM* migration modes in previous releases. 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. V12R1M500* 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 required 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 (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 12 migration process, and the activation of Db2 12 function levels, from your application updates for adoption of new function and resolution of incompatibilities.
Chapter 34. 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.

New content and changes for a function levels for function levels are generally marked with revision marks and usually include a function level indicator (for example: (FL 501)), which is also a link to the description of the function level that introduced the change. If an entire topic is new, a single function level indicator for the function level that introduced the topic precedes the body of the content, after the first paragraph.

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 the revision marks and function level indicators. 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.

Revision marks without associated function level indicators might indicate other types of changes, including:

- Technical revisions that reflect changed Db2 externals that are introduced by the Db2 12 release, by a new Db2 12 function level, or by maintenance after the general availability of Db2 12.
- Technical clarifications for unchanged Db2 externals in response to customer and internal feedback.

Related tasks
Adopting new capabilities in Db2 12 continuous delivery
In Db2 12, function levels and application compatibility levels control the adoption of most new capabilities by Db2 subsystems and Db2 applications.

Related reference
What's new in Db2 12 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 12 for z/OS and related products

Information about Db2 12 for z/OS and products that you might use in conjunction with Db2 12 is available online in IBM Knowledge Center or on library websites.

Obtaining Db2 for z/OS publications

Current Db2 12 for z/OS publications are available from the following websites:


Links to IBM Knowledge Center and the PDF version of each publication are provided.

Db2 for z/OS publications are also available for download from the IBM Publications Center (http://www.ibm.com/shop/publications/order).

In addition, books for Db2 for z/OS are available on a CD-ROM that is included with your product shipment:

- Db2 12 for z/OS Licensed Library Collection, LKxT-xxxx, in English. The CD-ROM contains the collection of books for Db2 12 for z/OS in PDF format. Periodically, IBM refreshes the books on subsequent editions of this CD-ROM.

Installable information center

You can download or order an installable version of the Information Management Software for z/OS Solutions Information Center, which includes information about Db2 12 for z/OS, QMF, IMS, and many Db2 Tools for z/OS products. You can install this information center on a local system or on an intranet server. For more information, see http://www-01.ibm.com/support/knowledgecenter/SSEPEK_11.0.0/com.ibm.db2z11.doc/src/alltoc/installabledzic.html.
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 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:

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 12 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 12 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 12 for z/OS.

General-use Programming Interface and Associated Guidance Information is identified where it occurs by the following markings:

`GUPI` 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:

`PSPI` Product-sensitive Programming Interface and Associated Guidance Information...

**Trademarks**

IBM, the IBM logo, and 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, NONINFRINGEMENT, 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.

The glossary is available in IBM Knowledge Center.
See the Glossary topic for definitions of Db2 for z/OS terms.
## Index

### Numerics

<table>
<thead>
<tr>
<th>10-byte format</th>
<th>extended format</th>
</tr>
</thead>
<tbody>
<tr>
<td>RBA and LRSN 149</td>
<td>RBA and LRSN 149</td>
</tr>
</tbody>
</table>

### A

<table>
<thead>
<tr>
<th>activation details summary</th>
<th>function levels 153</th>
</tr>
</thead>
<tbody>
<tr>
<td>APPLCOMPAT 97</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>application programs</th>
</tr>
</thead>
<tbody>
<tr>
<td>compatibility 97</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>application release incompatibilities 89</th>
</tr>
</thead>
<tbody>
<tr>
<td>arrays</td>
</tr>
<tr>
<td>functions 24</td>
</tr>
<tr>
<td>routines 24</td>
</tr>
<tr>
<td>stored procedures 24</td>
</tr>
<tr>
<td>AUTH_COMPATIBILITY</td>
</tr>
<tr>
<td>macro DSN6SPRM 40</td>
</tr>
</tbody>
</table>

### B

<table>
<thead>
<tr>
<th>BSDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>conversion 108</td>
</tr>
<tr>
<td>expanded format 108</td>
</tr>
<tr>
<td>buffer pools</td>
</tr>
<tr>
<td>in-memory 54</td>
</tr>
</tbody>
</table>

### C

<table>
<thead>
<tr>
<th>catalog levels</th>
</tr>
</thead>
<tbody>
<tr>
<td>determining 175</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>catalog tables</th>
</tr>
</thead>
<tbody>
<tr>
<td>history tables 40</td>
</tr>
<tr>
<td>temporal data 40</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>code levels</th>
</tr>
</thead>
<tbody>
<tr>
<td>determining 175</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>command release incompatibilities 103</th>
</tr>
</thead>
<tbody>
<tr>
<td>commands</td>
</tr>
<tr>
<td>changes 109</td>
</tr>
</tbody>
</table>

### D

<table>
<thead>
<tr>
<th>data validation</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSN1COPY 37</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Db2 commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>DISPLAY GROUP</td>
</tr>
<tr>
<td>examples 175</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Db2 enhancements after general availability 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Db2 function levels</td>
</tr>
<tr>
<td>determining 175</td>
</tr>
</tbody>
</table>

| DISPLAY GROUP                               |
| command examples 175                        |
| DRDA fast load 25, 28                       |
| DSN1COPY                                    |
| data validation 37                          |
| improvements 37                             |
| dynamic statement cache 54–57               |

### E

<table>
<thead>
<tr>
<th>enhancements by function level 59, 171, 179</th>
</tr>
</thead>
<tbody>
<tr>
<td>EXPLAIN tables</td>
</tr>
<tr>
<td>changes 137</td>
</tr>
<tr>
<td>Db2 12 format 137</td>
</tr>
<tr>
<td>encoding 137</td>
</tr>
</tbody>
</table>

### F

<table>
<thead>
<tr>
<th>function levels</th>
</tr>
</thead>
<tbody>
<tr>
<td>determining 175</td>
</tr>
<tr>
<td>summary of activation details 153</td>
</tr>
<tr>
<td>functions</td>
</tr>
<tr>
<td>array parameters 24</td>
</tr>
</tbody>
</table>

### G

<table>
<thead>
<tr>
<th>general-use programming information, described 196</th>
</tr>
</thead>
<tbody>
<tr>
<td>GUPI symbols 196</td>
</tr>
</tbody>
</table>

### H

<table>
<thead>
<tr>
<th>history tables</th>
</tr>
</thead>
<tbody>
<tr>
<td>Db2 catalog 40</td>
</tr>
</tbody>
</table>

### I

<table>
<thead>
<tr>
<th>IFCID (instrumentation facility component identifier)</th>
</tr>
</thead>
<tbody>
<tr>
<td>changed 141</td>
</tr>
<tr>
<td>new 141</td>
</tr>
</tbody>
</table>

| IFCID 0366 97                                      |
| IFCID 0376 97                                      |

| in-memory buffer pools 54                       |
| incompatibilities of releases                   |
| applications and SQL 89                        |
| commands 103                                     |
| security 105                                     |
| unsupported functions 157                       |
| utilities 100                                    |

| index compression                               |
| altering 35                                      |
insert algorithm 2 53
insert partition 39
installation
  installation SYSOPR 34
  SYSADM 34

J
JSON 25

K
KEEPDYNAMIC bind option
  ROLLBACK 91

L
links
  non-IBM Web sites 196
LRSN
  extended 149

M
macro DSN6SPRM
  AUTH_COMPATIBILITY 40
MERGE statement
  enhancements 25
migration
  considerations 87
  installation SYSOPR 34
  SYSADM 34

O
ODBC
  enhancements 28
ownership
  transferring
    TRANSFER OWNERSHIP statement 36

P
packages
  automatic rebinds 91
partition-by-range table spaces 39
pending definition changes
  column alterations 35
  recovery prior to materialization 35
percentile functions 28
performance enhancements
  buffer pools 54
plans
  automatic rebinds 91
point-in-time recovery
  prior to materialization of pending definition changes 35
POWER built-in function 93
product-sensitive programming information, described 196
programming interface information, described 196
PSPI symbols 196
pureXML
  multiple XML updates 24

R
RBA
  extended 149
real-time statistics improvements 45, 53
rebinds
  automatic 91
release incompatibilities
  applications and SQL 89
  commands 103
  security 105
  unsupported functions 157
utilities 100
removed functions 157
REPAIR
  CATALOG 37
resource limit facility
  static SQL statements 29
routines
  array parameters 24

S
security release incompatibilities 105
SQL PL
  in trigger 23
SQL release incompatibilities 89
SQL statements
  changes 113
  TRANSFER OWNERSHIP 36
SQL variables
  arrays 24
static SQL statements
  resource limit facility 29
stored procedures
  array parameters 24
Support for maintaining session data on the target server 29

T
temporal logical transactions 27
temporal referential constraint 27
trace field QW0366FN 97
trace field QW0376FN 97
TRANSFER OWNERSHIP statement
  database objects 36
  system objects 36
triggers
  SQL PL 23

U
unsupported functions 157
utilities
  changes 121
  release incompatibilities 100

X
XMLMODIFY
  multiple XML updates 24
Product Number: 5650-DB2
            5770-AF3