Db2 11.1 for Linux, UNIX, and Windows

IBM

Upgrading to Db2 Version 11.1

Db2 11.1 for Linux, UNIX, and Windows

Upgrading to Db2 Version 11.1

Notice regarding this document

This document in PDF form is provided as a courtesy to customers who have requested documentation in this format. It is provided As-Is without warranty or maintenance commitment.

Contents

Figures

Tables

Upgrade to Db2 Version 11.1

Upgrading your Db2® database product to a new release might require upgrading your Db2 environment components.

Your Db2 environment has several components such as Db2 servers, Db2 clients, database applications, routines, tools and, scripts. Upgrading these components requires an understanding of Db2 database products and their upgrade concepts.

The upgrade process includes the following tasks that you must perform to run your Db2 environment successfully on a new release:
- Planning your Db2 environment upgrade to a new release includes creating an upgrade strategy and an upgrade plan for each of the component in your Db2 environment. For more information, see “Plan your Db2 environment upgrade”
- Upgrading your existing instances and databases so that they run successfully in the new release. For more information, see “Upgrade Db2 servers” on page 9.
- Upgrading your client instances to keep the configuration of your existing clients. For more information, see “Upgrade Db2 clients” on page 122.
- Upgrading your database applications and routines that includes testing and modifying them to support changes in the new release. For more information, see “Upgrade database applications, routines, tools, and scripts” on page 137.

The upgrade task for each of the component in your Db2 environment includes the following information: :
- Upgrade prerequisites provide details about upgrade support, restrictions, and best practices that you must understand before upgrade.
- Pre-upgrade tasks describe all the preparation tasks that you must perform before upgrade.
- Upgrade tasks provide a step-by-step procedure to upgrade the Db2 environment components.
- Post-upgrade tasks describe all the tasks that you must perform after upgrade.

In the upgrade tasks, the term *pre-Db2 Version 11.1 releases* refers to Db2 Version 9.7, Db2Version 10.1, or Db2 Version 10.5.

Plan your Db2 environment upgrade

Your environment has several components such as Db2 servers, Db2 clients, database applications, routines, tools, and scripts. Planning your upgrade requires a thorough understanding of the upgrade process of each component in your environment.

First, devise a strategy on how to approach your environment upgrade. You must determine the order in which you are going to upgrade each component. The characteristics of your environment and the information in upgrade essentials, especially the best practices, and restrictions can help you determine your strategy.

The following is an example of a good *upgrade strategy* in which you test your database applications and routines and determine that they run successfully in Db2 Version 11.1.

1. Review the new, deprecated, and discontinued functionality for Db2 Version 11.1 and for any releases between the release you are upgrading from and Db2 Version 11.1.
2. Create a plan to modify your database applications and routines. Verify that they run successfully in Db2 Version 11.1.
3. Set up a Db2 Version 11.1 test server and create test databases.
4. Test your database applications and routines on a Db2 Version 11.1 test database to determine whether they run successfully. If your application requires a client, use a Db2 Version 11.1 client.
5. Upgrade your Db2 servers and Db2 clients in a test environment. Determine the issues and solutions. Use this information to adjust your upgrade plan.
6. Upgrade your Db2 servers to Db2 Version 11.1 in your production environment. Verify that the servers operate as expected.
7. Upgrade your Db2 clients to Db2 Version 11.1 in your production environment. Verify that the clients operate as expected.
8. Test your database applications and routines in the Db2 Version 11.1 upgraded production environment to determine whether they run as expected.
9. Make your upgraded environment available to users.

After you devise a strategy that gives an outline of your upgrade plan, you can define the upgrade plan details for each component in your environment. An *upgrade plan* must include the following details for each component:
- Upgrade prerequisites
- Pre-upgrade tasks
- Upgrade tasks
- Post-upgrade tasks

If you have previous upgrade plans, review and compare them with the upgrade plan for Db2 Version 11.1. Include in your new plan any steps that are related to internal procedures to request access, software installation, or other system services within your organization.

Review also the Db2 upgrade portal at www.ibm.com/support (formerly known as Db2 migration portal) that provides access to more resources and up-to-date information about the upgrade process as they become available. These resources include educational material, white papers, and webcast for upgrade.

Finally, plan to remove the use of deprecated functionality and incorporate new functionality from Db2 Version 11.1. Although you are only required to remove the use of discontinued functionality, you must also plan to remove the use of deprecated functionality after upgrade because they will become unsupported in a future release. Also, you must take advantage of new functionality for your database products, applications, and routines to enhance functions and improve performance.

Planning to upgrade Db2 servers

Planning the upgrade of Db2 servers requires that you review all of the applicable upgrade prerequisites, pre-upgrade tasks, upgrade tasks, and post-upgrade tasks.

About this task

This task helps you create an upgrade plan for Db2 servers in Db2 environments other than Db2 pureScale® environments.

Procedure

To create an upgrade plan for your Db2 servers:

1. Write the upgrade plan for Db2 servers, considering all of the details that apply to your environment:

Table 1. Upgrade plan details for Db2 servers.

Upgrade plan	Details
Prerequisites	Ensure that you: • Meet the installation requirements for Db2 database products that are described in *Installing Db2 Servers*. • Review the information in “Supported upgrade paths for Db2 servers” on page 11 • Meet all prerequisites for the upgrade task and subtasks, especially obtaining root or Local Administrator access and Db2 authorization. • Review the information in the “Upgrade essentials for Db2 servers” on page 12 topic. It includes the following: – “Db2 command actions to upgrade instances and databases” on page 12 – “Upgrade restrictions for Db2 servers” on page 14 – “Db2 server behavior changes” on page 17 – “Deprecated or discontinued functionality that affects Db2 server upgrades” on page 20 – “Disk space requirements for Db2 server upgrades” on page 21 – “Support changes for 32-bit and 64-bit Db2 servers” on page 22 – “Best practices for upgrading Db2 servers” on page 23 – “Migration from non-Db2 relational database management systems” on page 26
Pre-upgrade tasks	Review the list of tasks in the “Pre-upgrade tasks for Db2 servers” on page 27 topic. It includes the following: • “Verifying that your databases are ready for upgrade” on page 29 • “Backing up databases before or after upgrade” on page 32 • “Backing up Db2 server configuration and diagnostic information” on page 34 • “Increasing table space and log file sizes before upgrade” on page 36 • “Changing raw devices to block devices (Linux)” on page 38 • “Gathering pre-upgrade diagnostic information” on page 39 • “Upgrading Db2 servers in a test environment” on page 40 • “Taking a Db2 server offline for upgrade or for converting to a Db2 pureScale environment” on page 42

Table 1. Upgrade plan details for Db2 servers. (continued)

Upgrade plan	**Details**
Upgrade task	You must include these steps: • Install Db2 Version 11.1 • "Upgrading Db2 Version 10.5 or Db2 Version 10.1 instances" on page 45 (for both Windows and Linux/UNIX) • "Upgrading the Db2 Administration Server (DAS)" on page 47 • "Upgrading databases" on page 49 Review the following upgrade tasks to determine the additional steps that are required to upgrade your environment: • "Upgrading a Db2 server (Windows)" on page 43 • "Upgrading a Db2 server (Linux and UNIX)" on page 52 • "Upgrading Db2 servers with specific characteristics" on page 63 Take note of the time that is required to upgrade your databases.

Table 1. Upgrade plan details for Db2 servers. (continued)

Upgrade plan	Details
Post-upgrade tasks	Review the list of tasks in the "Post-upgrade tasks for Db2 servers" on page 102 topic. It includes the following: • If you set the **diaglevel** database manager configuration parameter to 3 or higher as recommended in the pre-upgrade tasks for Db2 servers, reset this parameter to the value set before the upgrade. • "Adjusting adaptive compression settings" on page 104 • "Adjusting the log space size in upgraded databases" on page 105 • "Backing up Db2 server configuration and diagnostic information" on page 34 • "Activating a database after upgrade" on page 106 • Modify storage group attributes. For details, see "Storage group attributes." in *Database Administration Concepts and Configuration Reference*. • "Managing Db2 server behavior changes" on page 107 • If the automatic collection of statistics that are failed on certain system catalog tables during database upgrade, see " Collecting catalog statistics" in *Troubleshooting and Tuning Database Performance* • "Rebinding packages in upgraded databases" on page 108 • Refresh the data in existing materialized query tables • "Upgrading explain tables" on page 109 • "Verifying upgrade of Db2 servers" on page 110 was successful • "Backing up databases before or after upgrade" on page 32 • Migrate to SQL replication Version 10.1. In addition, consider adding the following tasks to your upgrade plan: • Database log directories have been changed • If you upgrade a Db2 server that runs high availability disaster recovery (HADR), re-initializing HADR may be required. For details, see "Initializing high availability disaster recovery (HADR)" in *Data Recovery and High Availability Guide and Reference*. • After you upgrade statistics for your upgraded databases, determine whether index or table reorganization is necessary by running the **REORGCHK** command. For details, see "Determining when to reorganize tables and indexes" in *Troubleshooting and Tuning Database Performance*. • Tune your Db2 server after the upgrade is completed. See "Tuning database performance" in *Troubleshooting and Tuning Database Performance*. • Remove the use of "Deprecated or discontinued functionality that affects Db2 server upgrades" on page 20 • "Adopting new Version 11.1 functionality in upgraded databases" on page 111, where appropriate, to improve performance at the Db2 server level. Review manageability, performance, and scalability enhancements in What's New for Db2 Version 11.1 to determine what new functionality you might want to apply to your environment.

2. If you must be able to reverse the upgrade, add details to the plan about the tasks that are required to “Reversing Db2 server upgrade” on page 120.
3. Combine with the upgrade plan for other components such as Db2 data server clients, database applications, routines, tools, and scripts to create an overall upgrade plan for your Db2 environment.

Planning to upgrade Db2 clients

Planning the upgrade of your clients requires that you review all of the applicable upgrade prerequisites, pre-upgrade tasks, upgrade tasks and post-upgrade tasks.

Procedure

To create an upgrade plan for your Db2 clients:

1. Write the upgrade plan for Db2 clients, using all the details that apply to your environment:

Table 2. Upgrade plan details for Db2 clients.

Upgrade plan	**Details**
Prerequisites	Ensure that you: • Meet the installation requirements for Db2 database products described in *Installing Db2 Servers*. • Resolve any support issues in “Upgrade essentials for clients” on page 122 including client and server connectivity. • Meet all prerequisites for the upgrade task and subtasks, especially obtaining root or Local Administrator access and required Db2 authorization.
Pre-upgrade tasks	Include the following tasks: • “Upgrade Db2 servers” on page 9 • “Backing up client configuration information” on page 126 In addition, check the list of “Pre-upgrade tasks for clients” on page 125 for optional tasks that you might want to perform for your environment such as “Upgrading clients in a test environment” on page 127.
Upgrade task	You must include these steps: • Install Db2 Version 11.1 client • Upgrade client instance Review the following upgrade tasks to determine the additional steps that are required to upgrade your environment: • “Upgrading to Data Server Client (Windows)” on page 128 • “Upgrading to Data Server Runtime Client (Windows)” on page 129 • “Upgrading clients (Linux and UNIX)” on page 131
Post-upgrade tasks	Include the following tasks: • Review “Db2 server behavior changes” on page 17 • “Verifying your client upgrade” on page 137 was successful • Bind the database utilities and the Db2 CLI bind files. For details, see “Binding bind files after installing fix packs”.

2. Combine with the upgrade plan for other components such as Db2 servers, database applications, and routines to create an overall upgrade plan for your Db2 environment.

Planning to upgrade database applications and routines

Planning to upgrade your database applications and routines requires that you review all of the applicable pre-upgrade tasks, upgrade prerequisites, upgrade tasks, and post-upgrade tasks.

Procedure

To create an upgrade plan for your database applications and routines:

1. Write the upgrade plan for database applications, using all the details that apply to your environment:

Table 3. Upgrade plan details for database applications.

Upgrade plan	**Details**
Prerequisites	Ensure that you: • meet the installation prerequisitesinstallation requirements for Db2 database products described in *Installing Db2 Servers*. • meet the development software requirements. For details, see “Support for elements of the database application development environment” in *Getting Started with Database Application Development* • resolve any support issues in “Upgrade essentials for database applications” on page 138 during the upgrade. • meet all prerequisites for the upgrade task and subtasks, especially obtaining required Db2 authorization.
Pre-upgrade tasks	Include the following tasks: • “Upgrade Db2 clients” on page 122 or install the Db2 Version 11.1 application driver. • Test your database applications in a Db2 Version 11.1 testing environment. If your applications run successfully, the rest of the upgrade steps are not required. In addition, check the list of “Pre-upgrade tasks for database applications and routines” on page 145 for optional tasks that you might want to perform for your environment. Even if your current operating system and development software are supported, consider including the following tasks to improve application performance: • Upgrade your operating system to the latest supported level • Upgrade your development software to the latest supported level

Table 3. Upgrade plan details for database applications. (continued)

Upgrade plan	Details
Upgrade task	You must include these steps: • Modify your application code to support changes in Db2 Version 11.1 and to remove use of functionality that is discontinued in Db2 Version 11.1. • Modify your application to support changes specific to the development environment. • Rebuild all database applications after completing your modifications. • Test your database applications using Db2 Version 11.1. Review the following upgrade tasks to determine the additional steps that are required by your development environment to upgrade database applications: • "Upgrading embedded SQL applications" on page 148 • "Upgrading CLI applications" on page 149 • "Upgrading Java applications that use IBM Data Server Driver for JDBC and SQLJ" on page 150 • "Upgrading ADO.NET applications" on page 151 • "Upgrading scripts" on page 152
Post-upgrade tasks	Perform the recommended "Post-upgrade tasks for database applications and routines" on page 158, especially: • Tune performance of your database applications. • Remove the use of "Deprecated or discontinued functionality that affects Db2 server upgrades" on page 20. • "Adopting new Version 11.1 functionality in database applications and routines" on page 159 where appropriate.

2. Write the upgrade plan for routines, using all the details that apply to your environment:

Table 4. Upgrade plan details for routines.

Upgrade plan	Details
Prerequisites	Ensure that you: • meet the development software requirements. For details, see "Support for elements of the database application development environment" in *Getting Started with Database Application Development*. • resolve any support issues in "Upgrade essentials for routines" on page 144 during the upgrade. • meet all prerequisites for the upgrade task and subtasks, especially obtaining required Db2 authorization.
Pre-upgrade tasks	Include the following task: • Test your routines in a Db2 Version 11.1 testing environment. If your routines run successfully, the rest of the upgrade steps are not required. In addition, check the list of "Pre-upgrade tasks for database applications and routines" on page 145 for optional tasks that you might want to perform for your environment. Even if your development software is supported, consider upgrading your development software to the latest supported level.

Table 4. Upgrade plan details for routines. (continued)

Upgrade plan	Details
Upgrade task	You must include these steps: • Modify your routines to support changes in Db2 Version 11.1 and to remove use of functionality that is discontinued in Db2 Version 11.1. • Modify your routines to support changes specific to the development environment. • Rebuild all external routines after completing your modifications. • Retest your routines using Db2 Version 11.1. Review the following upgrade tasks to determine the additional steps that are required by your development environment to upgrade routines: • "Upgrading C, C++, and COBOL routines" on page 154 • "Upgrading Java routines" on page 156 • "Upgrading .NET CLR routines" on page 157
Post-upgrade tasks	Perform the recommended "Post-upgrade tasks for database applications and routines" on page 158, especially: • Remove the use of "Deprecated or discontinued functionality that affects Db2 server upgrades" on page 20 • "Adopting new Version 11.1 functionality in database applications and routines" on page 159 where appropriate

3. Combine with the upgrade plan for other components such as Db2 data server clients and Db2 servers to create an overall upgrade plan for your Db2 environment.

Upgrade Db2 servers

Upgrading to Db2 Version 11.1 requires that you upgrade your existing Db2 servers.

Upgrading your Db2 server requires that you install a Db2 Version 11.1 copy and then upgrade all the instances and databases to be able to run them under the Db2 Version 11.1 copy.

You can directly upgrade existing Db2 Version 9.7, Db2 Version 10.1, or Db2 Version 10.5 instances and databases to Db2 Version 11.1. Learn details, limitations about the upgrade process, and possible issues that you must be aware of in "Upgrade essentials for Db2 servers" on page 12. Refer to the upgrading Db2 server tasks for details on how to upgrade to Db2 Version 11.1. In the upgrading Db2 server topics, the term *pre-Db2 Version 11.1 copy* refers to Db2 Version 9.7, Db2 Version 10.1, or Db2 Version 10.5.

On Windows operating systems, you can automatically upgrade an existing pre-Db2 Version 11.1 copy. If you choose to upgrade your existing Db2 copy during installation, you need to only upgrade your databases after installation.

If your Db2 servers are running on a release before Db2 Version 9.7, upgrade them first to Db2 Version 9.7, Db2 Version 10.1, or Db2 Version 10.5, and then upgrade to Db2 Version 11.1. It is recommended that you upgrade to the latest fix pack of Db2 Version 9.7.

Upgrade to Db2 Version 11.1 is supported for the following Db2 products:

Table 5. Db2 database products supported for upgrade

Db2 Version	**Db2 product name**
Version 10.5	• Db2 Advanced Enterprise Server Edition • Db2 Enterprise Server Edition • Db2 Workgroup Server Edition • Db2 Personal Edition • Db2 Express® Server Edition • Db2 Connect Enterprise Edition • Db2 Connect Unlimited Edition • Db2 Connect Application Server Edition • IBM® Db2 Performance Optimization Feature for Enterprise Server Edition • Db2 Storage Optimization Feature • IBM Db2 Advanced Access Control Feature • IBM Db2 High Availability Feature for Express Edition • IBM Homogeneous Replication Feature for Db2 Enterprise Server Edition • IBM Data Server Client • IBM Data Server Runtime Client
Version 10.1	• Db2 Advanced Enterprise Server Edition • Db2 Enterprise Server Edition • Db2 Workgroup Server Edition • Db2 Personal Edition • Db2 Express Server Edition • Db2 Connect Enterprise Edition • Db2 Connect Unlimited Edition • Db2 Connect Application Server Edition • IBM Db2 Performance Optimization Feature for Enterprise Server Edition • Db2 Storage Optimization Feature • IBM Db2 Advanced Access Control Feature • IBM Db2 High Availability Feature for Express Edition • IBM Homogeneous Replication Feature for Db2 Enterprise Server Edition • IBM Data Server Client • IBM Data Server Runtime Client

Table 5. Db2 database products supported for upgrade (continued)

Db2 Version	Db2 product name
Version 9.7	• Db2 Enterprise Server Edition • Db2 Workgroup Server Edition • Db2 Personal Edition • Db2 Express Server Edition • Db2 Express-C • Db2 Connect Enterprise Edition • Db2 Connect Unlimited Edition • Db2 Connect Application Server Edition • IBM Db2 Performance Optimization Feature for Enterprise Server Edition • Db2 Storage Optimization Feature • IBM Db2 Advanced Access Control Feature • IBM Db2 High Availability Feature for Express Edition • IBM Homogeneous Replication Feature for Db2 Enterprise Server Edition • IBM Data Server Client • IBM Data Server Runtime Client

Note: It is recommended that the Db2 server be at the same fix pack level as the clients.
For Db2 products not supported, refer to “Deprecated or discontinued functionality that affects Db2 server upgrades” on page 20.

Supported upgrade paths for Db2 servers

You must understand the supported upgrade paths for Db2 servers before you upgrade your Db2 environment.

If you are upgrading from Db2 Version 9.7, Db2 Version 10.1, or Db2 Version 10.5, follow the upgrade plan that is detailed in “Planning to upgrade Db2 servers” on page 2.

If you are upgrading a Db2 pureScale instance from Db2 Version 10.1, follow the upgrade steps that are detailed in “Upgrading a Db2 pureScale server” on page 73.

If you are upgrading from a release earlier than Db2 Version 9.7, upgrade them first to Db2Version 9.7, Db2Version 10.1, or Db2Version 10.5, and then upgrade to Db2 Version 11.1.

Table 6. Upgrade paths

	Version 11.1 single partition Enterprise Server Edition	Version 11.1 multiple partition	Version 11.1 with Db2 pureScale Feature
Version 9.7, Version 10.1 or Version 10.5 single partition Enterprise Server Edition	Yes	Yes	Yes

Table 6. Upgrade paths (continued)

	Version 11.1 single partition Enterprise Server Edition	**Version 11.1 multiple partition**	**Version 11.1 with Db2 pureScale Feature**
Version 9.7, Version 10.1 or Version 10.5 multiple partition	Yes. You must drop all but one partition before or after you upgrade the instance to Version 11.1.	Yes	Yes. You cannot upgrade an instance from Version 11.1 multi-partition Enterprise Server Edition to a Db2 pureScale instance. You must first consolidate data on a single partition before or after you upgrade the instance and database to Version 11.1 and then convert the single partition Enterprise Server Edition instance to Db2 pureScale instance.
Version 10.1 or Version 10.5 Db2 pureScale Feature	No	No	Yes

Upgrade essentials for Db2 servers

Upgrading Db2 servers to Db2 Version 11.1 requires an understanding of upgrade concepts, upgrade restrictions, upgrade recommendations, and your Db2 server. When you have a complete understanding of what upgrading your Db2 server involves, you can create your own upgrade plan.

Consider the following factors to develop a complete understanding of upgrading Db2 servers to Db2 Version 11.1:

- “Db2 command actions to upgrade instances and databases”
- “Upgrade restrictions for Db2 servers” on page 14
- “Best practices for upgrading Db2 servers” on page 23
- “Disk space requirements for Db2 server upgrades” on page 21
- “Support changes for 32-bit and 64-bit Db2 servers” on page 22
- “Db2 server behavior changes” on page 17
- “Deprecated or discontinued functionality that affects Db2 server upgrades” on page 20
- “Migration from non-Db2 relational database management systems” on page 26

Db2 command actions to upgrade instances and databases

Learning what actions take place when you invoke the commands to upgrade instances and databases gives you a better understanding of the upgrade process for Db2 servers.

Instance upgrade

When the instance upgrade is called explicitly using the **db2iupgrade** command, or implicitly when you install Db2 Version 11.1 on Windows and select the **Work with Existing** option and then choose a pre-Version 11.1 copy with the **upgrade** action, the command does the following things:

- Calls the **db2ckupgrade** command.
- Upgrades an existing instance to a new instance under a Db2 Version 11.1 copy.
- Upgrades instance profile registry variables. The global profile registry variables set by the user are not upgraded.
- Upgrades the database manager configuration file.
- Sets the **jdk_path** database manager configuration parameter.
- Upgrades the `db2audit.cfg` audit configuration file when the audit facility is enabled.
- Uses the `SSLconfig.ini` SSL configuration file to set the new database manager configuration parameters to the corresponding SSL parameter value in this file and upgrades the instance profile registry setting `DB2COMM=SSL`.

For a successful instance upgrade, all files must exist for all instances and all files must have write access granted.

Review the **db2iupgrade** command for more information about the command and the options that can be specified.

Database directory upgrade

The database directory is accessed when you issue commands such as **LIST DATABASE DIRECTORY** or **UPGRADE DATABASE** command.

Database upgrade

When the database upgrade is called explicitly using the **UPGRADE DATABASE** command the following database entities might be converted during the database upgrade:

- Database configuration file
- Log file header
- Global log file header file
- Table root page for all tables
- Index root page for all tables
- Catalog tables
- Storage group files
- Buffer pool files
- Table space files
- History file

The **UPGRADE DATABASE** command upgrades the files SQLSPCS.1, SQLSPCS.2 , SQLSGF.1, and SQLSGF.2 to support new functionality on automatic storage table spaces such as removing storage paths from a database and rebalancing automatic storage table spaces after you add or drop storage paths from a database.

The **UPGRADE DATABASE** command automatically collects statistics for all system catalog tables during database upgrade. The following table shows the **RUNSTATS** command called for the automatic collection of statistics:

Table 7. ***RUNSTATS*** *command for automatic statistics collection*

auto_runstats	**User profile**	**RUNSTATS command**
Enabled	Exists	**RUNSTATS** command with the SET PROFILE parameter using the information in the STATISTICS_PROFILE column in SYSCAT.TABLES.
Enabled	Does not exist	**RUNSTATS** command with default parameters
Disabled	N/A	**RUNSTATS** command from the most recent call to the **RUNSTATS** command.[1]

Note:

1. If statistics were previously collected for the table, the **RUNSTATS** command is issued as indicated in the table. If there are no statistics collected for the table, the **RUNSTATS** command is not issued.

The automatic collection of statistics for all system catalog tables ignores any exclusion policies defined in the health monitor. Also, if you have manually modified your system catalog table statistics via updates to SYSSTATS views, manually reissue these updates to the SYSSTATS views.

Upgrade restrictions for Db2 servers

Before you start to upgrade your Db2 server, you must understand what the support for upgrade is and what the restrictions are.

What is supported?

- Upgrading to Db2 Version 11.1 is supported from Db2 Version 10.5, Db2 Version 10.1, or Db2 Version 9.7. If you have an earlier version of Db2, you must upgrade to Db2 Version 10.5 or Db2Version 10.1 Db2 Version 9.7 before upgrading to Db2 Version 11.1.
- Upgrading to a Db2 Version 11.1 non-root installation is supported from a Db2 Version 10.5, Db2 Version 10.1, or Db2 Version 9.7 non-root installation. Upgrading to a Db2 Version 11.1 non-root installation from a pre-Db2 Version 11.1 root installation is not supported.
- On Windows operating systems, the **upgrade** action shows for existing Db2 copies that can be upgraded during the installation of Db2 Version 11.1. This action automatically installs Db2 Version 11.1 and upgrades all of your instances and your Db2 Administration Server (DAS) running on the Db2 copy. This action also uninstalls the Db2 copy and any add-on products installed in this copy. If you do not choose the **upgrade** action, you must manually upgrade your instances and your DAS after installation.
- On Linux and UNIX operating systems, the **upgrade** action is not available and you can only install a new copy of Db2 Version 11.1. You have to manually upgrade your instances after installation. You can manually upgrade your existing DAS.
- Instance bit size is determined by the operating system where Db2 Version 11.1 is installed, and support for 32-bit kernels and 64-bit kernels has changed. See Table 12 on page 23.
- Upgrading from a system with multiple copies of Db2 Version 10.5, Db2 Version 10.1, or Db2 Version 9.7 of all levels is supported. On Windows operating systems, you must be aware of the restrictions on coexistence of previous versions of the Db2 database products. Refer to "Updating Db2 copies (Windows)" in *Database Administration Concepts and Configuration Reference*.

- Upgrading from a partitioned database environment with multiple database partitions is supported.
- Restoring full database offline backups from pre-Db2 Version 11.1 copies is supported. However, rolling forward of logs from a previous level is not possible. Review Backup and restore operations between different operating systems and hardware platforms "Backup and restore operations between different operating systems and hardware platforms" in *Data Recovery and High Availability Guide and Reference* for complete details about upgrade support using the **RESTORE DATABASE** command.
- In upgraded databases with the **RESTRICT_ACCESS** database configuration parameter set to YES, you must grant the USAGE privilege to non-DBADM users on SYSDEFAULTUSERWORKLOAD. Otherwise, these users are unable to submit any work to the database.
- Upgrading a database that contains encrypted data in a Db2 instance that does not support encryption (where the IBM Global Security Kit is not installed) is supported. The encrypted data is included in the upgraded database, but you cannot access the encrypted data. To access the encrypted data, upgrade or restore the database to an instance that supports encryption.

What is unsupported?

Db2 Version 11.1 installation fails if the following situations exist:

- The operating system is not supported. You must upgrade to a supported version of the operating system before you upgrade to Db2 Version 11.1 or upgrade to a new Db2 server that meets the operating system requirements. See "Upgrading to a new Db2 server" on page 68 and "Installation requirements for Db2 database products" in *Installing Db2 Servers*.
- A version of Db2 before Version 9.7 is installed.

The **db2iupgrade** command fails if the following situations exist:

- You do not have authorization to upgrade the instance.
- The instance that you are trying to upgrade is active. Run the **db2stop** command to stop the instance.
- The instance is already at Db2 Version 11.1 or later. Run the **db2iupdt** command to update to a different fix pack levels or copies of Db2 Version 11.1.
- You try to upgrade from Db2 Version 11.1 back to Db2 Version 10.5, Db2 Version 10.1, or Db2 Version 9.7. "Reversing Db2 server upgrade" on page 120 is possible, however, you must follow the prerequisites and steps in this procedure.
- The type of instance that you are trying to upgrade to the Db2 Version 11.1 copy is unsupported. The following table describes the upgrade support for each type of instance by Db2 database product:

Table 8. Instance upgrade support for Db2 Version 11.1 database products

Instance type	Node type	Upgrade support
client - default type for Db2 clients [1]	Client	• Upgrade to a client, a *standalone*, a *wse*, or an *ese* instance is supported.
standalone	Database server with local clients	• Upgrade to a *standalone*, a *wse*, or an *ese* instance is supported. • Upgrade to a *client* instance is unsupported.

Table 8. Instance upgrade support for Db2 Version 11.1 database products (continued)

Instance type	Node type	Upgrade support
ese - default type for Db2 Enterprise Server Edition, Db2 Advanced Enterprise Server Edition, Db2 Workgroup Server Edition, and Db2 Advanced Workgroup Server Edition.	Partitioned database server with local and remote clients or Enterprise Server Edition with local and remote clients	• Upgrade to an *ese* instance is supported. • Upgrade to a *standalone* or a *wse* instance from single database partition environments creates a *standalone* or *wse* instance[2](Linux and UNIX only) • Upgrade to a *client* instance is unsupported.
dsf - default type for Db2 pureScale feature	Enterprise Server Edition with local and remote clients	• Upgrade to a *dsf* instance is supported.

Note:

1. The highest level for each Db2 database product is the default instance type as indicated in Table 8 on page 15 ordered from lower to higher-level. Each instance type supports instance types of a lower-level. For example, the *ese* instance type supports *wse*, *standalone*, and *client*. You can use the **db2icrt** command with the **-s** parameter to create instances of a lower-level. If you do not specify the **-s** parameter, the instance is created using the highest level of instance type supported by the Db2 database product installed.
2. During upgrade, the previous database manager configuration parameters will be retained and available after the instance is upgraded to a new version.

- The **db2ckupgrade** command fails and causes the **db2iupgrade** command to fail. The **db2iupgrade** command calls the **db2ckupgrade** command to verify whether cataloged local databases are ready for upgrade to Db2 Version 11.1.

The **UPGRADE DATABASE** command fails if the following situations exist:

- You do not have authorization to upgrade the database.
- A cataloged database does not exist.
- Database upgrade encounters any of the problems described in the reason codes of error message "SQL1704N" in *Message Reference Volume 2*.
- User-defined distinct types (UDTs) are encountered with the names ARRAY, BINARY, CURSOR, DECFLOAT, ROW, VARBINARY, or XML. You must drop these UDTs and re-create them with different names before database upgrade.
- Database objects were created using restricted schema names described in the error message "SQL0553N" in *Message Reference Volume 2*. The list of restricted schema names now includes SYSPUBLIC.
- A Db2 Version 9.7, Version 10.1 or Version 10.5 Fix Pack 6 or earlier database is enabled as a high availability disaster recovery (HADR) standby database.
- A Db2 pureScale database is enabled as a high availability disaster recovery (HADR) standby database, and does not meet the supported 10.5 Fix Pack level for upgrading a standby without reinitialization.

Review the FAQ technote "http://www-01.ibm.com/support/docview.wss?uid=swg21298716" for supported 10.5 Fix Pack levels.

Db2 server behavior changes

Changes to Db2 registry variables, configuration parameters, database physical design characteristics, and database authorities and privileges can result in Db2 server behavior changes that might impact your upgrade.

Generally, instance profile variables that you set in your Db2 profile registry or your system environment retain their values after an instance upgrade. Some global profile registry variables, such as **DB2SYSTEM** and **DB2PATH**, are set by the Db2 installation procedure or instance upgrade. However, the global profile registry variables that you set by running the **db2set** command with the **-g** option are not upgraded. Therefore, you must define them after upgrade.

Existing database and database manager configuration parameters also, generally, retain their values after upgrade. However, the default values assigned to new parameters or the new default values assigned to existing parameters might change the behavior or performance of your applications.

Changes that impact all pre-Version 11.1 releases

The following tables describe in detail the upgrade impact of all of the changes to variables, database and database manager configuration parameters, physical design characteristics of databases, and database authorities and privileges:

- Registry Variables
  - New registry variables (“New registry variables”)
  - Changes to existing registry variables (“Changes to existing registry variables”)
  - Deprecated and discontinued registry variables
- Database manager configuration parameters
  - New database manager configuration parameters (“New database manager configuration parameters” on page 18)
  - Changes to existing database manager configuration parametersChanges to existing database manager configuration parameter (“Changes to existing database manager configuration parameters” on page 18)
  - Deprecated and discontinued database manager configuration parameters
- Database configuration parameters
  - New database configuration parameters (“New database configuration parameters” on page 18)
  - Changes to existing database configuration parameter (“Changes to existing database configuration parameters” on page 19)
  - Deprecated and discontinued database configuration parameters
- Changes to physical design characteristics of databases (“Changes to physical design characteristics of databases” on page 19)
- Changes to authorities and privileges (“Changes to authorities and privileges” on page 19)

New registry variables

New registry variables that are introduced in Db2 Version 11.1 have no impact on Db2 upgrade.

For more information, see “Some registry and environment variables have changed” in *What's New for Db2 Version 10.5*.

Changes to existing registry variables

The following table describes the upgrade impact of changes to existing registry variables:

Table 9. Changes to existing registry variables

Name	Upgrade impact
`DB2DSDRIVER_CFG_PATH`	This variable now specifies multiple configuration files at same or different locations with different names. If filename is not specified in a path and name pair, then the file name defaults to a value of db2dsdriver.cfg.

For more information, see "Some registry and environment variables have changed" in *What's New for Db2 Version 10.5*.

Deprecated and discontinued registry variables

You should remove the use of registry variables that are deprecated because the functionality associated with the variable is obsolete or has been replaced by new functionality. See "Deprecated registry variables" in *What's New for Db2 Version 10.5* to determine the upgrade impact of deprecated registry variables. See "Discontinued registry variables" in *What's New for Db2 Version 10.5* to determine the upgrade impact of discontinued registry variables.

If you are upgrading from Db2 Version 10.1 or earlier, consider removing deprecated registry variables in pre-Version 11.1 releases because the functionality associated with the variable is obsolete or replaced by new functionality. Also, remove the use of discontinued registry variables in pre-Version 11.1 releases as they do not have the intended effect. See "Changes that impact Version 10.1 or earlier releases" on page 20 for details.

New database manager configuration parameters

New database manager configuration parameters that are introduced in Db2 Version 11.1 have no impact on Db2 upgrade. If you are upgrading from Db2 Version 9.7, review new database manager configuration parameter in Version 10.1 that have an impact when upgrading from those releases. See Db2 server behaviour changes for details.

Changes to existing database manager configuration parameters

Changes to database manager configuration parameters introduced in Db2 Version 11.1 have no impact on Db2 upgrade. If you are upgrading from Db2 Version 9.7, review the changes for database manager configuration parameters in Version 10.1 that have an impact when upgrading from those releases. See Db2 server behaviour changes for details.

Deprecated and discontinued database manager configuration parameters

No database manager configuration parameters are deprecated or discontinued in this release. However, if you are upgrading from Db2 Version 9.7, consider removing deprecated database manager configuration parameters in Version 10.1 releases because the functionality associated with the parameters is obsolete or replaced by new functionality. Also, remove the use of discontinued database manager configuration parameters in Version 10.1 releases as they do not have the intended effect. See "Changes that impact Version 10.1 or earlier releases" on page 20 for details.

New database configuration parameters

The following table describes the upgrade impact of the default values of new database configuration parameters:

Table 10. New database configuration parameters

Name	Upgrade impact
dft_table_org	This parameter specifies whether a user table is created as a column-organized table (value COLUMN) or a row-organized table (value ROW) when the ORGANIZE BY COLUMN or the ORGANIZE BY ROW clause is not specified on the CREATE TABLE statement.The default value for this parameter is ROW, which has no impact on upgrade. If you have DDL scripts and you plan to change the default orientation of tables, modify your exiting scripts to specify the ORGANIZE BY COLUMN or the ORGANIZE BY ROW clause for the CREATE TABLE statements to ensure that tables are created with the proper orientation regardless of the setting of this parameter.

For more information, see "Changes to database configuration parameters" in *What's New for Db2 Version 10.5*.

Changes to existing database configuration parameters

The following table describes the upgrade impact of changes to existing database configuration parameters:

Table 11. Changes to existing database configuration parameters

Name	Upgrade impact
hadr_syncmode	In Version 10.5, the default value for **hadr_syncmode** is changed from NEARSYNC to ASYNC for Db2 pureScale databases. For all other database types, the default for **hadr_syncmode** remains NEARSYNC.
hadr_target_list	In Version 10.5, initializing HADR without setting this parameter is deprecated. You should set this parameter regardless of the number of standby databases as part of the initialization process.

For more information, see "Changes to database configuration parameters" in *What's New for Db2 Version 10.5*.

Deprecated and discontinued database configuration parameters

You must remove the use of database configuration parameters that are deprecated and discontinued because the functionality associated with the variable is obsolete or replaced by new functionality. See "Some database configuration parameters are deprecated or discontinued" in *What's New for Db2 Version 10.5* to determine the upgrade impact of deprecated and discontinued database configuration parameters.

If you are upgrading from Db2 Version 9.7, consider removing deprecated database configuration parameters in pre-Version 10.1 because the functionality associated with the parameter is obsolete or replaced by new functionality. Also, remove the use of discontinued database configuration parameters in Version 10.1 as they do not have the intended effect. See "Changes that impact Version 10.1 or earlier releases" on page 20 for details.

Changes to physical design characteristics of databases

Review the What's New and Changed documentation to determine whether there are any changes to the physical design characteristics of databases that impact upgrade.

Changes to authorities and privileges

There are no changes to the authorities and privileges in this release.

See "Upgrade impact from Db2 command changes" on page 141 and "Upgrade impact from SQL statement changes" on page 142 for a summary of Db2 command and SQL statement changes with upgrade impact. See the *Command Reference* and *SQL Reference* for details about all the changes in authorization.

Changes that impact Version 10.1 or earlier releases

If you are upgrading from Db2 Version 10.1 or earlier, also review all of the changes to variables, database and database manager configuration parameters, and physical design characteristics of databases between pre-Version 11.1 releases that might also impact your upgrade:

- Db2 server behavior changes between Db2 Version 9.7 and Db2 Version 10.1
- Db2 server behavior changes between Db2 Version 9.5 and Db2 Version 9.7

Deprecated or discontinued functionality that affects Db2 server upgrades

You should be aware of functionality that is deprecated or discontinued in Db2 Version 11.1 that can affect the upgrade of your Db2 server. Also, you should be aware of the Db2 products that are no longer supported because upgrade from these products to Db2 Version 11.1 is unsupported.

To deal with these functionality changes, you must perform additional tasks before or after upgrade. The following list describes changes that are not included in the pre-upgrade and post-upgrade tasks for Db2 servers:

Deprecated or discontinued commands

The **db2IdentifyType1** command and the **STATISTICS YES** parameter of the **LOAD** command are discontinued.

Review "Upgrade impact from Db2 command changes" on page 141 to learn what commands are deprecated and discontinued in Db2 Version 11.1 and how to manage this impact on your database applications and routines.

Raw logs

The use of raw devices for database logging has now been discontinued as of Db2 Version 11.1 and will not be available in future releases. You should use a file system instead of a raw device. Using a file system with non-buffered I/O capabilities enabled, such as Concurrent I/O (CIO) or Direct I/O (DIO), can give you performance comparable to that of using raw devices. The following example illustrates how to change the **newlogpath** parameter setting to a file system directory:

```
db2 UPDATE DATABASE CONFIGURATION USING newlogpath /disk2/newlogdir
```

The new setting does not become effective until the database is in a consistent state and all users are disconnected from the database. The database manager moves the logs to the new location after the first user connects to the database.

If upgrading a database configured with raw logs to Db2 Version 11.1 it is advised to change the **newlogpath** parameter before you begin the upgrade procedure as the **db2ckupgrade** utility will fail if it detects a raw device set for the primary log path. If **UPGRADE DATABASE** detects a raw device is set for the primary log path, it will switch to the default log path so that upgrade processing can proceed. At your earliest convenience you should change the **newlogpath** parameter to something more suitable.

Functionality that was deprecated or discontinued in Db2 pre-V10.5 releases

If you are upgrading from Db2 V10.5 versions, you must also review the changes made in Db2 Version 10.1 or Version 9.7 that might impact your environment after ugprading to Db2 Version 11.1. Review the following topic to learn about additional possible impacts on the upgrade of your Db2 server:

- Deprecated or discontinued functionality in Db2 Version 10.1 for upgrade from Db2 Version 10.1.
- Deprecated or discontinued functionality in Db2 Version 9.7 for upgrade from Db2 Version 9.7.

Disk space requirements for Db2 server upgrades

You must be aware that the upgrade process requires additional disk space. Ensure that you have enough free disk space to complete this process successfully. The following disk space recommendations are applicable for upgrading to Db2 Version 11.1.

System catalog and system temporary table spaces

Ensure that you have sufficient free space on the system catalog and the system temporary table spaces for the databases that you are upgrading. System catalog table space is required for both old and new database catalogs during upgrade. The amount of free space required varies, depending on the complexity of the database, as well as on the number and size of database objects.

System catalog table space (SYSCATSPACE)

Increasing the total size to twice the total of used space is recommended. In other words the amount of free space should be at least the same as the current amount of used space.

Temporary table space (TEMPSPACE1 is the default name)

Increasing the total size to twice the total size of the system catalog table space is recommended.

For the system catalog table space, free pages should be equal to or greater than used pages. Total pages for the system temporary table space should be twice the amount of total pages for the system catalog table space.

To increase the amount of free space on your automatic storage table spaces, you can increase the space on the current storage paths or add a new storage path.

To increase the amount of free space on your System Managed Space (SMS) table spaces, free sufficient disk space on the corresponding file systems or increase the size of your file systems if you are using a volume manager.

To increase the amount of free space on your Database Managed Space (DMS) table spaces, you can increase the size of existing containers. You can also add additional containers although this might trigger data rebalancing. You can reduce the size of the containers after upgrade.

Log file space

The database upgrade process makes changes to system catalog objects. All changes to each system catalog object are performed in a single transaction and need adequate log space to contain this transaction. If there is insufficient log space, this transaction is rolled back and upgrade does not complete successfully.

To ensure sufficient log file space is available, you can set the **logsecond** database configuration parameter to twice the current value of **logprimary** and **logsecond** if the file system containing the log files has enough disk free space to increase this parameter. If you already have available a large log file space, it might not be necessary to increase this parameter. Also on partitioned database environments, you only need to increase the log space in the catalog partition.

You must update these database configuration parameters values before you upgrade the instance to Db2 Version 11.1, because you will not be able to update these database configuration parameters until you issue the **UPGRADE DATABASE** command. If this command fails because there is insufficient log file space, then you can set these database configuration parameters to higher values and then re-issue the **UPGRADE DATABASE** command.

The new database configuration parameter settings for log space can be restored to their original value after the upgrade is complete.

Index space

Each index on every populated table requires one additional page per index to use the following functionality:

- Real-time statistics.
- Deferred cleanup roll out for MDC tables.
- Index rebuild on a populated table.

If you have a limited amount of free disk space for indexes, you can get the error message SQL0289N that indicates the table space is full. Ensure that you have enough free pages in the corresponding index table space to account for one additional page per index on populated tables before:

- Populating tables in databases created in Db2 Version 9.7 or later, real-time statistics are enabled by default in these newly created databases.
- Enabling deferred cleanup roll out by setting **DB2_MDC_ROLLOUT** to `DEFER`, or when **DB2_WORKLOAD** is set to `SAP`.
- Reorganizing or recreating indexes on populated tables.

Automatic storage files

If you enable automatic storage on an existing database by issuing the ALTER DATABASE statement with the ADD STORAGE ON clause, this statement creates the `SQLSGF.1` and `SQLSGF.2` files that are required for maintaining automatic storage.

Support changes for 32-bit and 64-bit Db2 servers

Db2 Version 9.1 or later provides support for 32-bit operating systems on Linux on x86 and Windows operating systems, and 64-bit operating systems on UNIX, Linux and Windows operating systems.

For more details about supported architectures on each operating system, see “Installation requirements for Db2 database products” in *Installing Db2 Servers*

You cannot specify the bit size for the instance when you create or upgrade an instance. The bit size for new instances is determined by the operating system where Db2 Version 11.1 is installed. The following table summarizes the Db2 Version 11.1 bit size support that is available for each of the following operating systems:

Table 12. Db2 Version 11.1 32-bit and 64-bit support available per operating system

Operating systems	Db2 Version 11.1 support available
• 32-bit Windows on x86 and x64 (Using 32-bit product) • 32-bit Linux on x86 • 64-bit Windows on x64 (Using Db2 Version 11.1 32-bit product)	For Db2 Version 11.1 Developer Edition: • 32-bit instances only • 32-bit Db2 client, and GUI tools packages • 32-bit IBM Software Development Kit (SDK) for Java™
• 64-bit Windows on x64 • 64-bit Linux kernel on x64, POWER®, and zSeries	• 64-bit instances • 32-bit and 64-bit Db2 libraries available • 64-bit Db2 server and client • 64-bit applications and routines • 32-bit client side application support • 32-bit fenced stored procedures/UDFs only (non-Java) • Java fenced Stored Procedures/UDFs • 64-bit IBM SDK for Java

The changes in 32-bit and 64-bit support can have an impact in your applications depending on the shared library path that you indicated when you linked the Db2 libraries to your applications. If you specified the Db2 installation path, the applications fail to run because the Db2 Version 11.1 copy has a different installation path. However, if you linked the libraries using the library path under the instance home directory, your applications will run successfully in the following cases:

- If you have 32-bit instances and you upgrade to Db2 Version 11.1 Developer Edition on a 32-bit system. You can only upgrade to 32-bit instances on 32-bit Windows or 32-bit Linux on x86. For any other editions in Db2 Version 11.1, you must upgrade to 64-bit system.
- If you have 64-bit instances and you upgrade to Db2 Version 11.1 on a 64-bit system. You can only upgrade to a 64-bit instance on a 64-bit system.

If you have 32-bit instances and you upgrade to Db2 Version 11.1 on a 64-bit system, you must manage incompatibilities so that your applications and routines can run successfully. Incompatibilities arise because of discontinued functionality or incorrect shared library path specification. Table 12 summarizes the details on the available 32-bit and 64-bit support. For example, 32-bit unfenced stored procedures in any supported language except Java are not supported. By dropping and recreating these stored procedures as fenced you can resolve this issue.

Best practices for upgrading Db2 servers

When planning your Db2 server upgrade, there are a number of best practices to consider. Review these best practices before you start your upgrade.

Review changes in existing Db2 database product functionality

Changes in existing functionality introduced in Db2 Version 11.1 can potentially impact your applications, scripts, maintenance processes, and any other aspects related your Db2 server upgrade process.

Changes in existing functionality introduced in pre-Db2 Version 11.1 releases can also have an impact. Review these changes and plan how to address these changes before the upgrade:

- Changed functionality in Db2 Version 9.7
- Changed functionality in Db2 Version 10.1
- Changed functionality in Db2 Version 10.5

Upgrading in a test environment allows you to learn about possible issues, evaluate the impact on your environment and find a resolution.

Perform hardware and operating system upgrades before Db2 database product upgrade

The supported UNIX, Linux and Windows operating systems have changed in Db2 Version 11.1. Review "Installation requirements for Db2 database products" in *Db2 pureCluster™ Feature Installation and Upgrade Guide* to determine whether your operating system version is supported and if you need to upgrade your operating system before installing Db2 Version 11.1. Newer versions of operating systems can also bring new hardware requirements.

Performing hardware and operating system upgrades separately from Db2 database product upgrade simplifies problem determination if you encounter upgrade difficulties. If you upgrade your software or hardware before a Db2 database product upgrade, ensure that your system is operating as expected before attempting to upgrade your Db2 database product.

If you have a Db2 Version 9.7 copy on SUSE Linux Enterprise Server 10, first apply Db2 Version 9.7 Fix Pack 2 or later, before you upgrade the operating system to SUSE Linux Enterprise Server 11.

If you are upgrading a pre-Db2 Version 11.1 copy on POWER4 processor-based systems, first upgrade to POWER10 processor-based systems before upgrading to Db2 Version 11.1. POWER3 processor-based systems are not supported in Db2 Version 11.1.

Benchmark Db2 server performance

Run a number of performance tests before upgrading your Db2 server. The **db2batch** benchmark tool helps you to collect elapsed and CPU times for running queries. You can use this tool to develop performance tests. Record the exact environment conditions where you run your tests.

Also, keep a record of the **db2expln** command output for each test query. Compare the results before and after upgrade. This practice can help to identify and correct any performance degradation that might occur.

Develop or Select a recovery plan for upgrade

For recoverable databases, a database upgrade can be a recoverable operation. Depending on your environment and upgrade window, it can be possible to take advantage of this recoverability feature to reduce your overall upgrade time and minimize your downtime. This recoverability feature can impact your need for taking an offline database backup before and after the database upgrade. See "Recovering through a Db2 server upgrade" on page 114 to know whether the feature is applicable for the databases in your environment. Your recovery plan must consider the following two scenarios:

- Reversing an upgrade or fall back from Db2 Version 11.1 to a pre-Version 11.1 release.
- Recovering through a database upgrade to a point in time in Version 11.1.

Develop a plan to reverse an upgrade

There is no utility to reverse an upgrade or fall back from Db2 Version 11.1 to a pre-Db2 Version 11.1 release. See "Reversing Db2 server upgrade" on page 120 to learn all the required steps to reverse a database upgrade.

Perform pre-upgrade tasks

There are several pre-upgrade tasks outlined in the "Pre-upgrade tasks for Db2 servers" on page 27 topic that you should execute for a successful upgrade, such as backing up Db2 configuration parameters settings, ensure that you have enough disk free space for table spaces and log files, and verifying that databases are ready for upgrade.

Determine whether to upgrade Db2 servers or clients first

Upgrading your Db2 servers before upgrading your data server clients is the traditional approach to avoid any known restrictions and limitations such as support of new Db2 database product functionality, network protocols, and connectivity. These restrictions and limitations are not associated with Db2 Connect.

Upgrading your data server clients first requires that you manage any incompatibilities between releases. If you must upgrade your client due to a software requirement, make sure that the software supports the Db2 database product version that you are running on your Db2 server. In this case, the software manages any incompatibilities between releases. See Best practices for upgrading clients in the Db2 Version 11.1 documentation for details about incompatibilities.

Upgrade database applications and routines

If you upgrade your Db2 server, you might also need to upgrade your database applications and routines to support changes for 64-bit instances, SQL stored procedures, Java Virtual Machine (JVM), and development software.

Review the factors that can impact your database application upgrade or routine upgrade and make any necessary changes to your database applications and routines to ensure that they run after the upgrade. See "Upgrade essentials for database applications" on page 138 and "Upgrade essentials for routines" on page 144 for details about the factors that can impact your database application upgrade or routine upgrade.

In an upgrade testing environment, you can test and verify that your database applications and routines run successfully in Db2 Version 11.1 to find out if you need to upgrade them. You can also upgrade your database applications and routines before you upgrade your production environment.

Upgrading Db2 High Availability Disaster Recovery (HADR) environments

Upgrading a single partition ESE Db2 Version 10.5 Fix Pack 7 or later primary and standby database to Db2 Version 11.1 is now supported without having to change the database role and without needing to reinitialize HADR. For more information, see "Upgrading Db2 servers in HADR environments (without standby reinitialization)" on page 86.

Upgrading a pureScale Db2 Version 10.5 primary and standby database to Version 11.1 is supported without needing to reinitialize HADR, if the 10.5 Fix Pack is at the supported level. Review the FAQ technote "http://www-01.ibm.com/support/docview.wss?uid=swg21298716" for supported 10.5 Fix Pack levels. For more information, see “Upgrading Db2 servers in HADR pureScale environments (without standby reinitialization)” on page 91.

Upgrading a Db2 Version 9.7, Version 10.1 or Version 10.5 Fix Pack 6 or earlier primary database to Version 11.1 changes the database role from primary to standard. Upgrading a Db2 Version 9.7, Version 10.1 or Version 10.5 Fix Pack 6 or earlier standby databases to Version 11.1 is not supported because these databases are in rollforward pending state. For more information, see “Upgrading Db2 servers in HADR environments” on page 85.

Migrating SQL replication environments

After upgrading your database servers, you can optionally migrate your SQL replication environment.

Upgrading Db2 Spatial Extender

If you had Db2 Spatial Extender installed and you upgraded your spatially-enabled databases to Db2 Version 10.1, see *Upgrading to Db2 Spatial Extender Version 10.1* in *Spatial Extender User's Guide and Reference* for upgrade details specific to Db2 Spatial Extender.

Upgrading Microsoft Cluster Server environments

In a Microsoft Cluster Server (MSCS) environment, install Db2 Version 11.1 as a new copy and then run the **db2iupgrade** command to upgrade the MSCS instance. See “Upgrading Db2 servers in Microsoft Cluster Server environments” on page 101 for details.

Upgrading from Query Patroller to Workload Manager

Query Patroller is discontinued. See Migrating to SQL replication Version 10.5 for details on how to migrate.

Migration from non-Db2 relational database management systems

Migrating from a non-Db2 relational database management system is a more complex process than migrating from a Db2 database product. Therefore, you must carefully determine what the migration process entails and create a porting plan.

The porting plan must include tasks such as, converting your database objects to create the equivalent database objects in a Db2 database, moving the actual data to the new Db2 database and porting your database applications. Porting your applications refers to converting SQL statements, modifying interface calls, and converting any database-specific code to access Db2 databases.

The most common approaches to converting database application code are manual conversion, dynamic call translation, and automated conversion. In general, conversion tools take source code as input and translate data management calls to equivalent SQL calls. Information from the source and target database, and the program code, is used to build the new SQL statements.

The IBM Database Conversion Workbench (DCW) integrates many of the tools that are used for database conversions into a single integrated development environment. Whether converting to Db2 database from another relational database management system (RDBMS), or moving from one version of Db2 database to another version, DCW provides an easy to use framework to take you through the conversion process. DCW is available as a complementary download from the IBM Database Conversion Workbench web page.

The most important and frequently accessed resources that IBM offers to help all aspects of migration from a non-Db2 relational database management systems are as follows:

- The Migration station Web page can help you to find the information that you need to port your application and its data from other database management systems. This web page describes the common migration steps and provides resources including tools and education. Additional resources are provided for IBM customers and IBM Business Partners.
- The worldwide IBM Innovation Centers for Business Partners offer a wide range of complimentary workshops and technical seminars. See the training resources page to find out details and schedules.
- The IBM Virtual Innovation Center™ (VIC) is an online knowledge and enablement center that provides educational courses, live mentoring, online technical support, solution roadmaps, client simulations, answers to FAQs, case studies, and discussion forums.
- The Db2 Migration Factory end-to-end offering for strategic IBM Business Partners that includes migration toolkits, complementary online education, information, sales teams, and other resources to assist you in planning and implementing your migration to Db2 products from Oracle, Sybase, and Microsoft SQL server.
- The developerWorks® Information Management website offers technical resources for Db2 Information Management software. It features product information, downloads, learning resources, support, and communities. On this website, you can find many articles and tutorials that can help you to learn about the functionality of Db2 database products and how to use them in your applications.

Pre-upgrade tasks for Db2 servers

Before you upgrade your Db2 server, review the upgrade essentials for Db2 servers, including recommendations, restrictions, and disk space requirements to identify the changes or restrictions that can affect your upgrade. You must be ready to address any issues before upgrade in order to have a successful upgrade.

Procedure

Prepare for the upgrade of your Db2 servers by performing the following tasks:

1. Ensure that you have at least one free page of index space per object index to eliminate the overhead of a potential index rebuild. If an index root page does not have enough free space during upgrade, then the index will need to grow by one page. If a free page cannot be found in the index object, then a page will be requested from the tablespace. If the tablespace is full, then the entire index object will be marked invalid and will be rebuilt when the underlying table is accessed for the first time after upgrade.
2. If you use distributed transactions involving Db2 databases, ensure that the databases to be upgraded do not contain any indoubt transactions by using the **LIST INDOUBT TRANSACTIONS** command to get a list of indoubt transactions and to interactively resolve any indoubt transactions.

3. Verify that the server time has not been reset or changed, and that the times associated with the members in multi-partitioned database environments are in sync. If the server time is not verified, the upgrade might return a SQL0440N error message.
4. Verify that databases are ready for Db2 upgrade to identify any problems before the actual upgrade. You must resolve them before you proceed with the upgrade.

   Refer to "Verifying that your databases are ready for upgrade" on page 29.
5. Upgrade from Db2 Query Patroller to Workload Manager. Query Patroller is discontinued. Perform the steps in "Migrating from Query Patroller to Db2 workload manager" in the Db2 Version 9.7 documentation.
6. Based on your recovery plan for upgrade, verify existing backup images or create a new backup of your databases to be able to upgrade them to a new upgraded system or restore them in the original pre-upgrade system.

   Refer to "Backing up databases before or after upgrade" on page 32.
7. Back up configuration and diagnostic information to have a record of your current configuration that you can compare with the configuration after the upgrade. You can also use this information to create new instances or databases using the same configuration that you had before upgrade.

   Refer to "Backing up Db2 server configuration and diagnostic information" on page 34.
8. For HADR environments that support upgrade without the need for standby reinitialization (single partition ESE Db2 Version 10.5 Fix Pack 7 or later databases, or Db2 pureScale V10.5 Fix Pack 9 or later databases) , Using high availability disaster recovery monitoring ensure that both the primary's log shipping functionality and the standby's log replaying functionality is working correctly. In order to use the new HADR upgrade procedure without the need for standby reinitializatrion, the **db2ckupgrade** command and the **UPGRADE DATABASE** command validates that the primary's log shipping position equals the standby's log replay position. If this cannot be validated or fails, then HADR must be stopped before upgrade and re-initialized after upgrade.
9. Archive all of the Db2 log files, either for SQL replication or Q replication if the log files are needed by the Capture or Q Capture programs, or for recovery through database upgrade, or for high availability disaster recovery (HADR) replication if the log files are needed to create a standby database.
10. Review the disk space requirements to ensure that you have enough free disk space, system temporary table space and log space for the upgrade and increase table space and log file sizes if necessary. Depending on the number of database objects, you might require more log space to perform the upgrade.

    Refer to "Disk space requirements for Db2 server upgrades" on page 21 and "Increasing table space and log file sizes before upgrade" on page 36.
11. Optional: For recoverable databases, if your recovery plan for upgrade involves a possible rollforward through upgrade, consider turning on the **logindexbuild** database configuration parameter. Database upgrade usually recreates indexes for catalog tables. This operation is not logged. This means that if something goes wrong post upgrade and you need to recover through database upgrade the ensuing rollforward will mark these indexes bad and on first access after the rollforward completes takes the time to recreate these indexes. If this is of a concern to your environment consider turning on **logindexbuild** and index recreation during upgrade is logged. This comes with a trade-off that additional log space is required. For HADR standby

databases that have reads on standby enabled, turning on **logindexbuild** on the primary is highly advised to allow for read only connections once upgrade is completed on the standby.

12. Windows only: If you obtained customized code page conversion tables from the Db2 support service, you need to backup all of the files in the `DB2OLD\conv` directory where *DB2OLD* is the location of your existing pre-Db2 Version 11.1 copy.

    You do not need to backup standard code page conversion tables. Upgrading your pre-Db2 Version 11.1 copy removes these tables because standard code page tables are contained in a Db2 Version 11.1 library.
13. Linux only: Change raw devices to block devices.

    Refer to "Changing raw devices to block devices (Linux)" on page 38.
14. Collect information to troubleshoot issues that might occur while performing an upgrade.

    Refer to "Gathering pre-upgrade diagnostic information" on page 39.
15. Optional: Upgrade your Db2 server in a test environment to identify upgrade issues and to verify that applications, scripts, tools and routines work as expected before upgrading your Db2 server in the production environment.

    Refer to "Upgrading Db2 servers in a test environment" on page 40.
16. Check the "diagnostic error capture level" using the **GET DBM CFG** command. If the diagnostic error capture level (set by the **diaglevel** parameter) is 2 or less, set this parameter to 3 or higher before upgrading. See "Setting the diagnostic log file error capture level" in *Troubleshooting and Tuning Database Performance*.
17. Perform one of the following actions:
    - For Linux and UNIX operating systems, . stop all Db2 processes.
    - For Windows operating systems, stop all Db2 instances, services and applications.
18. Take the Db2 server offline for upgrade.

    Refer to "Taking a Db2 server offline for upgrade or for converting to a Db2 pureScale environment" on page 42.
19. Refresh the data in existing materialized query tables. All materialized query tables that depend on the system views are dropped during database upgrade. After upgrade you must refresh the data in existing materialized query tables by using the **REFRESH TABLE** statement.

Verifying that your databases are ready for upgrade

Before you upgrade your databases, it is important to use the **db2ckupgrade** command to verify that your databases are ready for upgrade.

The **db2ckupgrade** command verifies that a list of conditions is true to succeed at the database upgrade. Also, this command writes to the log file, which is specified with the **-l** parameter, a warning message for a list of conditions that affect database upgrades. See the Command Reference for details about the list of conditions.

The **db2iupgrade** calls the **db2ckupgrade** command. The **db2iupgrade** fails if the **db2ckupgrade** command finds any of the conditions are not true, and returns the error code DBI1205E.

For HADR environments that support upgrade without the need for standby reinitialization (single partition ESE Db2 Version 10.5 Fix Pack 7 or later databases, or Db2 pureScale V10.5 Fix Pack 9 or later databases), an important part of the

HADR upgrade procedure is that the **db2ckupgrade** command validates the log positions of the primary database and all standby databases cataloged in the instances. To ensure the success of the **UPGRADE DATABASE** command in the new release, it is highly recommended to ensure that the **db2ckupgrade** command is run so that this log validation is done. If your upgrade procedure requires you to run **db2ckupgrade** manually, ensure to use the **-allChecks** option in order to validate the log positions.

Also, , Using high availability disaster recovery monitoring ensure that both the primary's log shipping functionality and the standby's log replaying functionality is working correctly before you run the **db2ckupgrade** command. When the **db2iupgrade** command calls the **db2ckupgrade** command, the log positions are validated.

Before you begin

- Ensure that you have SYSADM authority.
- Ensure that all the local databases that you want to upgrade are cataloged. For more information, see Cataloging databases. For more information, see "Cataloging databases" in *Database Administration Concepts and Configuration Reference*.
- On Linux or UNIX operating systems, decompress a Db2 Version 11.1 installation image to be able to run the **db2ckupgrade** command.
- Ensure that you meet the installation requirements for Db2 database products. See "Installation requirements for Db2 database products" in *Installing Db2 Servers*.

Procedure

To verify that your databases are ready for upgrade:

1. Log on to the Db2 server as the Db2 instance owner that you want to upgrade.
2. If the instance that owns the databases that you want to verify is not running, start the instance by running the **db2start** command.
3. From the command-line prompt, change to the appropriate directory:
   - On UNIX or Linux operating systems, change to the *DIRIMG* directory, where *DIRIMG* is the location where you decompressed the Db2 Version 11.1 installation image or the directory where you mounted the Db2 product DVD.
   - On Windows operating system, insert the Db2 Version 11.1 product DVD in the drive and change to the `\db2\Windows\utilities` directory. Or change to the *DIRIMG* directory, where *DIRIMG* is the location where you decompressed the Db2 Version 11.1 installation image.
4. Verify that the local databases that are owned by the current instance are ready to be upgraded and generate a log file by running the **db2ckupgrade** command, as follows:

   ```
   <DIRIMG> db2ckupgrade sample -l db2ckupgrade.log -u adminuser -p password
   db2ckupgrade was successful.  Database(s) can be upgraded.
   ```

   where:
   - *DIRIMG* is the location where you decompressed the Db2 Version 10.5 installation image or the directory where you mounted the Db2 product DVD.
   - *sample* is the database name

- `db2ckupgrade.log` is the log file that is created in the current directory that includes details on errors and warnings

When the **db2iupgrade** command runs the **db2ckupgrade** command, the `update.log` log file is specified for **db2ckupgrade** in the instance home directory for Linux and UNIX operating systems or in the current directory for Windows operating systems.

In a partitioned database environment, run the **db2ckupgrade** command only once. It checks all partitions. Similarly, in a Db2 pureScale environment, run the **db2ckupgrade** from only one member node.

5. If you created user-defined data types using a name that is a system built-in data type name, drop these user-defined data types and re-create them using a different name that is not restricted. The **db2ckupgrade** command returns the SQL0473N error message when user-defined data types have a name that is a system built-in data type name. If you try to upgrade the database, the **UPGRADE DATABASE** command fails.
6. If you created user-defined objects that are dependent on discontinued administrative routines, drop the dependent objects and re-create them using the routine or view that replaces the discontinued routine. The **db2ckupgrade** command returns the DBT5534W warning message when a user-defined object is dependent on discontinued administrative routines. If you upgrade a database that has dependent objects, the **UPGRADE DATABASE** command drops the discontinued administrative routines and marks the dependent objects inoperative or invalid.

   For more details, see "Some administrative routines are discontinued" in *What's New for Db2 Version 10.5*.
7. If you created workload management objects that have a conflict with system-reserved ID during database upgrade, drop these objects and re-create them after you upgrade the database. The **db2ckupgrade** command returns the DBT5512E error message when a workload management object cannot be upgraded because the ID of that object conflicts with a system-reserved ID. Perform the following actions:
   a. Generate the DDL statements to re-create the workload management objects by issuing the **db2look** command with the **wlm** parameter.
   b. Drop all of the workload management objects from the database.

   After upgrading the database, re-create the workload management objects in the upgraded database by issuing the DDL statements that you generated with the **db2look** command.
8. If you created database objects using restricted schema names, drop all the database objects that use reserved schema names and re-create them using a schema name that is not restricted. The **db2ckupgrade** command returns the SQL0553N error message when database objects have restricted schema names. If you try to upgrade the database, the **UPGRADE DATABASE** command fails.
9. If you have identifiers called NULL for column names, routine parameter names, or variable names, qualify, or delimit with quotes these identifiers in your SQL statements to avoid conflict with the NULL keyword.

   The **db2ckupgrade** command writes the ADM4102W warning message to the log file when a database has identifiers called "NULL". If you use identifiers called "NULL" that are not fully qualified or delimited with quotes in your SQL statements, the identifier name might resolve to the NULL keyword instead. This would result in a change in behavior from previous releases. See "Upgrade impact from SQL statement changes" on page 142 for details.

10. If workload connection attributes contain asterisks (*), replace the asterisks (*) with another character. The **db2ckupgrade** command writes the ADM4103W warning message to the log file when workload connection attributes contain asterisks (*).

    Starting with Db2 Version 9.7, you can use a single asterisk (*) as a wildcard character. In some workload attributes, if the intention is to represent an actual asterisk, then you can use two asterisks (**). The UPGRADE DATABASE command replaces the single asterisk (*) with two asterisks (**) depending the type of connection attribute.
11. If you created global variables of XML data type or created compiled SQL functions with parameters of XML data type or XML data type in the RETURNS clause, you must upgrade to the Version 10.1 Fix Pack 1 software or later fix pack releases that support the XML data type in these database objects. If you decide to upgrade to the Version 10.1 software, you must drop these database objects and re-create them specifying a supported data type.

    The **db2ckupgrade** command writes the ADM4004W warning message to the log file when a database has global variables of XML data type or compiled SQL functions with parameters of XML data type or XML data type in the RETURNS clause. The XML data type is not supported on these database objects. Therefore, these database objects are invalidated during the database upgrade.
12. Ensure that the log file for **db2ckupgrade** command contains the following text: `Version of DB2CKUPGRADE being run: Version 11.1`. This text confirms that you are running the correct level of the **db2ckupgrade** command.
13. Check and fix any invalid flavor fields on SQLSPCS files by using the **fixtbspflvr** tool. Details about this tool can be obtained from http://www.ibm.com/support/.

Backing up databases before or after upgrade

For recoverable databases, a database upgrade can be a recoverable operation. Depending on your environment and upgrade window, it can be possible to take advantage of this recoverability feature to reduce your overall upgrade time and minimize your downtime.

This can impact your need for taking an offline database backup before and after the database upgrade. You must review Recovering through a DB2 server upgrade to see whether this is acceptable for the databases in your environment.

If your recovery plan for database upgrade will make use of roll forward through database upgrade, then before the upgrade process to Db2 Version 11.1, it is suggested that you perform an online database backup to reduce the recovery time in case of failure with the upgrade procedure.

If you have a recent backup image available for use that is sufficient as well. Regardless, verify by using **db2ckbkp** that a good backup image exists and ensure all log files needed to recover from the identified backup image exist and are valid by using **db2cklog**. If your recovery plan for database upgrade will make use of roll forward through database upgrade, then after the upgrade process to Db2 Version 11.1, it is recommended that you perform a full online database backup at your earliest convenience. A backup post-upgrade reduces the recovery time and simplifies the recovery procedure in case of failures.

If your recovery plan for database upgrade will not make use of roll forward through database recovery, then before and after the upgrade process to Db2

Version 11.1, it is recommended that you perform a full offline database backup. If an error occurs during the upgrade process, you need full database backups to recover and upgrade your databases.

Note:
- Online database backup images can only be restored to the Db2 Version where they were created. If you intend to restore the backup image to Db2 Version 11.1 then it must be an offline database backup image.
- After you upgrade your instances to Db2 Version 11.1, you cannot back up databases until you upgrade them.

Before you begin

- To back up a database, you require SYSADM, SYSCTRL, or SYSMAINT authority.
- Databases must be cataloged. To view a list of all the cataloged databases in the current instance, enter the following command as an instance user:
  ```
  db2 LIST DATABASE DIRECTORY
  ```

Procedure

To perform a full offline back up for each of your local databases:

1. **For offline backup**: Disconnect all applications and users from the database. To get a list of all database connections for the current instance, issue the **LIST APPLICATIONS** command:
   ```
   db2 LIST APPLICATIONS
   ```
   If all applications are disconnected, this command returns the following message:
   ```
   SQL1611W No data was returned by the Database System Monitor.
   SQLSTATE=00000
   ```
   To disconnect all applications and users, use the **FORCE APPLICATION** command:
   ```
   db2 FORCE APPLICATION ALL
   ```
2. Backup your database using the **BACKUP DATABASE** command. In a Db2 pureScale environment, you can run the **BACKUP DATABASE** command from any member. The following is an example for UNIX operating systems:

   Offline :
   ```
   db2 BACKUP DATABASE database_alias USER username USING password TO backup-dir
   ```
   Online:
   ```
   db2 BACKUP DATABASE database_alias USER username USING password ONLINE TO backup-dir
   ```
   where *database_alias* is the database alias, the user name is *username*, the password is *password*, and the directory to create back up files is *backup-dir*.

   In partitioned database environments, back up all database partitions. For details, see "Backing up partitioned databases" in *Data Recovery and High Availability Guide and Reference*.

   If you activated and configured Db2 Advanced Copy Services (ACS) on your databases in Db2 Version 9.7 or later, you can use the **USE SNAPSHOT** parameter to perform a snapshot backup. However, you can only restore a snapshot backup to an instance of the same version. You cannot use snapshot backup to upgrade to a new server. For details, see Performing a snapshot backup in *Data Recovery and High Availability Guide and Reference*.

For offline backup: If you performed a full online or offline database backup recently and you cannot perform another one before upgrading, you can perform an incremental offline database backup instead

3. Optional: Test the integrity of a backup image to ensure that the image can be restored using the **db2ckbkp** command. The following command is an example on UNIX operating systems:

   ```
   cd backup-dir
   db2ckbkp SAMPLE.0.arada.NODE0000.CATN0000.20091014114322.001

   [1] Buffers processed:  #######

   Image Verification Complete - successful.
   ```

4. Optional: Check the integrity of the log files needed to recover from the backup image used in step 3. For details, see Checking archive log files with the **db2cklog** tool in *Data Recovery and High Availability Guide and Reference*.

Backing up Db2 server configuration and diagnostic information

Backing up your settings for database and database manager configuration parameters before Db2 server upgrade, or conversion to a Db2 pureScale environment, allows you to verify Db2 server behavior after upgrade, or converting to Db2 pureScale environment, and to recreate instances and databases.

In addition, you can collect information from your Db2 servers about the database system catalogs, Db2 registry variables settings, explain table data, and diagnostic information. This information can help in problem determination if you encounter any post-upgrade or any post-conversion to a Db2 pureScale environment, differences in the database manager behavior or performance.

Before you begin

You must have SYSADM authority in order to execute all of the following tasks, although some tasks require lesser authority privileges or none.

Procedure

To back up your Db2 server configuration and diagnostic information:

1. Collect information from your Db2 servers by running the **db2support** command for all your databases that you are going to upgrade, or convert to Db2 pureScale, in all your instances. This command allows you to collect information about the database system catalog, database and database manager configuration parameters settings, Db2 registry variables settings, explain table data, and diagnostic information required by Db2 support in case of problems.

   ```
   db2support output-directory -d database-name -cl 0
   ```

   The **-cl 0** parameter collects the database system catalog, database and database manager configuration parameters settings, Db2 registry variables settings. The information collected is stored in the `db2support.zip` compressed zip file under the output directory. A summary report in HTML format is included. In the `db2supp_opt.zip` file that is also included, you should check the `optimizer.log` file to verify that the collection of information was performed successfully.

   Keep this zip file for several months after you complete the upgrade, or conversion to Db2 pureScale. The information in the zip file can help in quickly resolving any performance issues with the new release.

2. Back up the information about all the packages for your applications associated with each database. Use the following command to list packages associated with your databases and redirect the command output to a file:

   ```
   db2 LIST PACKAGES FOR SCHEMA schema-name
         SHOW DETAIL > /upgrade/sample_pckg.txt
   ```

   The FOR SCHEMA clause allows you to list all packages for a specific schema, if your application has several schemas you need to repeat this command for each schema name or use the FOR ALL clause.

   ```
   db2 LIST PACKAGES FOR ALL
         SHOW DETAIL > /upgrade/sample_pckg.txt
   ```

3. If you enabled the audit facility, back up the audit configuration of your instances by issuing the following command:

   ```
   db2audit describe > audit_instance-name.cfg
   ```

   If you have multiple instances, repeat this command for each instance.
4. Back up all your external routines. See "Backup and restore of external routine library and class files" in *Administrative Routines and Views*. The following example shows how to backup all external routines created using the default path in UNIX operating systems:

   ```
   cp -R $INSTHOME/sqllib/function $INSTHOME/routine_backup
   ```

   Where *INSTHOME* is set to the home directory of the instance owner. If you have specified a full path that is not under the default routines path when you created your external routines in the database, you must ensure the existing libraries remain on their original location.
5. Optional: The **db2support** command HTML report includes the database manager configuration parameter settings for the instance that owns the specified database. You can use the **GET DATABASE MANAGER CONFIGURATION** command to back up your settings for database manager configuration parameters and redirect the command output to a file to save these settings for each instance:

   ```
   db2 GET DBM CFG > dbm_instname.cfg
   ```

   where *instname* is the instance name.
6. Optional: The **db2support** command HTML report includes the database configuration parameter settings for the specified database. You can use the **GET DATABASE CONFIGURATION** command to back up your settings for database configuration parameters and redirect the command output to a file to save these settings for each database:

   ```
   db2 CONNECT TO database_alias
   db2 GET DB CFG FOR database_alias
         SHOW DETAIL > db_database_alias.cfg
   ```

   where *database_alias* is the database alias. The **SHOW DETAIL** clause displays the values calculated by the database manager when configuration parameters are set to AUTOMATIC.

   Database configuration parameters can be the same on each database partition in a partitioned database environment. If they are not the same, back up the database configuration parameter settings for each database partition.
7. Optional: The **db2support** command generates a file with the output of the **db2look** command for the specified database. However if you need additional information not present in the generated DDL file, you can use this command to save the DDL information for your databases and the statements to re-create your database objects:

   ```
   db2look -d sample -e -o sample_tbs.db2 -l -x
   ```

8. Optional: The **db2support** command HTML report includes the environment and registry variable settings for the instance that owns the specified database. You can use the **db2set** command to back up your Db2 profile registry variables settings and redirect the command output to a file to save these settings:

   ```
   db2set -all > reg_instname.txt
   ```

   If you set Db2 environment variables, use the appropriate system command to list environment variables and their values. For example, on AIX® you can issue the following command:

   ```
   set |grep DB2 > env_instname.txt
   ```

   When possible, use the output from the set command and run the **db2set** command to set these environment variables as registry variables in the Db2 profile registry.

What to do next

Take your server offline before converting to a Db2 pureScale environment.

Increasing table space and log file sizes before upgrade

Before you start upgrading your Db2 server, you must ensure that you have a sufficient amount of free space on your system catalog table space and temporary table space, and enough log space to upgrade your databases.

Before you begin

Ensure that you have SYSCTRL or SYSADM authority to be able to increase the size of table spaces and log space.

About this task

Additional considerations are required in partitioned database environments to increase table space sizes because table spaces span across database partitions. Also, you only need to increase the log space in the catalog database partition server.

Procedure

To increase the size of your table spaces and log space:

1. Connect to the database you want to upgrade:

   ```
   db2 CONNECT TO sample
   ```

2. Determine your table space disk usage by issuing the following query:

   ```
   db2 "SELECT SUBSTR(TBSP_NAME,1,15) NAME, TBSP_TYPE TYPE,
        TBSP_AUTO_RESIZE_ENABLED AUTO_RESIZE, TBSP_NUM_CONTAINERS CONTAINERS,
        TBSP_TOTAL_PAGES TOTAL_PGS, TBSP_USED_PAGES USED_PGS, TBSP_FREE_PAGES FREE_PGS,
        TBSP_MAX_SIZE MAX_SZ, TBSP_PAGE_SIZE PG_SZ
        FROM SYSIBMADM.TBSP_UTILIZATION
        WHERE TBSP_CONTENT_TYPE IN ('ANY','SYSTEMP')"

   NAME            TYPE AUTO_RESIZE CONTAINERS TOTAL_PGS USED_PGS FREE_PGS MAX_SZ PG_SZ
   --------------- ---- ----------- ---------- --------- -------- -------- ------ -----
   SYSCATSPACE     DMS            1          1      8192     7576      612     -1  8192
   TEMPSPACE1      SMS            -          1        10       10        0      -  8192

     2 record(s) selected.
   ```

   Take note of the number of containers, total pages, used pages, free pages, MAXSIZE, and page size.

3. Increase the size of the system catalog table spaces using one of the following options:
   - If you have an SMS table space, ensure that you have at least *the same amount of used pages available as free disk space*; in this example, about 60 MB.
   - If you have a DMS table space and the number of used pages is greater than the number of free pages, use the following formula to calculate the number of pages to increase per container:

     ```
     number_of_pages = ( used_pages - free_pages ) /
                         number_of_containers_in_SYSCATSPACE
     ```

     Then use the following command to increase the size of all containers in the system catalog table space:

     ```
     db2 "ALTER TABLESPACE SYSCATSPACE EXTEND (ALL number_of_pages)"
     ```
   - If you have a DMS table space with AUTORESIZE enabled and MAXSIZE is set to NONE, ensure that you have at least *twice the amount of used pages* available in free disk space. If MAXSIZE is set to an integer value that is less than twice the amount of used pages, then you need to increase MAXSIZE using the ALTER TABLESPACE statement as shown in the following example:

     ```
     db2 "ALTER TABLESPACE SYSCATSPACE
               MAXSIZE (2*used_pages_in_SYSCATSPACE*page_size/1024) K"
     ```

   In our example, the query results in the previous step shows that SYSCATSPACE is a DMS table space with AUTORESIZE enabled and a MAXSIZE value of -1 which indicates unlimited maximum size. Therefore, you must have twice the amount of used pages available in free disk space.
4. Increase the size of the temporary table spaces using one of the following options:
   - If you have an SMS table space you only need to ensure that you have at least twice the amount of total pages for the system catalog table space in free disk space; in this example, about 128 MB.
   - If you have a DMS table space, use the following formula to calculate the number of pages to increase per container:

     ```
     number_of_pages = ( number_of_total_pages_in_SYSCATSPACE  ) /
                         number_of_containers_in_TEMPSPACE1
     ```

     Use the following command to increase the size of all containers in the temporary table space:

     ```
     db2 "ALTER TABLESPACE TEMPSPACE1 EXTEND (ALL number_of_pages)"
     ```
   - If you have a DMS table space with AUTORESIZE enabled and MAXSIZE is set to NONE, ensure that you have at least twice the amount of total pages for the system catalog table space in free disk space. If MAXSIZE is set to an integer value that is less than twice the amount of total pages for the system catalog table space, then you need to increase MAXSIZE using the ALTER TABLESPACE statement:

     ```
     db2 "ALTER TABLESPACE TEMPSPACE1
               MAXSIZE (2*total_pages_in_SYSCATSPACE*page_size/1024) K"
     ```
5. Determine the current log space size using the GET DATABASE CONFIGURATION command. The following example shows how to record the values for **logfilsiz**, **logprimary**, and **logsecond** database configuration parameters on Linux and UNIX operating systems:

```
db2 GET DB CFG FOR sample |grep '(LOG[FPS]'| tee logsize.txt
 Log file size (4KB)                         (LOGFILSIZ) = 1000
 Number of primary log files                (LOGPRIMARY) = 3
 Number of secondary log files               (LOGSECOND) = 2
```

6. Increase your log space size using the following commands:

   ```
   db2 UPDATE DB CFG FOR sample using LOGSECOND
       (current_value of LOGPRIMARY + current_value of LOGSECOND) * 2
   ```

   If you already have a large log space, you might not need to increase it.
7. Optional: Enable infinite active logging instead of increasing the log space, by setting **logsecond** to -1 and enabling archive logging. Infinite active logging allows an active unit of work to span the primary logs and archive logs, effectively allowing a transaction to use an infinite number of log files. You should be aware that if the upgrade fails, the time to roll back the transactions will depend on how many archived logs need to be retrieved. The following command shows an example on how to enable archive logging to disk and infinite logging:

   ```
   db2 UPDATE DB CFG FOR sample using LOGARCHMETH1 DISK:archive-dir
   db2 UPDATE DB CFG FOR sample using LOGSECOND -1
   ```

   where *archive-dir* is the directory to archive the log files.

   All applications must disconnect from this database before the new values become effective.

Changing raw devices to block devices (Linux)

Changing raw (character) devices to block devices on Linux operating systems is required before you upgrade to .

The previous raw I/O method that required binding the block device to a raw (character) device using the raw utility is deprecated since Db2 Version 9.1, and will be removed in a future release of Db2 database product. This raw I/O method is also deprecated in the Linux operating system and will be removed in a future release of Linux.

The block device method uses Direct I/O to achieve an equivalent performance compared to using the raw (character) device method.

Before you begin

Ensure the database is offline in order to relocate the containers.

Restrictions

In a partitioned database environment, the **db2relocatedb** command must be run against every database partition that requires changes. A different configuration file must be supplied for each database partition, and must include the NODENUM value of the database partition being changed.

If you are restoring from a pre-Version 9.7 backup in Db2 Version 9.7, you must do a redirected restore to indicate block devices instead of raw character devices for your containers.

Procedure

1. Perform a full offline backup of your database.

2. Shut down your database. Also consider putting the database in quiesce mode using the **QUIESCE DATABASE** command as shown in the following example:
   ```
   db2 CONNECT TO sample
   db2 QUIESCE DATABASE DEFER FORCE CONNECTIONS
   db2 DEACTIVATE DATABASE database-alias
   ```
3. Use the **raw -a** system command to see which raw bindings you defined. This information will help you determine the block device you should use to replace a raw device for each container on your table spaces.
4. Create a configuration file for the **db2relocatedb** command. Use the clause **CONT_PATH** to specify the old value with the new value. For example, you can create the `moveraw.cfg` file with the following content:
   ```
   DB_NAME=SAMPLE
   DB_PATH=/databases/SAMPLE
   INSTANCE=db2inst1
   NODENUM=0
   CONT_PATH=/dev/raw/raw1,/dev/sda1
   CONT_PATH=/dev/raw/raw2,/dev/sda2
   ```
5. Execute the **db2relocatedb** command to change the configuration of the database files as shown in the following example:
   ```
   db2relocatedb -f moveraw.cfg
   ```
6. Activate your database as shown in the following example:
   ```
   db2 ACTIVATE DATABASE database-alias
   ```
7. Test that your database is functioning as expected. Connect to the database and execute queries on tables created on the table spaces that you relocated.
8. If you put the database in quiesce mode, you can restore the access and activate the database using the **UNQUIESCE DATABASE** command as shown in the following example:
   ```
   db2 CONNECT TO sample
   db2 UNQUIESCE DATABASE
   ```

Gathering pre-upgrade diagnostic information

Before creating or upgrading an instance and before updating to the next fix pack, you might need to gather diagnostic information to help troubleshoot any problem that might come up after the upgrade or update.

Before you begin

Some of the collections that are performed will take a long time to complete. Please have a sufficient amount of time before your scheduled upgrade or update to complete the collection of the diagnostic information.

About this task

If you plan to create or upgrade an instance, or update to the next available fix pack, it is helpful to gather performance, configuration, and environment information to help diagnose any future problems that might arise after you perform the upgrade or update. The gathering of this diagnostic information is done through the **db2fodc -preupgade** and **db2support -preupgrade** commands.

Restrictions

You must be using Version 9.7 Fixpack 5 or later to use the **db2fodc -preupgade** and **db2support -preupgrade** commands.

Procedure

To gather a sufficient amount of information to diagnose any future problems that might arise when performing an upgrade or update, you need to perform the following steps:

1. Issue the **`db2fodc -preupgrade -db`** *database_name* command at high usage and idle times.

   This command collects performance related information that might be needed for future problems. After collection is completed the information is stored in a newly created directory named `FODC_Preupgrade_<`*`timestamp`*`>_<`*`member`*`>`.

   Note: To gather better performance information, issue the **`db2fodc -preupgrade`** command multiple times at different usage levels. This gives IBM support a more complete picture of the performance of Db2.
2. Issue the **`db2support -preupgrade -d`** *database_name* command.

   This command collects configuration and environment information, and information from the FODC preupgrade directories created previously.

Results

After collection is completed a `db2support_preupgrade.zip` file which contains all the collected information is created in the current directory.

What to do next

If any problems arise after the upgrade or update you might be required to send the `db2support_preupgrade.zip` file to IBM support for analysis. The `db2support_preupgrade.zip` file must be kept until it is determined that the upgrade or update is functioning normally.

Upgrading Db2 servers in a test environment

Upgrading Db2 servers in a test environment before you upgrade them in your production environment allows you to address any problems during the upgrade process more effectively and to evaluate the impact of changes introduced in Db2 Version 11.1.

You can also verify that applications, scripts, tools and maintenance procedures work properly before upgrading your production environment. In addition, you can assess the disk requirements and the time that it takes to upgrade the database, to solidify your upgrade plan.

Before you begin

You must have root user authority on Linux and UNIX operating systems or Local Administrator authority on Windows. You must also have SYSADM authority.

Procedure

To duplicate your production environment in a test environment, perform the following tasks:

1. Install Db2 Version 10.1or earlier. If you already have a Db2 copy, you do not have to create a new one.
2. Create your instance duplicates as test instances.

3. Perform the steps in “Creating database duplicates” in the testing instances. You can duplicate your databases without data to test only database upgrade or using a data subset to test all your application functionality. Database upgrade converts only system catalog objects. Therefore, the volume of data in the tables does not impact the disk requirements or the time that it takes to upgrade the database.
4. Perform the pre-upgrade tasks that apply to your Db2 server.
5. Install Db2 Version 11.1.
6. Perform the steps in “Upgrading Db2 Version 10.5 or Db2 Version 10.1 instances” on page 45.
7. Perform the steps in “Upgrading databases” on page 49. Keep a record of the time it takes to upgrade each database and the size of the system catalog table space, system temporary table space, and log space. The following example shows how to do this on an AIX operating system:

   ```
   time db2 UPGRADE DATABASE nsample | tee upgrade_time.log
   db2 connect to nsample
   db2 "SELECT SUBSTR(TBSP_NAME,1,15) NAME, MEMBER, TBSP_TYPE TYPE,
          TBSP_AUTO_RESIZE_ENABLED AUTO_RESIZE, TBSP_TOTAL_PAGES TOTAL_PGS,
          TBSP_USED_PAGES USED_PGS, TBSP_FREE_PAGES FREE_PGS,
          TBSP_PAGE_SIZE PG_SZ, TBSP_EXTENT_SIZE EXTENT_SZ,
          TBSP_PREFETCH_SIZE PREFETCH_SZ, TBSP_NUM_CONTAINERS CONTAINERS
        FROM TABLE(MON_GET_TABLESPACE(NULL,-2)) AS T
        WHERE TBSP_CONTENT_TYPE IN ('ANY','SYSTEMP')" | tee tbs_details.log
   db2 GET DB CFG FOR nsample | grep '(LOG[FPS]' | tee log_size.log
   ```

   Use this information in your upgrade plan.
8. If you found any issues upgrading your test databases, find a resolution to these issues before upgrading your production environment. Add the tasks to resolve these issues to your upgrade plan.
9. Perform the steps in “Post-upgrade tasks for Db2 servers” on page 102 that apply to your Db2 server.
10. Perform the steps in “Verifying upgrade of Db2 servers” on page 110 to ensure the upgrade was successful.
11. Test your applications, scripts, tools and maintenance procedures by connecting to the test databases that you upgraded to the Db2 Version 11.1 copy if your test databases are populated with data.

Creating database duplicates:

Creating production database duplicates in a test environment allows you to test upgrading your databases before you upgrade them in your production environment.

Before you begin

Ensure that you have SYSCTRL or SYSADM authority.

About this task

This procedure uses DDL scripts to create database duplicates. If you have enough resources, you can also create database duplicates by restoring a database backup to create a new database. See “Restoring to a new database” in *Data Recovery and High Availability Guide and Reference* for details.

Procedure

To create a database duplicate for testing database upgrade:

1. Log on as the instance owner on the production database server and use the **db2look** command to generate DDL scripts with all the existing objects in your databases. The following command shows how to generate the `sample.ddl` script for the SAMPLE database:

   ```
   db2look -d sample -a -e -m -l -x -f -o sample.ddl
   ```

   Edit the generated DDL scripts and change:
   - The database name in the CONNECT statements
   - The path of the user table space containers or data and reduce the sizes to a minimum size since to re-create a database with no data or just a data subset

   You can use your own DDL scripts to create test databases in the test instance instead of generating DDL scripts.
2. Log on as the instance owner in the test database server and create your database duplicates. The following example shows how to create a database duplicate of the SAMPLE database using the `sample.ddl` script:

   ```
   db2 CREATE DATABASE NSAMPLE
   db2 -tvsf sample.ddl
   db2 UPDATE DBM CONFIGURATION USING diaglevel 4
   ```

   All significant upgrade events are logged in the **db2diag** log files when the **diaglevel** database manager configuration parameter is set to 3 (default value) or higher. A value of 4 captures additional information that can be helpful in problem determination.
3. Adjust the size of the system catalog table space, temporary table space, and log space in your test databases if required. Refer to "Increasing table space and log file sizes before upgrade" on page 36.
4. Export data subsets of your production databases and import these data subsets into your test databases. For details, see "Exporting Data" and "Importing Data" in *Data Movement Utilities Guide and Reference*. You only need a data subset if you are going to test your applications in your testing environment.
5. Verify that your database duplicates were created successfully by connecting to the them and issue a small query.

Taking a Db2 server offline for upgrade or for converting to a Db2 pureScale environment

Before you can continue with the upgrade process, or the conversion of your environment for Db2 pureScale, you must take your Db2 server offline by stopping the Db2 license service, stopping all command line processor sessions, disconnecting applications and users, and stopping the database manager.

Before you begin

You must have SYSADM authority.

Procedure

To take your Db2 server offline:

1. Stop the Db2 license service:

   ```
   db2licd -end
   ```

2. Disconnect all applications and users. To get a list of all database connections for the current instance, issue the **LIST APPLICATIONS** command. If all applications are disconnected, this command returns the following message:

   ```
   db2 list applications
     SQL1611W No data was returned by the Database System Monitor.
     SQLSTATE=00000
   ```

   To disconnect all applications and users, use the **FORCE APPLICATION** command:

   ```
   db2 force application all
   ```
3. Stop all command line processor sessions by entering the following command in each session that was running the command line processor.

   ```
   db2 terminate
   ```
4. When all applications and users are disconnected, stop the database manager instance:

   ```
   db2stop
   ```

What to do next

Convert your existing Db2 instances to a Db2 pureScale environment.

Upgrading a Db2 server (Windows)

Upgrading a Db2 server on Windows to Version 11.1 requires that you install a new Version 11.1 copy and then upgrade your existing instances and databases to this new copy.

If you choose to automatically upgrade your existing pre- Version 11.1 copy during the Version 11.1 installation, your instances and Db2 administration server (DAS) are upgraded but you still need to upgrade your databases after installation. If you choose to install a new Version 11.1 copy, you must manually upgrade your instances, your DAS, and databases.

This upgrade task describes the steps for direct upgrade to Db2 Version 11.1 from Version 10.1 or earlier. Review the steps in upgrading environments with specific characteristics and determine which task applies better to your environment.

Before you begin

- Ensure that you have Local Administrator authority. See the Prerequisites section in "Installing Db2 servers (Windows)" in *Installing Db2 Servers* for additional authorization details.
- Ensure that you meet the installation requirements for Db2 database products. Refer to "Installation requirements for Db2 database products" in *Installing Db2 Servers*.
- Review upgrade recommendations and disk space requirements. Refer to "Best practices for upgrading Db2 servers" on page 23 and "Disk space requirements for Db2 server upgrades" on page 21.
- Perform pre-upgrade tasks. Refer to "Pre-upgrade tasks for Db2 servers" on page 27.

Restrictions

- This procedure applies only to upgrade from Db2 32-bit servers when you install the Version 11.1 32-bit database product or from 64-bit servers when you install the Version 11.1 64-bit database product. The instance bit size is determined by

the operating system and the Version 11.1 database product that you install, see "Support changes for 32-bit and 64-bit Db2 servers" on page 22 for details.
- Additional upgrade restrictions apply. Refer to "Upgrade restrictions for Db2 servers" on page 14. Review the complete list.

Procedure

To upgrade a Db2 server to Db2 Version 11.1:

1. Log on to the Db2 server as a user with Local Administrator authority.
2. Install Db2 Version 11.1 by running the `setup` command to launch the Db2 Setup wizard. You have three choices:
   - To automatically upgrade a Db2 copy, all the instances running on the selected Db2 copy, and your DAS, select the **Work with Existing** option on the **Install a Product** panel. Then, in the **Work with Existing** window, choose the Db2 copy name with the **upgrade** action. The selected Db2 copy and add-on products are uninstalled.

     You will get a warning that recommends that you run the `db2ckupgrade` command if you have local databases. If you completed the pre-upgrade tasks, ignore this warning and continue the upgrade. Otherwise, verify that your databases are ready for Db2 upgrade before continuing with the installation. Refer to "Verifying that your databases are ready for upgrade" on page 29.
   - To create a new copy of Db2 Version 11.1, select the **Install New** option on the **Install a Product** panel.
   - To create a response file and perform a response file installation, select the **Work with Existing** option on the **Install a Product** panel. Then in the **Work with Existing** window, choose the Db2 copy name with the **upgrade** action. Finally, in the **Select the installation, response file creation, or both** window, select the **Save my installation setting in a response file** option to create a response file for a response file installation. The response file has the required UPGRADE_PRIOR_VERSIONS keyword, the Db2 copy name to upgrade, and the installation path.

     The result of the response file installation will be the same as in the first choice, all your instances running on the selected Db2 copy and your DAS are automatically upgraded to the Db2 Version 11.1 copy.
3. Install all Db2 add-on products that were installed in the Db2 copy from which you are upgrading.
4. If you installed a new copy of Db2 Version 11.1, upgrade your Db2 Version 10.1 or earlier instances to this new copy. Refer to "Upgrading Db2 Version 10.5 or Db2 Version 10.1 instances" on page 45.
5. Optional: If you installed a new copy, upgrade the DAS if you want to keep your existing DAS configuration and use new functionality available in Db2 Version 11.1. Refer to "Upgrading the Db2 Administration Server (DAS)" on page 47.
6. Upgrade your databases. Refer to "Upgrading databases" on page 49.

What to do next

After upgrading the Db2 server, perform the recommended post-upgrade tasks such as resetting the diagnostic error level to its pre-upgrade value, adjusting log space size, and rebinding packages. In addition, verify that the upgrade of your Db2 server was successful. Refer to "Post-upgrade tasks for Db2 servers" on page 102 and "Verifying upgrade of Db2 servers" on page 110.

Upgrading Db2 Version 10.5 or Db2 Version 10.1 instances

As part of the overall process of upgrading your Db2 database server to Db2 Version 11.1, you must upgrade your instances.

Before you begin

- You must have root user authority on Linux and UNIX operating systems or Local Administrator authority on Windows.
- You must install any Db2 database add-on products that were installed in the Db2 copy from which you are upgrading.
- Before you run the **db2iupgrade** command, do the following steps:
  - Verify that your databases are ready for Db2 upgrade. This step is important in partitioned database environments because the **db2ckupgrade** command might return an error in one database partition and cause the instance upgrade to fail. For HADR environments that support upgrade without the need for standby reinitialization (single partition ESE Db2 Version 10.5 Fix Pack 7 or later databases, or Db2 pureScale Version 10.5 Fix Pack 9 or later databases, ensure that both the primary's log shipping functionality and the standby's log replaying functionality is working properly. Refer to "Verifying that your databases are ready for upgrade" on page 29.
  - On Linux and UNIX operating systems, ensure that there is 5 GB of free space in the `/tmp` directory. The instance upgrade trace and log files are written to `/tmp`.
  - Gather pre-upgrade diagnostic information to help diagnose any problem that might occur after the upgrade.

About this task

On Linux and UNIX operating systems, you must manually upgrade your instances. On Windows operating systems, you must manually upgrade them if you did not choose to automatically upgrade your existing Db2 copy during the Db2 Version 11.1 installation.

Restriction

- On Linux and UNIX operating systems, you must not setup the instance environment for the root user. Running the **db2iupgrade** or the **db2icrt** command for a root user is not supported.
- For more restrictions on instance upgrade, review "Upgrade restrictions for Db2 servers" on page 14.
- You must be upgrading from Db2 Version 10.5 or earlier.

Procedure

To manually upgrade your existing instances to Db2 Version 11.1 using the **db2iupgrade** command:

1. Determine whether you can upgrade your existing instances to a Db2 Version 11.1 copy that you installed by do the following actions:
   - Determine the node type. The following examples show how to use the **GET DBM CFG** command from the command line to find out the node type:

Operating system	Examples

Linux and UNIX	`db2 GET DBM CFG | grep 'Node type'`
	`Node type = Enterprise Server Edition with local and remote clients`

Operating system	Examples

Windows	`db2 GET DBM CFG | find  "Node type"`
	`Node type = Enterprise Server Edition with local and remote clients`

- Review Table 8 on page 15 to determine the instance type by using the node type and whether instance upgrade is supported. In the previous example, the node type is “Enterprise Server Edition with local and remote clients” therefore the instance type is “ese” and you can upgrade only to a Db2 Version 11.1 copy of Db2 Server Edition.

If you cannot upgrade your instance to any Db2 Version 11.1 copy that you installed, you must install a copy of the Db2 Version 11.1 database product that supports upgrade of your instance type before you can proceed with the next step.

2. Disconnect all applications and users. To get a list of all database connections for the current instance, issue the **LIST APPLICATIONS** command. If all applications are disconnected, this command returns the following message:

```
db2 list applications
  SQL1611W No data was returned by the Database System Monitor.
  SQLSTATE=00000
```

   To disconnect all applications and users, use the **FORCE APPLICATION** command:

```
db2 force application all
```

3. Stop all command line processor sessions by entering the following command in each session that was running the command line processor.

```
db2 terminate
```

4. When all applications and users are disconnected, stop each database manager instance:

```
db2stop
```

5. Log on to the Db2 database server with root user authority on Linux and UNIX operating systems or Local Administrator authority on Windows operating systems.
6. Upgrade your existing instances by running the **db2iupgrade** command from the target Db2 Version 11.1 copy location. The **db2iupgrade** command must be run only on the instance owning node. The following table shows how to run the **db2iupgrade** command to upgrade your instances:

Operating system	Command syntax
Linux and UNIX	`$DB2DIR/instance/db2iupgrade [ -u fencedID ] InstName`[a]
Windows	`"%DB2PATH%"\bin\db2iupgrade InstName /u:user,password`[b]

Note:

a. Where *DB2DIR* is set to the location you specified during Db2 Version 11.1 installation, *fencedID* is the user name under which the fenced user-defined functions (UDFs) and stored procedures run, and *InstName* is the login name of the instance owner. This example upgrades the instance to the Db2 database product from where **db2iupgrade** command is run. Use the **-k** option if you want to keep the pre-upgrade instance type (for example wse, ese).

b. Where **DB2PATH** is set to the location you specified during Db2 Version 11.1 installation, *user*, and *password* are the user name and password under which the Db2 service runs, and *InstName* is the name of the instance.

If you did not install all Db2 database add-on products that were installed in the Db2 copy from which you are upgrading, the instance upgrade fails and returns a warning message. If you plan to install these products later on or you no longer need the functionality that is provided by these products, use the **-F** parameter to upgrade the instance.

The **db2iupgrade** command calls the **db2ckupgrade** commandto verify that the local databases are ready for upgrade. The `update.log` is specified as the log file for **db2ckupgrade**, and the default log file that is created for **db2iupgrade** is `/tmp/db2ckupgrade.log.processID`. On Linux and UNIX operating systems, the log file is created in the instance home directory. On Windows operating systems, the log file is created in the current directory where you are running the **db2iupgrade** command. The **db2iupgrade** does not run when the **db2ckupgrade** command reports errors. Check the log file if you encounter any errors.

7. Ensure that the standby node is updated with the current instance information by performing the following actions:
   a. Remove the old instance information from the standby database by issuing the following command on the standby node: *new db2 upgrade path*/bin/**db2greg -delinstrec instancename=***old_instance_name*
   b. Add the new instance information to the standby database by issuing the following command on the standby node: *new db2 upgrade path*/instance/**db2iset -a** *new_instance_name*

   You can issue the previous commands on all nodes instead of just on the standby node.
8. Log on to the Db2 database server as a user with sufficient authority to start your instance.
9. Restart your instance by running the **db2start** command:
   ```
   db2start
   ```
10. Verify that your instance is running on to Db2 Version 11.1 by running the **db2level** command:
    ```
    db2level
    ```

    The Informational tokens must include a string like "Db2 Version 11.1.*X*.*X*" where *X* is a digit number.

Upgrading the Db2 Administration Server (DAS)

Upgrading your Db2 Administration Server (DAS) is only necessary to keep your existing DAS configuration.

Otherwise, you can drop your existing DAS and create a new DAS in Db2 Version 11.1. See "Creating a Db2 administration server (DAS) " in Installing Db2 Servers.

Start using IBM Data Studio and IBM Optim™ tools. For a mapping between these recommended tools and Control Center tools, see "Table of recommended tools versus Control Center tools" in the *What's New for Db2 Version 10.5* book.

Important: The Db2 Administration Server (DAS) has been deprecated in Version 9.7 and might be removed in a future release. The DAS is not supported in Db2 pureScale environments. Use software programs that use the Secure Shell protocol for remote administration. For more information, see "Db2 administration server (DAS) has been deprecated" at .

Before you begin

- Ensure that you have SYSADM authority, and root access on Linux and UNIX operating systems or Local Administrator authority on Windows operating systems.

Restrictions

- You can have only one DAS per computer.

Procedure

To upgrade the DAS:

1. Log on to the Db2 server as root on Linux and UNIX operating systems or Local Administrator authority on Windows.
2. Upgrade your existing DAS by running the **dasmigr** command:

Operating system	Command syntax
Linux and UNIX	`$`*DB2DIR*`/instance/dasmigr`
Windows	`%`**DB2PATH**`%\bin\dasmigr`

   Where *DB2DIR* and **DB2PATH** indicate the location that you specified during Db2 Version 11.1 installation.

   If the DAS is running, the **dasmigr** command stops the DAS before upgrade and starts the DAS after upgrade.
3. If you created a tools catalog database and want to use your existing scripts and schedules in Db2 Version 11.1, perform the following steps:
   - Upgrade the instance that owns the tools catalog database. For details, see “Upgrading Db2 Version 10.5 or Db2 Version 10.1 instances” on page 45.
   - Upgrade the tools catalog database. For details, see “Upgrading databases” on page 49
   - Verify that the DAS is configured to access the upgraded tools catalog database by running the **GET ADMIN CFG** command to display the current configuration settings for the tools catalog database:

```
db2 GET ADMIN CFG

         Admin Server Configuration
 ...
 Tools Catalog Database                    (TOOLSCAT_DB) = toolsdb
 Tools Catalog Database Instance         (TOOLSCAT_INST) = db2inst1
 Tools Catalog Database Schema         (TOOLSCAT_SCHEMA) = cc
 Scheduler User ID                                       =
```

     Use the **UPDATE ADMIN CFG** command if you must change any configuration settings for the tools catalog database.

   You should upgrade your tools catalog whether you decide to upgrade your DAS or not.
4. If you do not upgrade or do not have a tools catalog database, you can create one in a Db2 Version 11.1 instance to use the task scheduling capability. See “CREATE TOOLS CATALOG command ” in *Command Reference*.

Results

You can now use the DAS to administer Db2 Version 11.1 instances, as well as pre-Db2 Version 11.1 instances.

Upgrading databases

After you upgraded your instances to Db2 Version 11.1, you must upgrade each database under each instance.

Before you begin

- Ensure that you have SYSADM authority.
- Ensure that all the local databases that you want to upgrade are cataloged.
- Ensure that you backed up your databases as indicated in "Pre-upgrade tasks for Db2 servers" on page 27.
- Ensure that you installed Db2 Version 11.1 and upgraded the instance to Db2 Version 11.1.

Restrictions

- Review the steps in "Upgrade restrictions for Db2 servers" on page 14 for database upgrade.

Procedure

To upgrade a Db2 database to Db2 Version 11.1:

1. Log on to the Db2 server as the instance owner or a user with SYSADM authority.
2. Optional: Rename or delete the **db2diag** log files so that new files are created. Also, remove or move to another directory any existing dump files, trap files, and alert log files in the directory indicated by the **diagpath** parameter. By doing this, the files only contain information about the upgrade process that helps you to isolate and understand any problem that might occur during database upgrade.
3. Optional: Issue the **db2 LIST DATABASE DIRECTORY** command to ensure that the database is in the list of all catalogued databases in the current instance.
4. Upgrade the database using the **UPGRADE DATABASE** command:

   ```
   db2 UPGRADE DATABASE database-alias USER username USING password
   ```

   where *database-alias* is the name or the alias of the database you want to upgrade and the username and password to authenticate a user with SYSADM authority.

   To avoid the overhead of an automatic rebind, consider using the **REBINDALL** parameter, which specifies that a **REBIND** of all packages is performed during upgrade.
5. If the **UPGRADE DATABASE** command fails and returns the SQL1704N error message with a reason code that describes the cause of the failure, find this SQL error code and determine the action to take from the list of the possible solutions for each reason code. One of the most common causes of upgrade failure is that the log file space is not large enough, in which case the following error is returned:

   ```
   SQL1704N  Database upgrade failed.  Reason code "3".
   ```

   You must increase log file size and execute the **UPGRADE DATABASE** command again. For details, see "Increasing table space and log file sizes before upgrade" on page 36. After the database upgrade is complete reset the value of **logfilsiz**, **logprimary** and **logsecond** database configuration parameters.

There are additional error codes that are returned by the **UPGRADE DATABASE** command for specific cases that are not supported by database upgrade. These cases are described in "Upgrade restrictions for Db2 servers" on page 14.

6. If the **UPGRADE DATABASE** command returns the SQL1243W warning message, you must drop or rename the SYSTOOLS.DB2LOOK_INFO table. Otherwise, the ALTER TABLE and COPY SCHEMA statements will fail to run. Check if the SYSTOOLS.DB2LOOK_INFO table exists by running the following command:

   ```
   db2 "SELECT tabname, tabschema, definer FROM syscat.tables
          WHERE tabschema = 'SYSTOOLS' AND tabname = 'DB2LOOK_INFO'"
   ```

   If you created this table, rename it by running the RENAME statement:

   ```
   db2 RENAME SYSTOOLS.DB2LOOK_INFO TO new-table-name
   ```

   If you did not create this table, remove it by running the DROP command:

   ```
   db2 DROP TABLE SYSTOOLS.DB2LOOK_INFO
   ```
7. If the **UPGRADE DATABASE** command returns the SQL1499W warning message and writes the ADM7535W warning message with all the details to the administration notification log, then the command failed to refresh the table space attributes in the catalog table. However, the database was upgraded successfully. However, the database was upgraded successfully.
8. If the **UPGRADE DATABASE** command returns the SQL1499W warning message and writes the ADM4003E warning message with all the details to the administration notification log, then the command failed to upgrade the Db2 Text Search catalogs or indexes due to an error in a stored procedure.
9. If the **UPGRADE DATABASE** command returns the SQL1499W warning message and writes the ADM7534W warning message with all the details to the administration notification log, then the command failed to refresh the table space attributes in the catalog table. However, the database was upgraded successfully. However, the database was upgraded successfully.
10. If the **UPGRADE DATABASE** command returns the SQL1499W warning message and writes the ADM4101W warning message to the administration notification log, take note of the system catalog tables reported in the ADM4101W message so that you collect statistics on these tables as part of the post-upgrade tasks.
11. If the **UPGRADE DATABASE** command returns the SQL1499W warning message and writes the ADM4102W warning message to the administration notification log, qualify or delimit with quotes the identifiers called NULL in your SQL statements to avoid conflict with the NULL keyword.

    If you use identifiers called NULL for column names, routine parameter names, or variable names in an SQL statement that are not fully qualified or delimited with quotes, the identifier name might resolve to the NULL keyword instead. This would result in a change in behavior from previous releases. Refer to "Upgrade essentials for database applications" on page 138 for details.
12. If the **UPGRADE DATABASE** command returns the SQL1499W warning message and writes the ADM9516W warning message to the administration notification log, verify that the **indexrec** configuration parameter is set to `RESTART` and issue the **RESTART DATABASE** command to rebuild indexes marked as invalid during database upgrade. Otherwise, index rebuild starts on your first access to the table and you might experience an unexpected degradation in response time.

13. If the **UPGRADE DATABASE** command returns the SQL0473N error message, you must reverse the database migration and re-create all user-defined data types that use a system built-in data type name with a different name that is not restricted. See "Reversing Db2 server upgrade" on page 120.

    To avoid the **UPGRADE DATABASE** command failure, re-create these user-defined data types during "Verifying that your databases are ready for upgrade" on page 29.
14. If the **UPGRADE DATABASE** command returns the DBT5512 error message, the command failed to upgrade the database because the ID of a workload management object conflicts with a system-reserved ID. To upgrade the database, perform the following actions:
    a. Generate the DDL statements to re-create the workload management objects by issuing the **db2look** command with the **-wlm** parameter.
    b. Drop all of the workload management objects from the database.
    c. Resolve all issues that are reported by the **db2ckupgrade** command and block the database from being upgraded.
    d. Upgrade the database.
    e. Re-create the workload management object in the upgraded database by issuing the DDL statements that t you generated with the **db2look** command.
15. If the **UPGRADE DATABASE** command returns the SQL1700N error message, you must reverse the database migration and re-create database objects that use restricted schema names with a schema name that is not restricted. See "Reversing Db2 server upgrade" on page 120.

    To avoid the **UPGRADE DATABASE** command failure, re-create these database objects during "Verifying that your databases are ready for upgrade" on page 29.
16. If the **UPGRADE DATABASE** command returns the ADM4003E error message, then upgrade the Db2 Text Search catalog and indexes manually. For details, see **SYSTS_UPGRADE_CATALOG** and **SYSTS_UPGRADE_INDEX**.
17. Compare your database configuration settings after upgrade with the configuration settings you had before you upgraded your database. Verify that the following settings and database information are the same:
    - Database configuration parameter settings
    - Table spaces information
    - Packages information for your applications only

    You need not check package information for system generated packages. The information about system generated packages can change after upgrade.
18. Verify that your database upgrade is successful. Connect to the upgraded databases and issue a small query:
    ```
    db2 connect to sample

      Database Connection Information

     Database server        = DB2/AIX64 10.1.0
     SQL authorization ID   = TESTDB2
     Local database alias   = SAMPLE

    db2 "select * from syscat.dbauth"
    ```

    Alternatively, if you have sample files that are installed, run the `testdata.db2` script:

```
cd samplefile-dir-clp
db2 connect to sample
db2 -tvf testdata.db2
```

where *samplefile-dir-clp* is `DB2DIR/samples/clp` on Linux and UNIX and `DB2DIR\samples\clp` on Windows, *DB2DIR* represents the location that is specified during Db2 Version 11.1 installation, and sample is the database name.

What to do next

After upgrading a Db2 database, perform the recommended post-upgrade tasks to ensure a successful database upgrade. See “Post-upgrade tasks for Db2 servers” on page 102.

Upgrading a Db2 server (Linux and UNIX)

Upgrading a Db2 server to Version 11.1 on Linux and UNIX requires that you install Version 11.1 copy to a new path and then manually upgrade your existing instances and databases to this new copy.

Before you begin

Before upgrading the Db2 server:
- Ensure that you have root access.
- Ensure that you meet the installation requirements for Db2 database products. Refer to “Installation requirements for Db2 database products” in *Installing Db2 Servers*.
- Review upgrade recommendations and disk space requirements. Refer to “Best practices for upgrading Db2 servers” on page 23 and “Disk space requirements for Db2 server upgrades” on page 21.
- Perform pre-upgrade tasks. Refer to “Pre-upgrade tasks for Db2 servers” on page 27.

About this task

This upgrade task describes the steps for direct upgrade to Db2 Version 11.1 from Version 10.1 or earlier regardless of the instance bit size. Review “Upgrading Db2 servers with specific characteristics” on page 63 and determine which task applies better to your environment.

Restrictions
- On Linux and UNIX operating systems except for Linux on x86, your existing 32-bit or 64-bit instances are upgraded to Db2 Version 11.1 64-bit instances. The operating system and Db2 Version 11.1 database product that you installed determines the instance bit size, see “Support changes for 32-bit and 64-bit Db2 servers” on page 22 for details.
- Additional upgrade restrictions apply. Refer to “Upgrade restrictions for Db2 servers” on page 14. Review the complete list.

Procedure

To upgrade a Db2 server to Db2 Version 11.1:
1. Log on to the Db2 server as root.

2. Install Db2 Version 11.1. See "Installing Db2 servers using the DB2 Setup wizard (Linux and UNIX)" in *Installing Db2 Servers*. Run the **db2setup** command and select the **Install New** option on the **Install a Product** panel to install a new copy of Db2 Version 11.1.
3. Install all Db2 add-on products that were installed in the Db2 copy from which you are upgrading.
4. Upgrade Db2 Version 10.1 or earlier instances from the same installation path that you indicated during the Version 11.1 installation. Refer to "Upgrading Db2 Version 10.5 or Db2 Version 10.1 instances" on page 45.
5. Optional: Upgrade your DAS if you want to keep your existing DAS configuration and use new functionality available in Db2 Version 11.1. Refer to "Upgrading the Db2 Administration Server (DAS)" on page 47.
6. Upgrade databases. Refer to "Upgrading databases" on page 49.

What to do next

After upgrading the Db2 server, perform the recommended "Post-upgrade tasks for Db2 servers" on page 102 such as resetting the diagnostic error level, adjusting log space size, and rebinding packages. In addition, verify that the upgrade of your Db2 server was successful.

Upgrading Db2 Version 10.5 or Db2 Version 10.1 instances

As part of the overall process of upgrading your Db2 database server to Db2 Version 11.1, you must upgrade your instances.

Before you begin

- You must have root user authority on Linux and UNIX operating systems or Local Administrator authority on Windows.
- You must install any Db2 database add-on products that were installed in the Db2 copy from which you are upgrading.
- Before you run the **db2iupgrade** command, do the following steps:
  - Verify that your databases are ready for Db2 upgrade. This step is important in partitioned database environments because the **db2ckupgrade** command might return an error in one database partition and cause the instance upgrade to fail. For HADR environments that support upgrade without the need for standby reinitialization (single partition ESE Db2 Version 10.5 Fix Pack 7 or later databases, or Db2 pureScale Version 10.5 Fix Pack 9 or later databases, ensure that both the primary's log shipping functionality and the standby's log replaying functionality is working properly. Refer to "Verifying that your databases are ready for upgrade" on page 29.
  - On Linux and UNIX operating systems, ensure that there is 5 GB of free space in the `/tmp` directory. The instance upgrade trace and log files are written to `/tmp`.
  - Gather pre-upgrade diagnostic information to help diagnose any problem that might occur after the upgrade.

About this task

On Linux and UNIX operating systems, you must manually upgrade your instances. On Windows operating systems, you must manually upgrade them if you did not choose to automatically upgrade your existing Db2 copy during the Db2 Version 11.1 installation.

Restriction

- On Linux and UNIX operating systems, you must not setup the instance environment for the root user. Running the **db2iupgrade** or the **db2icrt** command for a root user is not supported.
- For more restrictions on instance upgrade, review "Upgrade restrictions for Db2 servers" on page 14.
- You must be upgrading from Db2 Version 10.5 or earlier.

Procedure

To manually upgrade your existing instances to Db2 Version 11.1 using the **db2iupgrade** command:

1. Determine whether you can upgrade your existing instances to a Db2 Version 11.1 copy that you installed by do the following actions:
   - Determine the node type. The following examples show how to use the **GET DBM CFG** command from the command line to find out the node type:

Operating system	Examples

Linux and UNIX	`db2 GET DBM CFG | grep 'Node type'`
	`Node type = Enterprise Server Edition with local and remote clients`

Windows	`db2 GET DBM CFG | find  "Node type"`
	`Node type = Enterprise Server Edition with local and remote clients`

   - Review Table 8 on page 15 to determine the instance type by using the node type and whether instance upgrade is supported. In the previous example, the node type is "Enterprise Server Edition with local and remote clients" therefore the instance type is "ese" and you can upgrade only to a Db2 Version 11.1 copy of Db2 Server Edition.

   If you cannot upgrade your instance to any Db2 Version 11.1 copy that you installed, you must install a copy of the Db2 Version 11.1 database product that supports upgrade of your instance type before you can proceed with the next step.
2. Disconnect all applications and users. To get a list of all database connections for the current instance, issue the **LIST APPLICATIONS** command. If all applications are disconnected, this command returns the following message:
   ```
   db2 list applications
     SQL1611W No data was returned by the Database System Monitor.
     SQLSTATE=00000
   ```
   To disconnect all applications and users, use the **FORCE APPLICATION** command:
   ```
   db2 force application all
   ```
3. Stop all command line processor sessions by entering the following command in each session that was running the command line processor.
   ```
   db2 terminate
   ```
4. When all applications and users are disconnected, stop each database manager instance:
   ```
   db2stop
   ```
5. Log on to the Db2 database server with root user authority on Linux and UNIX operating systems or Local Administrator authority on Windows operating systems.
6. Upgrade your existing instances by running the **db2iupgrade** command from the target Db2 Version 11.1 copy location. The **db2iupgrade** command must be

run only on the instance owning node. The following table shows how to run the **db2iupgrade** command to upgrade your instances:

Operating system	Command syntax
Linux and UNIX	`$DB2DIR/instance/db2iupgrade [ -u fencedID ] InstName`[a]
Windows	`"%DB2PATH%"\bin\db2iupgrade InstName /u:user,password`[b]

Note:

a. Where *DB2DIR* is set to the location you specified during Db2 Version 11.1 installation, *fencedID* is the user name under which the fenced user-defined functions (UDFs) and stored procedures run, and *InstName* is the login name of the instance owner. This example upgrades the instance to the Db2 database product from where **db2iupgrade** command is run. Use the **-k** option if you want to keep the pre-upgrade instance type (for example wse, ese).

b. Where **DB2PATH** is set to the location you specified during Db2 Version 11.1 installation, *user*, and *password* are the user name and password under which the Db2 service runs, and *InstName* is the name of the instance.

If you did not install all Db2 database add-on products that were installed in the Db2 copy from which you are upgrading, the instance upgrade fails and returns a warning message. If you plan to install these products later on or you no longer need the functionality that is provided by these products, use the **-F** parameter to upgrade the instance.

The **db2iupgrade** command calls the **db2ckupgrade** commandto verify that the local databases are ready for upgrade. The `update.log` is specified as the log file for **db2ckupgrade**, and the default log file that is created for **db2iupgrade** is `/tmp/db2ckupgrade.log.processID`. On Linux and UNIX operating systems, the log file is created in the instance home directory. On Windows operating systems, the log file is created in the current directory where you are running the **db2iupgrade** command. The **db2iupgrade** does not run when the **db2ckupgrade** command reports errors. Check the log file if you encounter any errors.

7. Ensure that the standby node is updated with the current instance information by performing the following actions:
   a. Remove the old instance information from the standby database by issuing the following command on the standby node: *new db2 upgrade path*/bin/**db2greg -delinstrec instancename=***old_instance_name*
   b. Add the new instance information to the standby database by issuing the following command on the standby node: *new db2 upgrade path*/instance/**db2iset -a** *new_instance_name*

   You can issue the previous commands on all nodes instead of just on the standby node.
8. Log on to the Db2 database server as a user with sufficient authority to start your instance.
9. Restart your instance by running the **db2start** command:
   ```
   db2start
   ```
10. Verify that your instance is running on to Db2 Version 11.1 by running the **db2level** command:
   ```
   db2level
   ```

The Informational tokens must include a string like "Db2 Version 11.1.*X*.*X*" where *X* is a digit number.

Upgrading the Db2 Administration Server (DAS)

Upgrading your Db2 Administration Server (DAS) is only necessary to keep your existing DAS configuration.

Otherwise, you can drop your existing DAS and create a new DAS in Db2 Version 11.1. See “Creating a Db2 administration server (DAS) ” in Installing Db2 Servers.

Start using IBM Data Studio and IBM Optim tools. For a mapping between these recommended tools and Control Center tools, see “Table of recommended tools versus Control Center tools” in the *What's New for Db2 Version 10.5* book.

Important: The Db2 Administration Server (DAS) has been deprecated in Version 9.7 and might be removed in a future release. The DAS is not supported in Db2 pureScale environments. Use software programs that use the Secure Shell protocol for remote administration. For more information, see “Db2 administration server (DAS) has been deprecated” at .

Before you begin

- Ensure that you have SYSADM authority, and root access on Linux and UNIX operating systems or Local Administrator authority on Windows operating systems.

Restrictions

- You can have only one DAS per computer.

Procedure

To upgrade the DAS:

1. Log on to the Db2 server as root on Linux and UNIX operating systems or Local Administrator authority on Windows.
2. Upgrade your existing DAS by running the **dasmigr** command:

Operating system	Command syntax
Linux and UNIX	`$`*`DB2DIR`*`/instance/dasmigr`
Windows	`%`**`DB2PATH`**`%\bin\dasmigr`

   Where *DB2DIR* and **DB2PATH** indicate the location that you specified during Db2 Version 11.1 installation.

   If the DAS is running, the **dasmigr** command stops the DAS before upgrade and starts the DAS after upgrade.
3. If you created a tools catalog database and want to use your existing scripts and schedules in Db2 Version 11.1, perform the following steps:
   - Upgrade the instance that owns the tools catalog database. For details, see “Upgrading Db2 Version 10.5 or Db2 Version 10.1 instances” on page 45.
   - Upgrade the tools catalog database. For details, see “Upgrading databases” on page 49
   - Verify that the DAS is configured to access the upgraded tools catalog database by running the **GET ADMIN CFG** command to display the current configuration settings for the tools catalog database:

```
db2 GET ADMIN CFG

         Admin Server Configuration
 ...
 Tools Catalog Database                      (TOOLSCAT_DB) = toolsdb
 Tools Catalog Database Instance           (TOOLSCAT_INST) = db2inst1
 Tools Catalog Database Schema           (TOOLSCAT_SCHEMA) = cc
 Scheduler User ID                                         =
```

Use the **UPDATE ADMIN CFG** command if you must change any configuration settings for the tools catalog database.

You should upgrade your tools catalog whether you decide to upgrade your DAS or not.

4. If you do not upgrade or do not have a tools catalog database, you can create one in a Db2 Version 11.1 instance to use the task scheduling capability. See "CREATE TOOLS CATALOG command " in *Command Reference*.

Results

You can now use the DAS to administer Db2 Version 11.1 instances, as well as pre-Db2 Version 11.1 instances.

Upgrading databases

After you upgraded your instances to Db2 Version 11.1, you must upgrade each database under each instance.

Before you begin

- Ensure that you have SYSADM authority.
- Ensure that all the local databases that you want to upgrade are cataloged.
- Ensure that you backed up your databases as indicated in "Pre-upgrade tasks for Db2 servers" on page 27.
- Ensure that you installed Db2 Version 11.1 and upgraded the instance to Db2 Version 11.1.

Restrictions

- Review the steps in "Upgrade restrictions for Db2 servers" on page 14 for database upgrade.

Procedure

To upgrade a Db2 database to Db2 Version 11.1:

1. Log on to the Db2 server as the instance owner or a user with SYSADM authority.
2. Optional: Rename or delete the **db2diag** log files so that new files are created. Also, remove or move to another directory any existing dump files, trap files, and alert log files in the directory indicated by the **diagpath** parameter. By doing this, the files only contain information about the upgrade process that helps you to isolate and understand any problem that might occur during database upgrade.
3. Optional: Issue the **db2 LIST DATABASE DIRECTORY** command to ensure that the database is in the list of all catalogued databases in the current instance.
4. Upgrade the database using the **UPGRADE DATABASE** command:

   ```
   db2 UPGRADE DATABASE database-alias USER username USING password
   ```

where *database-alias* is the name or the alias of the database you want to upgrade and the username and password to authenticate a user with SYSADM authority.

To avoid the overhead of an automatic rebind, consider using the **REBINDALL** parameter, which specifies that a **REBIND** of all packages is performed during upgrade.

5. If the **UPGRADE DATABASE** command fails and returns the SQL1704N error message with a reason code that describes the cause of the failure, find this SQL error code and determine the action to take from the list of the possible solutions for each reason code. One of the most common causes of upgrade failure is that the log file space is not large enough, in which case the following error is returned:

   ```
   SQL1704N  Database upgrade failed.  Reason code "3".
   ```

   You must increase log file size and execute the **UPGRADE DATABASE** command again. For details, see "Increasing table space and log file sizes before upgrade" on page 36. After the database upgrade is complete reset the value of **logfilsiz**, **logprimary** and **logsecond** database configuration parameters.

   There are additional error codes that are returned by the **UPGRADE DATABASE** command for specific cases that are not supported by database upgrade. These cases are described in "Upgrade restrictions for Db2 servers" on page 14.
6. If the **UPGRADE DATABASE** command returns the SQL1243W warning message, you must drop or rename the SYSTOOLS.DB2LOOK_INFO table. Otherwise, the ALTER TABLE and COPY SCHEMA statements will fail to run. Check if the SYSTOOLS.DB2LOOK_INFO table exists by running the following command:

   ```
   db2 "SELECT tabname, tabschema, definer FROM syscat.tables
          WHERE tabschema = 'SYSTOOLS' AND tabname = 'DB2LOOK_INFO'"
   ```

   If you created this table, rename it by running the RENAME statement:

   ```
   db2 RENAME SYSTOOLS.DB2LOOK_INFO TO new-table-name
   ```

   If you did not create this table, remove it by running the DROP command:

   ```
   db2 DROP TABLE SYSTOOLS.DB2LOOK_INFO
   ```
7. If the **UPGRADE DATABASE** command returns the SQL1499W warning message and writes the ADM7535W warning message with all the details to the administration notification log, then the command failed to refresh the table space attributes in the catalog table. However, the database was upgraded successfully. However, the database was upgraded successfully.
8. If the **UPGRADE DATABASE** command returns the SQL1499W warning message and writes the ADM4003E warning message with all the details to the administration notification log, then the command failed to upgrade the Db2 Text Search catalogs or indexes due to an error in a stored procedure.
9. If the **UPGRADE DATABASE** command returns the SQL1499W warning message and writes the ADM7534W warning message with all the details to the administration notification log, then the command failed to refresh the table space attributes in the catalog table. However, the database was upgraded successfully. However, the database was upgraded successfully.
10. If the **UPGRADE DATABASE** command returns the SQL1499W warning message and writes the ADM4101W warning message to the administration notification

log, take note of the system catalog tables reported in the ADM4101W message so that you collect statistics on these tables as part of the post-upgrade tasks.

11. If the **UPGRADE DATABASE** command returns the SQL1499W warning message and writes the ADM4102W warning message to the administration notification log, qualify or delimit with quotes the identifiers called NULL in your SQL statements to avoid conflict with the NULL keyword.

    If you use identifiers called NULL for column names, routine parameter names, or variable names in an SQL statement that are not fully qualified or delimited with quotes, the identifier name might resolve to the NULL keyword instead. This would result in a change in behavior from previous releases. Refer to "Upgrade essentials for database applications" on page 138 for details.

12. If the **UPGRADE DATABASE** command returns the SQL1499W warning message and writes the ADM9516W warning message to the administration notification log, verify that the **indexrec** configuration parameter is set to RESTART and issue the **RESTART DATABASE** command to rebuild indexes marked as invalid during database upgrade. Otherwise, index rebuild starts on your first access to the table and you might experience an unexpected degradation in response time.

13. If the **UPGRADE DATABASE** command returns the SQL0473N error message, you must reverse the database migration and re-create all user-defined data types that use a system built-in data type name with a different name that is not restricted. See "Reversing Db2 server upgrade" on page 120.

    To avoid the **UPGRADE DATABASE** command failure, re-create these user-defined data types during "Verifying that your databases are ready for upgrade" on page 29.

14. If the **UPGRADE DATABASE** command returns the DBT5512 error message, the command failed to upgrade the database because the ID of a workload management object conflicts with a system-reserved ID. To upgrade the database, perform the following actions:
    a. Generate the DDL statements to re-create the workload management objects by issuing the **db2look** command with the **-wlm** parameter.
    b. Drop all of the workload management objects from the database.
    c. Resolve all issues that are reported by the **db2ckupgrade** command and block the database from being upgraded.
    d. Upgrade the database.
    e. Re-create the workload management object in the upgraded database by issuing the DDL statements that t you generated with the **db2look** command.

15. If the **UPGRADE DATABASE** command returns the SQL1700N error message, you must reverse the database migration and re-create database objects that use restricted schema names with a schema name that is not restricted. See "Reversing Db2 server upgrade" on page 120.

    To avoid the **UPGRADE DATABASE** command failure, re-create these database objects during "Verifying that your databases are ready for upgrade" on page 29.

16. If the **UPGRADE DATABASE** command returns the ADM4003E error message, then upgrade the Db2 Text Search catalog and indexes manually. For details, see **SYSTS_UPGRADE_CATALOG** and **SYSTS_UPGRADE_INDEX**.

17. Compare your database configuration settings after upgrade with the configuration settings you had before you upgraded your database. Verify that the following settings and database information are the same:
    - Database configuration parameter settings
    - Table spaces information
    - Packages information for your applications only

    You need not check package information for system generated packages. The information about system generated packages can change after upgrade.
18. Verify that your database upgrade is successful. Connect to the upgraded databases and issue a small query:

    ```
    db2 connect to sample

      Database Connection Information

     Database server        = DB2/AIX64 10.1.0
     SQL authorization ID   = TESTDB2
     Local database alias   = SAMPLE

    db2 "select * from syscat.dbauth"
    ```

    Alternatively, if you have sample files that are installed, run the `testdata.db2` script:

    ```
    cd samplefile-dir-clp
    db2 connect to sample
    db2 -tvf testdata.db2
    ```

    where *samplefile-dir-clp* is *DB2DIR*`/samples/clp` on Linux and UNIX and *DB2DIR*`\samples\clp` on Windows, *DB2DIR* represents the location that is specified during Db2 Version 11.1 installation, and sample is the database name.

What to do next

After upgrading a Db2 database, perform the recommended post-upgrade tasks to ensure a successful database upgrade. See “Post-upgrade tasks for Db2 servers” on page 102.

Upgrading a Db2 server to Db2 pureScale environment

Upgrading to Db2 pureScale environment on Linux and UNIX requires that you first upgrade to Db2 Version 11.1 and then convert to Db2 pureScale environment.

Before you begin

- Ensure that you have root access.
- Ensure that you meet the installation requirements for Db2 database products. Refer to “Installation requirements for Db2 database products” in *Installing Db2 Servers*.
- Review upgrade recommendations and disk space requirements. Refer to “Best practices for upgrading Db2 servers” on page 23 and “Disk space requirements for Db2 server upgrades” on page 21.
- Perform pre-upgrade tasks. Refer to “Pre-upgrade tasks for Db2 servers” on page 27.

Restrictions

- Only automatic storage table spaces are supported in a Db2 pureScale environment and all the table spaces must be on a IBM Spectrum Scale storage path.
- Additional upgrade restrictions apply. Refer to "Upgrade restrictions for Db2 servers" on page 14. Review the complete list.

About this task

The following tasks describe the steps for direct upgrade from Db2 Version 10.1 or earlier to Db2 Version 11.1 pureScale environment.

Scenario 1:
About this task

In this scenario, the catalog table space is Database Managed Space (DMS) and the user table spaces are mostly System Managed Space (SMS). The database is not enabled for automatic storage.

Procedure

To upgrade a Db2 server to Db2 Version 11.1 pureScale environment:

1. Log on to the Db2 server as root.
2. Install Db2 Version 11.1. See "Installing Db2 servers using the Db2Setup wizard (Linux and UNIX)" in *Installing Db2 Servers*. Run the **db2setup** command and select the **Install New** option on the **Install a Product** panel to install a new copy of Db2 Version 11.1. Do not create an instance.
3. Run the **db2ckupgrade** command from the new Db2 installation path to verify that your databases are ready for upgrade.
4. Upgrade your Db2 Version 9.7 or Version 10.1 instances. Refer to "Upgrading Db2 Version 10.5 or Db2 Version 10.1 instances" on page 45.
5. Upgrade your databases. Refer to "Upgrading databases" on page 49.
6. Run the **db2cluster_prepare** command to set up IBM Spectrum Scale storage path. Refer to . Setting up IBM Spectrum Scale file system for a Db2 pureScale environment.
7. Create a new automatic storage enabled Db2 Version 11.1 database on the IBM Spectrum Scale storage path.
8. Load the data into the new database using the **db2move** with the **copy** option.
9. Run the **db2checkSD** command to verify that your database is ready to be upgraded to a Db2 pureScale environment.
10. Convert the instance to a Db2 Version 11.1 pureScale environment. Refer to . Converting your existing Db2 instances to a Db2 Version 11.1 pureScale environment.

Scenario 2:
About this task

In this scenario, the catalog table space is System Managed Space (SMS) and the user table spaces are Database Managed Space (DMS). The database is not managed by automatic storage.

Procedure

To upgrade a Db2 server to Db2 Version 11.1 pureScale environment:

1. Log on to the Db2 server as root.
2. Install Db2 Version 11.1. See "Installing Db2 servers using the Db2 Setup wizard (Linux and UNIX)" in *Installing Db2 Servers*. Run the **db2setup** command and select the **Install New** option on the **Install a Product** panel to install a new copy of Db2 Version 11.1. Create an ESE instance.
3. Run the **db2cluster_prepare** command to set up IBM Spectrum Scale storage path. Refer to . Setting up IBM Spectrum Scale file system for a Db2 pureScale environment.
4. Create a new automatic storage enabled Db2 Version 11.1 database on the IBM Spectrum Scale storage path.
5. Convert the table spaces to use automatic storage. Use a redirected restore operation with the **TRANSPORT** and **SET TABLESPACE CONTAINERS** command and specify the **USING AUTOMATIC STORAGE** parameter, to move all the schemas to the new database. Refer to . Converting table spaces to use automatic storage.
6. Run the **db2checkSD** command to verify that your database is ready to be upgraded to a Db2 pureScale environment.
7. Convert the instance to a Db2 Version 11.1 pureScale environment. Refer to . Converting your existing Db2 instances to a Db2 Version 11.1 pureScale environment.

Scenario 3:
About this task

In this scenario, the catalog table space is Database Managed Space (DMS) and the user table spaces are mostly Database Managed Space (DMS).

Procedure

To upgrade a Db2 server to Db2 Version 11.1 pureScale environment:

1. Log on to the Db2 server as root.
2. Run the **ALTER DATABASE** command with the **ADD STORAGE ON** storage-path option to enable your database for automatic storage.
3. Install Db2 Version 11.1. See "Installing Db2 servers using the Db2 Setup wizard (Linux and UNIX)" in *Installing Db2 Servers*. Run the **db2setup** command and select the **Install New** option on the **Install a Product** panel to install a new copy of Db2 Version 11.1. Create a Db2 Enterprise Server Edition instance.
4. Run the **db2cluster_prepare** command to set up IBM Spectrum Scale storage path. Refer to . Setting up IBM Spectrum Scale file system for a Db2 pureScale environment.
5. Run the **db2ckupgrade** command from the new Db2 installation path to verify that your databases are ready for upgrade.
6. Take a full offline backup of your database.
7. Do a redirected restore of the backup database into the Db2 Version 11.1 instance. Convert the table spaces to automatic storage and move the table spaces to IBM Spectrum Scale storage path. Refer to . Converting table spaces to use automatic storage.
8. Run the **db2checkSD** command to verify that your database is ready to be upgraded to a Db2 pureScale environment.

9. Convert the instance to a Db2 Version 11.1 pureScale environment. Refer to . Converting your existing Db2 instances to a Db2 Version 11.1 pureScale environment.

Upgrading Db2 servers with specific characteristics

There are many factors that can impact the overall upgrade process, and the complexity of your environment is one of these factors.

If you installed multiple Db2 product components, if you are upgrading from a 32-bit Windows operating system to a 64-bit Windows operating system, or if you are upgrading from a partitioned database environment, you must perform upgrade tasks that include steps specific to that environment instead of the basic Db2 server upgrade task.

Determine which of the following upgrade tasks apply to your Db2 server, and perform those tasks:

- "Upgrading Db2 32-bit servers to 64-bit systems (Windows)"
- "Upgrading non-root installations" on page 64
- "Upgrading a Db2 server with multiple Db2 copies" on page 66
- "Upgrading to a new Db2 server" on page 68
- "Upgrading a Db2 server by using online backups from a previous release" on page 71
- "Upgrading partitioned database environments" on page 72
- "Upgrading a Db2 pureScale server" on page 73
- "Upgrade Db2 High Availability Disaster Recovery (HADR) environments" on page 85
- Upgrading Db2 Text Search for administrator or root install
- Upgrading Db2 Text Search for non-root install (Linux and UNIX)
- Upgrading a multi-partition instance without Db2 Text Search
- "Upgrading Db2 servers in Microsoft Cluster Server environments" on page 101
- Upgrading Db2 Spatial Extender Version 11.1

Upgrading Db2 32-bit servers to 64-bit systems (Windows)

On the Windows operating systems, there are two ways to upgrade your Db2 32-bit server to a Db2 Version 11.1 64-bit server.

One way is to upgrade your existing 32-bit server to Version 11.1 32-bit server, and then upgrade to Version 11.164-bit server.

The other way is to upgrade to a new computer where Db2 Version 11.1 64-bit database product is installed.

Before you begin

- Ensure that you have Local Administrator authority.
- Ensure that the Db2 server is running 64-bit windows operating system.
- Review "Best practices for upgrading Db2 servers" on page 23 and "Disk space requirements for Db2 server upgrades" on page 21.
- Perform pre-upgrade tasks. See "Pre-upgrade tasks for Db2 servers" on page 27.

Restrictions

- This procedure is covered by this task and only applies to Windows on x64.

- Additional upgrade restrictions apply. See "Upgrade restrictions for Db2 servers" on page 14. Review the complete list.

Procedure

To upgrade a pre-Db2 Version 11.1 32-bit server to a Db2 Version 11.1 64-bit server:

1. Log on to the Db2 server as a user with Local Administrator authority.
2. If you have multiples copies of Db2 Version 10.1, or Db2 Version 9.7 or 32-bit server, perform the following actions to have all instances running under one Db2 copy:
   - Update all your instances to run under one Db2 Version 10.1 or Db2 Version 9.7 32-bit server copy. You can only update instances of the same version.
   - If you have instances running on multiple pre-Db2 Version 11.1 copies of different version, upgrade all instances to the highest release of pre- Version 11.1 copies. For example, if you have a Version 10.1 and a Version 9.7 instance, upgrade your Version 9.7 instance to the Db2 Version 10.1 32-bit server copy.
   - Uninstall all the remaining Db2 server copies except the Db2 server copy where all instances are running. You should have only one Db2 Version 10.1 32-bit server copy, or Db2 Version 9.7 32-bit server copy
3. Install Db2 Version 11.1 64-bit database product and select the **Work with Existing** option on the **Install a Product** panel. See "Installing Db2 servers (Windows) " in *Installing Db2 Servers*. Then in the **Work with an existing** window, choose the Db2 copy name with the **upgrade** action.
4. If you want your applications to access Db2 Version 11.1 copy through the default interface, set the Db2 Version 11.1 copy as the Db2 default copy. See "Changing the default Db2 and default IBM database client interface copy after installation (Windows)" in *Installing Db2 Servers*.
5. Upgrade your databases.
6. If you want to have your instances running on multiples copies of Db2 Version 11.1, install additional Version 11.1 copies and issue the **`db2iupdt`** command to run an instance under a different Version 11.1 copy.

What to do next

After upgrading the Db2 server, perform the recommended post-upgrade tasks such as resetting the diagnostic error level, adjusting log space size, and rebinding packages. In addition, verify that the upgrade of your Db2 server was successful. See "Post-upgrade tasks for Db2 servers" on page 102 and "Verifying upgrade of Db2 servers" on page 110.

Upgrading non-root installations

Upgrading Db2 Version 10.1, or earlier non-root installations to Version 11.1 on Linux and UNIX requires that you install Version 11.1 as a non-root user and then upgrade your databases to the non-root installation.

Before you begin

Before upgrading a non-root installation:

- Ensure that you meet the installation requirements for Db2 database products. See "Installation requirements for Db2 database products" in *Installing Db2 Servers*.

- Review upgrade recommendations and disk space requirements. See "Best practices for upgrading Db2 servers" on page 23 and "Disk space requirements for Db2 server upgrades" on page 21.
- Perform pre-upgrade tasks that apply, especially verifying that your databases are ready for upgrade. Upgrading the non-root instance verifies that your local databases are ready for upgrade. If this verification fails, the non-root instance upgrade also fails and the Db2 database product is not installed. See "Pre-upgrade tasks for Db2 servers" on page 27 and "Verifying that your databases are ready for upgrade" on page 29.

Restrictions

- You cannot upgrade a Db2 Version 10.1 or earlier root installation to a Version 11.1 non-root installation.
- You can upgrade databases from a Db2 Version 9.7 root installation to a Version 11.1 non-root installation by restoring database backups taken in the Version 9.7 root installation. Use the same process described in "Upgrading to a new Db2 server" on page 68.
- On Linux and UNIX operating systems except for Linux on x86, your existing 32-bit or 64-bit instances are upgraded to Db2 Version 11.1 64-bit instances. The operating system and Db2 Version 11.1 database product that you installed determines the instance bit size, see "Support changes for 32-bit and 64-bit Db2 servers" on page 22 for details.
- Additional upgrade restrictions apply. Review the complete list in "Upgrade restrictions for Db2 servers" on page 14.

Procedure

To upgrade a non-root installation to Db2 Version 11.1:

1. Log on to the Db2 server as the non-root user or owner for the Db2 Version 10.1 or earlier non-root installation.
2. Review Table 8 on page 15 to determine the instance type using the nodetype and the Db2 database product to which you can upgrade the non-root instance.

   The Db2 database product installation verifies that you can upgrade the non-root instance to the Db2 database product that you select for installation. If this verification fails, the installation fails and you can only end the installation.
3. Stop the non-root instance.
4. Install Db2 Version 11.1 as a non-root user and select the **upgrade** option. See "Installing a Db2 product as a non-root user" in *Installing Db2 Servers*.

   The **upgrade** option backs up the Db2 Version 10.1 or earlier non-root configuration files, installation directory, installs a new Db2 copy, and upgrades the non-root instance.

   However, the installation directory is not backed up if you specify the **`-f nobackup`** parameter and the Db2 Version 10.1 or earlier copy is removed.

   The Db2 product installation also verifies the following conditions:
   - The directory `*INSTHOME*/sqllib_v101` or `*INSTHOME*/sqllib_v97` does not exist.
   - The non-root instance is stopped.
   - The local databases running under the non-root instance are ready for upgrade.

   If any of these verifications fail and:

- You are running the **db2setup** command, a message box appears indicating the condition that failed. Take the appropriate corrective action and then select the **upgrade** option and continue.
- You are using a response file or running the **db2_install** command, the installer will exit with error. Take the appropriate corrective action and then reissue the **db2setup** command specifying the response file or reissue the **db2_install** command.

5. If the Db2 database product installation fails and you specified the **-f nobackup** parameter, manually install the Db2 database product and then run the **db2nrupgrade** command to upgrade the non-root instance as follows:
   ```
   cd $HOME/sqllib/instance
   db2nrupgrade -b BackupDir
   ```
   Where *BackupDir* is the backup directory for the configuration files of the non-root installation before upgrade. The backup directory is in the **db2setup** log in the format of `sqllib_v`*VR* where *V* is the version number and *R* is the release number of the old copy. For example, if you have Version 9.7 installed and then install Version 11.1 using the **db2setup** command, you can find the name of the backup directory as `sqllib_v101` in the **db2setup** log file.
6. If the Db2 database product installation fails, review the installation log file to determine the cause and how to resolve the issue before attempting the installation again. By default, the installation log file is located in the `/tmp` directory.
7. Upgrade databases. See "Upgrading databases" on page 49.
8. Enable root-based features by running the **db2rfe** command.
9. If you had additional Db2 products installed in your Db2 Version 10.1 or earlier non-root copy, install one Db2 product at a time.

What to do next

After upgrading the non-root installation, perform the recommended post-upgrade tasks such as resetting the diagnostic error level, adjusting log space size, and rebinding packages. In addition, verify that the upgrade of your Db2 server was successful. See "Post-upgrade tasks for Db2 servers" on page 102 and "Verifying upgrade of Db2 servers" on page 110.

Upgrading a Db2 server with multiple Db2 copies

Upgrading a Db2 server with multiple pre- Version 11.1 copies, requires that you install Db2 Version 11.1 as a new copy and then manually upgrade the instances and databases after installation.

You can have a Db2 server with multiple copies of Db2 database products Version 10.1 and Version 9.7 installed.

You can manually upgrade a pre-Db2 Version 11.1 instance at any fix pack level by executing the **db2iupgrade** command from the target Db2 Version 11.1 copy of your choice.

After an instance is upgraded to a Db2 Version 11.1 copy, you cannot upgrade it to another Version 11.1copy. However, you can update an instance between different Version 11.1 copies using the **db2iupdt** command.

Before you begin

- Ensure that you have root access on Linux and UNIX operating systems or Local Administrator on Windows.
- Ensure that you meet the installation requirements for Db2 database products. The requirements for operating systems have changed.
- Review upgrade recommendations and disk space requirements. See "Best practices for upgrading Db2 servers" on page 23 and "Disk space requirements for Db2 server upgrades" on page 21.
- Perform pre-upgrade tasks. See "Pre-upgrade tasks for Db2 servers" on page 27.

Restrictions

- This procedure does not apply to upgrade from Db2 32-bit servers to 64-bit systems on Windows. Refer to "Upgrading Db2 32-bit servers to 64-bit systems (Windows)" on page 63 for details.
- On Linux and UNIX operating systems, you must not setup the instance environment for the root user. Running the **db2iupgrade** or the **db2icrt** command for a root user is not supported.
- Review the upgrade restrictions for Db2 servers. See "Upgrade restrictions for Db2 servers" on page 14.

Procedure

To upgrade a Db2 server with multiple Db2 copies:

1. Log on to the Db2 server as root or a user with Local Administrator authority.
2. Install Db2 Version 11.1 at a new install path by running the Db2 Setup wizard and select the **Install New** option on the **Install a Product** panel. Refer to the following tasks for details:
   - Installing Db2 servers (Windows) in *Installing Db2 Servers*
   - Installing Db2 servers (Linux and UNIX) in *Installing Db2 Servers*

   You can install multiple Db2 Version 11.1 copies, if you want to upgrade your existing instances to different Db2 Version 11.1 copies.
3. Upgrade instances using the **db2iupgrade** command from the installation path of the Db2 Version 11.1 copy of your choice. See "Upgrading Db2 Version 10.5 or Db2 Version 10.1 instances" on page 45. For example, assume that you have the following Db2 copies and instances on an AIX server and a Windows server:

Table 13. Directory examples for Db2 copies.

Instance name	OS	Db2 copy directory
db2inst1	AIX	`/opt/IBM/db2/V9.7`
db2inst2	AIX	`/opt/IBM/db2/V10.1`
db2inst3	AIX	`/home/db2/myV10.1`
No instances created	AIX	`/opt/IBM/db2/V10.5` `/home/db2/myV10.5`
DB2_97	Windows	`C:\Program Files\IBM\SQLLIB_97\`
No instances created	Windows	`C:\Program Files\IBM\SQLLIB_10.5\`

   You can then run the following commands to successfully upgrade your instances to Db2 Version 11.1:

Table 14. Instance upgrade command examples.

Upgrade Instance	**Commands**

db2inst1	`cd /opt/IBM/db2/V10.5/instance`
	`./db2iupgrade -u db2fenc1 db2inst1`

db2inst2	`cd /opt/IBM/db2/V10.5/instance`
	`./db2iupgrade db2inst2`

db2inst3	`cd /home/db2/myV10.5/instance`
	`./db2iupgrade db2inst3`

DB2_97	`cd C:\Program Files\IBM\SQLLIB_10.5\`
	`db2iupgrade DB2_97 /u:db2admin1,password1`

4. Optional: Upgrade the Db2 Administration Server if you want to keep your existing configuration to administer your Db2 Version 11.1 instances. See "Upgrading the Db2 Administration Server (DAS)" on page 47.
5. Log on to the Db2 server as a user with SYSADM authority.
6. Upgrade databases. See "Upgrading databases" on page 49.

What to do next

After upgrading the Db2 server, perform the recommended post-upgrade tasks such as resetting the diagnostic error level, adjusting log space size, and rebinding packages. In addition, verify that the upgrade of your Db2 server was successful. See "Post-upgrade tasks for Db2 servers" on page 102 and "Verifying upgrade of Db2 servers" on page 110.

Upgrading to a new Db2 server

If you want to upgrade to a new Db2 Version 11.1 server, re-create your instances and then upgrade your databases by restoring a pre- Version 11.1 database backup. After restoring the database backup, the **RESTORE DATABASE** command automatically runs the **UPGRADE DATABASE** command.

Before you begin

- Ensure that you have root access on Linux and UNIX operating systems or Local Administrator authority on Windows.
- Ensure that you have SYSADM authority.
- Ensure that you meet the "Installation requirements for Db2 database products" in *Installing Db2 Servers*. The requirements for operating systems have changed.
- Review upgrade recommendations and disk space requirements. See "Best practices for upgrading Db2 servers" on page 23 and "Disk space requirements for Db2 server upgrades" on page 21.
- Perform pre-upgrade tasks. See "Pre-upgrade tasks for Db2 servers" on page 27.

Restrictions

- Review the upgrade restrictions for Db2 servers. See "Upgrade restrictions for Db2 servers" on page 14.

About this task

Use this procedure to upgrade your databases to a new server that has the same operating system as the old server. You can also use this procedure to upgrade your databases when the backup and restore operations are supported between the operating systems. For more information about this support, see Backup and restore operations between different operating systems and hardware platforms.

Procedure

To upgrade to a new Db2 Version 11.1 server:

1. Perform a full offline database backup of your existing databases and any other pre-upgrade tasks that apply. See "Backing up databases before or after upgrade" on page 32. If you performed full *offline* database backups recently and you cannot perform another one before upgrade, you can perform an incremental *offline* database backup instead.
2. Log on to the new Db2 server as root on Linux and UNIX operating systems or user with Local Administrator authority on Windows operating systems.
3. Install Db2 Version 11.1 on the new Db2 server.
4. Create your instances on the new Db2 server by running the **db2icrt** command from the Db2 Version 11.1 copy location that you installed in the previous step. See "Creating an instance using db2icrt" in *Installing Db2 Servers*. If the new \Db2 server has similar resources, then restore the database manager configuration parameter values for each instance using the **UPDATE DBM CFG** command and the values that you saved in the pre-upgrade tasks.
5. Optional: Create a new Db2 Administration Server (DAS) on Db2 Version 11.1. You need a DAS if you want to keep your existing DAS configuration and use new functionality available in Version 11.1.
6. Transfer pre-Db2 Version 11.1 backup files for all the databases that you want to upgrade to the new Db2 server.
7. Log on to the Db2 server as a user with SYSADM authority.
8. Upgrade the database using the **RESTORE DATABASE** command. The following example shows how to restore the sample database on UNIX operating systems:

   ```
   db2 RESTORE DATABASE sample FROM /db2/backups
   ```

   where *sample* is the database name and `/db2/backups` is the directory for the database backup file.

   If you performed an incremental *offline* database backup before upgrade, you must have access to the most recent full *offline* database backup and the incremental *offline* database backup and use an automatic incremental restore to upgrade the database. See "Using incremental restore in a test and production environment" in *Data Recovery and High Availability Guide and Reference*. A manual incremental restore will fail because each **RESTORE DATABASE** command tries to upgrade the database before the database is completely recovered. The following example shows how to perform an automatic incremental restore:

   ```
   db2 RESTORE DATABASE sample INCREMENTAL AUTOMATIC
       TAKEN AT timestamp WITHOUT PROMPTING
   ```

   In a partitioned database environment, you must execute the **RESTORE DATABASE** command in all database partitions starting with the catalog partition first. If sqlcode 7535 is returned as follows:

   ```
   SQL2517W The database was restored and then upgraded to the current release.
   The database upgrade returned sqlcode "7535" and tokens "*N".
   ```

   then you can run the **UPGRADE DATABASE** command again.
9. When the database was restored but the database was not upgraded, the **RESTORE DATABASE** command returns the following error and includes the upgrade error message with the reason code:

```
SQL2519N  The database was restored but the restored database was not upgraded
   to the current release.  Error "-1704" with tokens "3" is returned.
   SQLSTATE=57011
```

The error message SQL1704N indicates the database upgrade failed. Find this SQL error code in the *Message Reference Volume 2* to read the list of the possible solutions for each reason code. In the previous example, tokens "3" means reason code 3 which indicates that the upgrade failed because the database logs are full. If this error occurs, complete the following steps to upgrade the database:

a. Increase the size of the log files. See "Increasing table space and log file sizes before upgrade" on page 36.
b. Upgrade the database using the **UPGRADE DATABASE** command. See "Upgrading databases" on page 49.
c. If the log file size is still not large enough, the following error is returned:
   ```
   SQL1704N  Database upgrade failed.  Reason code "3".
   ```
   You must increase the log file size and attempt to upgrade the database again.
d. After the database upgrade is completed reset the size of the log files to their pre-upgrade values.

10. Optional: Configure your new Db2 server to use the new resources available by running the **AUTOCONFIGURE** command to calculate the buffer pool sizes, and the database manager and database configuration parameters values. The following example shows how to run this command to only display recommended values for the sample database:
    ```
    db2 CONNECT TO sample
    db2 AUTOCONFIGURE USING MEM_PERCENT 80
          WORKLOAD_TYPE complex
          NUM_STMTS 1 TPM 73
          ADMIN_PRIORITY performance
          IS_POPULATED YES
          NUM_REMOTE_APPS 15
          ISOLATION CS
        APPLY NONE;
    ```
    If you choose not to run this command or not to apply the recommended values, manually configure your Db2 server to use the new resources. Otherwise, your databases might not perform as expected.
11. Restore any external routines that you backed up in the pre-upgrade tasks. See "Backup and restore of external routine library and class files" in *Administrative Routines and Views*
12. Verify your database upgrade is successful. Connect to the upgraded databases and issue a small query:
    ```
       db2 CONNECT TO sample

         Database Connection Information

        Database server        = DB2/AIX64 10
        SQL authorization ID   = TESTDB2
        Local database alias   = SAMPLE

       db2  "SELECT * FROM SYSCAT.DBAUTH"
    ```
    Alternatively, if you have sample files installed, run the `testdata.db2` script:

```
cd samplefile-dir-clp
db2 connect to sample
db2 -tvf testdata.db2
```

where *samplefile-dir-clp* is `DB2DIR/samples/clp` on Linux and UNIX and `DB2DIR\samples\clp` on Windows, DB2DIR represents the location specified during Db2 Version 11.1 installation, and sample is the database name.

What to do next

After upgrading the Db2 server, perform the recommended post-upgrade tasks such as resetting the diagnostic error level, adjusting log space size, and rebinding packages. In addition, verify that the upgrade of your Db2 server was successful. See "Post-upgrade tasks for Db2 servers" on page 102 and "Verifying upgrade of Db2 servers" on page 110.

Upgrading a Db2 server by using online backups from a previous release

You can rebuild your database on a previous release by using online database backups from the same release and then upgrade to Db2 Version 11.1.

Before you begin

Before upgrading your Db2 server:

- Ensure that you have root access on Linux and UNIX operating systems or Local Administrator authority on Windows.
- All necessary full or incremental online pre-Db2 Version 11.1 database backups of your databases so that you can rebuild your databases by using these online backups.

Restrictions

Perform this task only under the following conditions:

- If you cannot upgrade the existing instances and databases.
- If you did not perform full *offline* database backups recently or incremental *offline* database backups as indicated in "Pre-upgrade tasks for Db2 servers" on page 27.

Procedure

To upgrade a Db2 server by using online backups from a previous release:

1. Transfer pre-Db2 Version 11.1 online database backup files for all the databases that you want to upgrade to the Db2 server.
2. If you do not have a Db2 copy with the same version as the online database backups, install a Db2 copy of the same version. For example, if you performed the online database backups from a Db2 Version 10.1 copy, you must have a Version 10.1 copy installed on the Db2 server.
3. If you do not have an instance running on the Db2 copy with the same version as the online backups, create an instance under this Db2 copy.
4. Log on to the Db2 server as a user with SYSADM authority.
5. Rebuild your databases by using the **RESTORE DATABASE** command with the **REBUILD WITH ALL TABLESPACES IN DATABASE** parameter followed by the **ROLLFORWARD DATABASE** command. For example:

```
RESTORE DB db-name
       REBUILD WITH ALL TABLESPACES IN DATABASE
       TAKEN AT timestamp-backup;
ROLLFORWARD DB db-name
       TO END OF LOGS AND STOP;
```

You can choose to rebuild your database with just a subset of table spaces. However, you must drop all table spaces in restore pending state after you issue the **ROLLFORWARD DATABASE** command. You cannot upgrade databases with table spaces in restore pending state.

Refer to "Database rebuild" in Data Recovery and High Availability Guide and Reference for more details.

6. Verify that the databases that you rebuild are in consistent state by issuing the **GET DB CFG** command as shown in the following example for Windows operating system:

   ```
   db2 GET DB CFG FOR sample | FIND "All committed transactions have been written"

   All committed transactions have been written to disk = YES
   ```

7. Upgrade the Db2 server by using one of the following tasks:
   - "Upgrading a Db2 server (Windows)" on page 43
   - "Upgrading a Db2 server (Linux and UNIX)" on page 52

Upgrading partitioned database environments

Upgrading partitioned database environments requires that you install Db2 Version 11.1 as a new copy in all database partition servers, upgrade the instances and then upgrade the databases.

Before you begin

- Ensure that you have root access on Linux and UNIX operating systems or Local Administrator authority on Windows.
- Ensure that you have SYSADM authority.
- Review the "Installation requirements for Db2 database products" in *Installing Db2 Servers*. The prerequisites for operating systems have changed.
- Review "Best practices for upgrading Db2 servers" on page 23 and "Disk space requirements for Db2 server upgrades" on page 21.
- Perform pre-upgrade tasks. Refer to "Pre-upgrade tasks for Db2 servers" on page 27.

Restrictions

- The database partition server where the catalog partition resides must be up and running.
- Use only the **Install New** option in the **Install a Product** panel to install Db2 Version 11.1. If you choose the **upgrade** action when you select the **Work with Existing** option on the **Install a Product** panel, the installation process fails.
- Additional upgrade restrictions apply. Refer to "Upgrade restrictions for Db2 servers" on page 14. Review the complete list.

Procedure

To upgrade Db2 servers in a partitioned database environment:

1. Perform a full offline backup for all database partitions. Use the **BACKUP DATABASE** command with the **ON ALL DBPARTITIONNUMS** parameter to back up all

partitions. Verify that your databases are ready for upgrade, and perform any other pre-upgrade tasks that apply. Refer to “Pre-upgrade tasks for Db2 servers” on page 27.

Note: Although database upgrade is a recoverable operation, for partitioned database environments recovery through migration is not supported. So ensure a full offline backup for all database partitions exists before proceeding.

2. Log on as root on Linux and UNIX operating systems or as a user with Local Administrator authority on Windows operating systems.
3. Install Db2 Version 11.1 on each participant database partition server and setup your partitioned database environment. See “Setting up a partitioned database environment” in *Installing Db2 Servers*. Select the **Install New** option in the **Install a Product** panel. Do not select the **Work with Existing** option.
4. Upgrade each instance on the database partition server that owns the instance. Refer to “Upgrading Db2 Version 10.5 or Db2 Version 10.1 instances” on page 45. The first entry in the `db2nodes.cfg` file of the instance is the database partition server instance owner.
5. Upgrade each database by running the **UPGRADE DATABASE** command on the catalog partition. Refer to “Upgrading databases” on page 49. The catalog partition must be available when you issue the **UPGRADE DATABASE** regardless on what database partition you issue this command from.

   If any database partitions are not available, these database partitions are not upgraded. Also, if the **UPGRADE DATABASE** command is stopped, the remaining database partitions are not upgraded. However, you can run the **UPGRADE DATABASE** command again to process these particular database partitions afterward when they are available.
6. Create a new Db2 Administration Server (DAS) on each database partition server. If you need to keep your existing DAS settings, you can upgrade the DAS on each participating database partition server instead of creating a new DAS. Refer to “Upgrading the Db2 Administration Server (DAS)” on page 47.

What to do next

After upgrading the Db2 server, perform the recommended post-upgrade tasks such as resetting the diagnostic error level, adjusting log space size, and rebinding packages. In addition, verify that the upgrade of your Db2 server was successful. Refer to “Post-upgrade tasks for Db2 servers” on page 102 and “Verifying upgrade of Db2 servers” on page 110.

Upgrading a Db2 pureScale server

Upgrading a Db2 pureScale server to Db2 Version 11.1 on Linux and UNIX requires that you install a new Version 11.1 copy and then manually upgrade your existing instances and databases to this new copy.

Before you begin

Before you upgrade the Db2 server:

- Ensure that you have root access.
- Ensure that you meet the installation requirements for your operating system. Refer to “Installation prerequisites for Db2 pureScale Feature (AIX)” and “Installation prerequisites for Db2 pureScale Feature (LINUX)” in *Installing Db2 Servers*.

- Review upgrade recommendations and disk space requirements. Refer to “Best practices for upgrading Db2 servers” on page 23 and “Disk space requirements for Db2 server upgrades” on page 21.
- Perform pre-upgrade tasks such as verifying that your databases are ready for upgrade and backing up database before upgrade. For more information, see “Pre-upgrade tasks for Db2 servers” on page 27.

About this task

This upgrade task describes the steps to upgrade to Db2 Version 11.1 pureScale server.

Restrictions

- Review the complete list of upgrade restrictions at “Upgrade restrictions for Db2 servers” on page 14.

Procedure

To upgrade to Db2 Version 11.1 pureScale server:

1. Log on to the Db2 server as the instance owner.
2. Disconnect all applications and users on every member node. To get a list of all database connections for the current instance, issue the **LIST APPLICATIONS** command. If all applications are disconnected, this command returns the following message:

   ```
   db2 list applications
     SQL1611W No data was returned by the Database System Monitor.
     SQLSTATE=00000
   ```

   To disconnect all applications and users, use the **FORCE APPLICATION** command:

   ```
   db2 force application all
   ```
3. Stop all command line processor sessions by entering the following command in each session that was running the command line processor:

   ```
   db2 terminate
   ```
4. When all applications and users are disconnected and all command line processor sessions are stopped, stop the database manager instance:

   ```
   db2stop
   ```

   Note: The time that is needed for the **db2stop** command to complete the action depends on the number of cluster caching facilities (CFs) and members in the cluster (up to 10 minutes).
5. If the **db2stop** command is not successful and fails to stop the database manager instance timeout due to new incoming connections, run the following command to disconnect all user connections and stop the database manager instance:

   ```
   db2stop force
   ```

   Ensure that the **db2stop force** command completes successfully.
6. Stop all instance processes in members and cluster caching facilities (CFs) in any order by issuing the **db2stop instance on <hostname>** command on any node, where *hostname* is the name of each member or CF in the cluster.
7. Install Db2 Version 11.1 by performing the following steps:
   a. Log on to the Db2 server with root user authority.

b. Put the cluster management software into maintenance mode on all members and cluster caching facilities (CFs) by issuing the following command from the *pre-Db2 Version 11.1 installation path* . This command stops the peer domain services on all hosts and prevents it from restarting during system maintenance. You can issue the following command only one time on any one of the members or CFs in the cluster.

```
OLD_DB2_INSTALL_DIR/bin/db2cluster -cm -enter -maintenance -all
```

where *OLD_DB2_INSTALL_DIR* is the *pre-Db2 Version 11.1 installation path.* For example, if you are upgrading from Db2 Version 10.5 to Db2 Version 11.1, the *OLD_DB2_INSTALL_DIR* is specified as `/opt/IBM/db2/V10.5`.

c. Put the cluster file system into maintenance mode on all members and CFs by issuing the following command from the *pre-Db2 Version 11.1 installation path.* This command stops all hosts from accessing the cluster files system (IBM Spectrum Scale) during system maintenance. You can issue the following command only one time on any one of the members or CFs in the cluster.

```
OLD_DB2_INSTALL_DIR/bin/db2cluster -cfs -enter -maintenance -all
```

If the above command returns a warning indicating the IBM Spectrum Scale module cannot be unloaded, a reboot will be required before proceeding to next step.

d. Install Db2 Version 11.1 using the Db2 Setup wizard. Run the **db2setup** command and select the **New Install** option on the **Welcome** panel. Choose **Db2 Version 11.1.0.0 Advanced Editions with Db2 pureScale** and click **Next** to install a new copy of Db2 Version 11.1.

```
db2setup -l /tmp/db2setup.log -t /tmp/db2setup.trc
```

In the **Configuration** panel, uncheck **Create an instance**. The Db2 installer still performs the installation, but, you can create an instance at a later point by running the **db2icrt** or **db2isetup** command. The Db2 Setup wizard provides a clear flow through which you can start a Db2 pureScale Feature installation from one member and successfully setup a Db2 pureScale environment across multiple members. The cluster management software and the cluster file system software are also upgraded during the installation to meet the V11.1 requirements. For more information, see "Installing the Db2 pureScale feature by using the Db2 Setup wizard (Linux and UNIX)" in *Installing Db2 Servers*.

e. Ensure that the IBM Spectrum Scalecluster is at the minimum release level by using the following steps:

   1) Use the **mmlsconfig** command to determine the minimum release level and whether the `usePersistentReserve` attribute is set to `YES`, as in the following example:

   ```
   root@XXX:/> /usr/lpp/mmfs/bin/mmlsconfig minReleaseLevel,usePersistentReserve
   minReleaseLevel 3.4.0.7
   usePersistentReserve yes
   ```

   If the `minReleaseLevel` value is 3.5 or higher, or if `usePersistentReserve` is set to `NO`, then go to step f.

   2) Use the **mmchconfig** command to update the IBM Spectrum Scale configuration information to the most current format that is supported by your IBM Spectrum Scale level, as in the following example:

   ```
   /usr/lpp/mmfs/bin/mmchconfig release=LATEST
   Verifying that all nodes in the cluster are up-to-date ...
   Verifying GPFS is stopped on all nodes ...
   ```

```
mmchconfig: Command successfully completed
mmchconfig: Propagating the cluster configuration data to all
 affected nodes.  This is an asynchronous process.
```

3) Verify that the release level is updated, by issuing the **mmlsconfig** command again:

```
/usr/lpp/mmfs/bin/mmlsconfig minReleaseLevel
minReleaseLevel 3.5.0.7
```

f. As a root user, take the cluster management software out of maintenance mode by issuing the following command from the *pre-Db2 Version 11.1 installation path.* You can issue the following command only one time on any one of the members or CFs in the cluster.

```
OLD_DB2_INSTALL_DIR/bin/db2cluster -cm -exit -maintenance -all
```

Note: If the old Db2 installation path is of Db2 Version 10.1, you must take the cluster management software out of maintenance mode by issuing the following command:

```
OLD_DB2_INSTALL_DIR/bin/db2cluster -cm -exit -maintenance
```

g. Ensure that all of the db2mnt resource groups are online, by issuing the **lssam** command as provided in the following example:

```
lssam -s "Name like '%db2mnt%rg'"
Online IBM.ResourceGroup:db2mnt-db2sd-rg Nominal=Online
        '- Online IBM.Application:db2mnt-db2sd-rs
                |- Online IBM.Application:db2mnt-db2sd-rs:XXX
                '- Online IBM.Application:db2mnt-db2sd-rs:YYY
```

If any groups are reported as `Unknown`, like in the following example, then reenter and exit maintenance mode, as described in step b and step f:

```
lssam -s "Name like '%db2mnt%rg'"
Unknown IBM.ResourceGroup:db2mnt-db2sd-rg Control=MemberInProblemState Nominal=Online
        '- Online IBM.Application:db2mnt-db2sd-rs  Control=MemberInProblemState
                |- Online IBM.Application:db2mnt-db2sd-rs:XXX
                '- Online IBM.Application:db2mnt-db2sd-rs:YYY
```

Note: In this scenario, you need to repeat steps b and f only; you do not need to repeat steps c, d, and e.

h. As a root user, take the cluster file system software out of maintenance mode by issuing the following command from the *pre-Db2 Version 11.1 installation path.*

```
OLD_DB2_INSTALL_DIR/bin/db2cluster -cfs -exit -maintenance -all
```

i. As a root user, commit changes to the cluster file system by issuing the following command from the Db2 Version 11.1 installation path. You can issue the following command only one time on any one of the members or CFs in the cluster.

```
NEW_DB2_INSTALL_DIR/bin/db2cluster -cfs -commit
```

where *NEW_DB2_INSTALL_DIR* is the *Db2 Version 11.1 installation path*. For example, if you are upgrading from Db2 Version 10.5 to Db2 Version 11.1, the *NEW_DB2_INSTALL_DIR* is specified as `/opt/IBM/db2/V11.1`.

j. As a root user, commit changes to the cluster management software by issuing the following command from the Db2 Version 11.1 installation path. You can issue the following command only one time on any one of the members or CFs in the cluster.

```
NEW_DB2_INSTALL_DIR/bin/db2cluster -cm -commit
```

where *NEW_DB2_INSTALL_DIR* is the *Db2 Version 11.1 installation path*.

   k. As an instance owner, restart the Db2 instance processes on all members and CFs in any order with updated resources for the cluster management software and the cluster file system software by issuing the **db2start instance on <hostname>** command on any node, where the hostname is the name of each member or CF in the cluster.
8. Install all Db2 add-on products that were installed in the Db2 copy from which you are upgrading.
9. Upgrade Db2 pureScale instances. Refer to "Upgrading Db2 pureScale instances."
10. Upgrade databases. Refer to "Upgrading databases" on page 49.

What to do next

After you upgrade the Db2 server, perform the recommended "Post-upgrade tasks for Db2 servers" on page 102 such as resetting the diagnostic error level, adjusting log space size, and rebinding packages. In addition, verify that the upgrade of your Db2 server was successful.

Upgrading Db2 pureScale instances:

As part of the overall process of upgrading your Db2 database server to Db2 Version 11.1, you must upgrade your pureScale instances.

Before you begin

- Your instance must be a Db2 pureScale instance.
- You must have root user authority.
- Ensure that you meet installation prerequisites.
- Verify that the new Db2 installation binary files are available on all members and cluster caching facilities (CFs) in the Db2 pureScale cluster in the same installation path.
- You must install any Db2 database add-on products that were installed in the Db2 copy from which you are upgrading.
- Before you run the **db2iupgrade** command, complete the following steps:
  - Verify that your databases are ready for Db2 upgrade. This step is important in Db2 pureScale environments because the **db2ckupgrade** command might return an error in one member and cause the instance upgrade to fail. Refer to "Verifying that your databases are ready for upgrade" on page 29.
  - Gather pre-upgrade diagnostic information to help diagnose any problem that might occur after the upgrade. For details, see "Gathering pre-upgrade diagnostic information" on page 39.

Restrictions

- You must not set up the instance environment for the root user. Running the **db2iupgrade** or the **db2icrt** command when you set up the instance environment is not supported.
- For more restrictions on instance upgrade, review "Upgrade restrictions for Db2 servers" on page 14.

Procedure

To manually upgrade your existing instances to Db2 Version 11.1 using the **db2iupgrade** command:

1. Log on to the Db2 server with root user authority.

2. Stop the Db2 pureScale instance by issuing the **db2stop** command. If you do not stop the instance before issuing the **db2iupgrade** command, your instance upgrade might fail.
3. Upgrade your existing instances by issuing the **db2iupgrade** command from the target Db2 Version 11.1 copy location.

   For Db2 Cancun Release 10.5.0.4 and above, issue the **db2iupgrade** command from the member in a Db2 pureScale cluster:

   ```
   $DB2DIR/instance/db2iupgrade -g [ -u fencedID ] InstName
   ```

   where *DB2DIR* is set to the location that you specified during Db2 Version 11.1 installation, **-g** is the default parameter that upgrades all the members and CFs in the Db2 pureScale cluster, *fencedID* is the user name under which the fenced user-defined functions (UDFs) and stored procedures run, and *InstName* is the login name of the instance owner. For example, if there are two members (Member 1, and Member 2) and two CFs (CF1 and CF2), if you issue the **db2iupgrade** command with the **-g** parameter from a member, all the members and CFs in the Db2 pureScale cluster are upgraded. Also, the instances are on the target level after upgrade.

   For Db2 Version 10.5 Fix Pack 3 and below, issue the **db2iupgrade** command from the Version 11.1 installation path of all the members first and then from the CFs:

   ```
   $DB2DIR/instance/db2iupgrade [ -u fencedID ] InstName
   ```

   Where *DB2DIR* is set to the location that you specified during Db2 Version 11.1 installation, *fencedID* is the user name under which the fenced user-defined functions (UDFs) and stored procedures run, and *InstName* is the login name of the instance owner.

   If you did not install all Db2 database add-on products that were installed in the Db2 copy from which you are upgrading, the instance upgrade fails and returns a warning message. If you plan to install these products later on or you no longer need the functionality that is provided by these products, use the **-F** parameter to upgrade the instance.
4. Update the number of Db2 Fast Communications Manager ports that are reserved for an instance during its creation in the `/etc/services` file. If you plan to extend your cluster after upgrade and to have some hosts with multiple members, increase the number of ports from the default value of four to six on all the hosts in the cluster. With this change, you can extend your cluster to a maximum of three members per host where three ports are reserved for each of the logical members and three ports are reserved for the idle processes that are used for restart light.

   Note: The ports must be contiguous and same across all the members and CFs in the cluster. When a range of six contiguous ports is not available, the port range can be changed provided they remain the same in all the hosts and they are contiguous.
   For example, the port range for an instance *db2inst1* in the `/etc/services` file before upgrade is as follows:

   ```
   DB2_db2inst1        60000/tcp
   DB2_db2inst1_1      60001/tcp
   DB2_db2inst1_2      60002/tcp
   DB2_db2inst1_END    60003/tcp
   ```

   After upgrade and before you start the instance, update the port range in the `/etc/services` file as follows:

```
DB2_db2inst1        60000/tcp
DB2_db2inst1_1      60001/tcp
DB2_db2inst1_2      60002/tcp
DB2_db2inst1_3      60003/tcp
DB2_db2inst1_4      60004/tcp
DB2_db2inst1_END    60005/tcp
```

5. Manually convert the tiebreak redundancy group ID from 3 to 4 in cluster with the GPFS replication setup. Refer to Post-upgrade task for cluster using GPFS replication.
6. Log on to the Db2 database server as an instance user to start your instance.
7. Verify that your instances are running on to Db2 Version 11.1 by running the **db2level** command: The Informational tokens must include a string like "Db2 Version 11.1.*X*.*X*" where *X* is a digit number.
8.

   Note: Starting from Db2 Version 11.1 Modification 1 Fix Pack 1, this step is no longer required.
   As a root user, rebuild the contents of the network resiliency condition and response resources in the cluster by issuing the **db2cluster -cfs -repair -network_resiliency -all** command. You can issue the command only once from any member or CF in the cluster.

   ```
   NEW_DB2_INSTALL_DIR/bin/db2cluster -cfs -repair -network_resiliency -all
   ```

   where *NEW_DB2_INSTALL_DIR* is the *Db2 Version 11.1 installation path*

What to do next

After upgrading your Db2 pureScale instance, you must upgrade your database. For more details, see "Upgrading databases" on page 49.

Upgrading databases:

After you upgraded your instances to Db2 Version 11.1, you must upgrade each database under each instance.

Before you begin

- Ensure that you have SYSADM authority.
- Ensure that all the local databases that you want to upgrade are cataloged.
- Ensure that you backed up your databases as indicated in "Pre-upgrade tasks for Db2 servers" on page 27.
- Ensure that you installed Db2 Version 11.1 and upgraded the instance to Db2 Version 11.1.

Restrictions

- Review the steps in "Upgrade restrictions for Db2 servers" on page 14 for database upgrade.

Procedure

To upgrade a Db2 database to Db2 Version 11.1:

1. Log on to the Db2 server as the instance owner or a user with SYSADM authority.
2. Optional: Rename or delete the **db2diag** log files so that new files are created. Also, remove or move to another directory any existing dump files, trap files,

and alert log files in the directory indicated by the **diagpath** parameter. By doing this, the files only contain information about the upgrade process that helps you to isolate and understand any problem that might occur during database upgrade.

3. Optional: Issue the **db2 LIST DATABASE DIRECTORY** command to ensure that the database is in the list of all catalogued databases in the current instance.
4. Upgrade the database using the **UPGRADE DATABASE** command:

   ```
   db2 UPGRADE DATABASE database-alias USER username USING password
   ```

   where *database-alias* is the name or the alias of the database you want to upgrade and the username and password to authenticate a user with SYSADM authority.

   To avoid the overhead of an automatic rebind, consider using the **REBINDALL** parameter, which specifies that a **REBIND** of all packages is performed during upgrade.
5. If the **UPGRADE DATABASE** command fails and returns the SQL1704N error message with a reason code that describes the cause of the failure, find this SQL error code and determine the action to take from the list of the possible solutions for each reason code. One of the most common causes of upgrade failure is that the log file space is not large enough, in which case the following error is returned:

   ```
   SQL1704N  Database upgrade failed.  Reason code "3".
   ```

   You must increase log file size and execute the **UPGRADE DATABASE** command again. For details, see “Increasing table space and log file sizes before upgrade” on page 36. After the database upgrade is complete reset the value of **logfilsiz**, **logprimary** and **logsecond** database configuration parameters.

   There are additional error codes that are returned by the **UPGRADE DATABASE** command for specific cases that are not supported by database upgrade. These cases are described in “Upgrade restrictions for Db2 servers” on page 14.
6. If the **UPGRADE DATABASE** command returns the SQL1243W warning message, you must drop or rename the SYSTOOLS.DB2LOOK_INFO table. Otherwise, the ALTER TABLE and COPY SCHEMA statements will fail to run. Check if the SYSTOOLS.DB2LOOK_INFO table exists by running the following command:

   ```
   db2 "SELECT tabname, tabschema, definer FROM syscat.tables
          WHERE tabschema = 'SYSTOOLS' AND tabname = 'DB2LOOK_INFO'"
   ```

   If you created this table, rename it by running the RENAME statement:

   ```
   db2 RENAME SYSTOOLS.DB2LOOK_INFO TO new-table-name
   ```

   If you did not create this table, remove it by running the DROP command:

   ```
   db2 DROP TABLE SYSTOOLS.DB2LOOK_INFO
   ```
7. If the **UPGRADE DATABASE** command returns the SQL1499W warning message and writes the ADM7535W warning message with all the details to the administration notification log, then the command failed to refresh the table space attributes in the catalog table. However, the database was upgraded successfully. However, the database was upgraded successfully.
8. If the **UPGRADE DATABASE** command returns the SQL1499W warning message and writes the ADM4003E warning message with all the details to the administration notification log, then the command failed to upgrade the Db2 Text Search catalogs or indexes due to an error in a stored procedure.

9. If the **UPGRADE DATABASE** command returns the SQL1499W warning message and writes the ADM7534W warning message with all the details to the administration notification log, then the command failed to refresh the table space attributes in the catalog table. However, the database was upgraded successfully. However, the database was upgraded successfully.
10. If the **UPGRADE DATABASE** command returns the SQL1499W warning message and writes the ADM4101W warning message to the administration notification log, take note of the system catalog tables reported in the ADM4101W message so that you collect statistics on these tables as part of the post-upgrade tasks.
11. If the **UPGRADE DATABASE** command returns the SQL1499W warning message and writes the ADM4102W warning message to the administration notification log, qualify or delimit with quotes the identifiers called NULL in your SQL statements to avoid conflict with the NULL keyword.

    If you use identifiers called NULL for column names, routine parameter names, or variable names in an SQL statement that are not fully qualified or delimited with quotes, the identifier name might resolve to the NULL keyword instead. This would result in a change in behavior from previous releases. Refer to "Upgrade essentials for database applications" on page 138 for details.
12. If the **UPGRADE DATABASE** command returns the SQL1499W warning message and writes the ADM9516W warning message to the administration notification log, verify that the **indexrec** configuration parameter is set to `RESTART` and issue the **RESTART DATABASE** command to rebuild indexes marked as invalid during database upgrade. Otherwise, index rebuild starts on your first access to the table and you might experience an unexpected degradation in response time.
13. If the **UPGRADE DATABASE** command returns the SQL0473N error message, you must reverse the database migration and re-create all user-defined data types that use a system built-in data type name with a different name that is not restricted. See "Reversing Db2 server upgrade" on page 120.

    To avoid the **UPGRADE DATABASE** command failure, re-create these user-defined data types during "Verifying that your databases are ready for upgrade" on page 29.
14. If the **UPGRADE DATABASE** command returns the DBT5512 error message, the command failed to upgrade the database because the ID of a workload management object conflicts with a system-reserved ID. To upgrade the database, perform the following actions:
    a. Generate the DDL statements to re-create the workload management objects by issuing the **db2look** command with the **-wlm** parameter.
    b. Drop all of the workload management objects from the database.
    c. Resolve all issues that are reported by the **db2ckupgrade** command and block the database from being upgraded.
    d. Upgrade the database.
    e. Re-create the workload management object in the upgraded database by issuing the DDL statements that t you generated with the **db2look** command.
15. If the **UPGRADE DATABASE** command returns the SQL1700N error message, you must reverse the database migration and re-create database objects that use restricted schema names with a schema name that is not restricted. See "Reversing Db2 server upgrade" on page 120.

To avoid the **UPGRADE DATABASE** command failure, re-create these database objects during "Verifying that your databases are ready for upgrade" on page 29.

16. If the **UPGRADE DATABASE** command returns the ADM4003E error message, then upgrade the Db2 Text Search catalog and indexes manually. For details, see **SYSTS_UPGRADE_CATALOG** and **SYSTS_UPGRADE_INDEX**.
17. Compare your database configuration settings after upgrade with the configuration settings you had before you upgraded your database. Verify that the following settings and database information are the same:
    - Database configuration parameter settings
    - Table spaces information
    - Packages information for your applications only

    You need not check package information for system generated packages. The information about system generated packages can change after upgrade.
18. Verify that your database upgrade is successful. Connect to the upgraded databases and issue a small query:

    ```
    db2 connect to sample

      Database Connection Information

     Database server        = DB2/AIX64 10.1.0
     SQL authorization ID   = TESTDB2
     Local database alias   = SAMPLE

    db2 "select * from syscat.dbauth"
    ```

    Alternatively, if you have sample files that are installed, run the `testdata.db2` script:

    ```
    cd samplefile-dir-clp
    db2 connect to sample
    db2 -tvf testdata.db2
    ```

    where *samplefile-dir-clp* is *DB2DIR*`/samples/clp` on Linux and UNIX and *DB2DIR*`\samples\clp` on Windows, *DB2DIR* represents the location that is specified during Db2 Version 11.1 installation, and sample is the database name.

What to do next

After upgrading a Db2 database, perform the recommended post-upgrade tasks to ensure a successful database upgrade. See "Post-upgrade tasks for Db2 servers" on page 102.

Upgrading of IBM Spectrum Scale level with a security-related efix:

While the best practice for upgrading IBM Spectrum Scale Scale software level is through the Db2 fixpack update leveraging existing online or offline software update procedures, IBM Spectrum Scale security efix, which may be released in between Db2 fixpacks, requires manual installation procedures. The instructions outlined in this topic provides the steps of updating a Db2 pureScale cluster with IBM Spectrum Scale security fixes in an online rolling fashion on AIX and Linux.

Before you begin

Before you install IBM Spectrum Scale security fix

- Your instance must be a Db2 pureScale instance.
- You must have root user authority.
- You must have root access.

Restrictions

- The base level of target IBM Spectrum Scale security efix must be the same as the current committed IBM Spectrum Scale level in the pureScale cluster. For example, if the current IBM Spectrum Scale level in the pureScale cluster is at 3.5.0.24 efix5, and IBM Spectrum Scale releases a new security efix9 a month later, the efix to be applied must be from the same IBM Spectrum Scale base level 3.5.0.24. The procedures documented in this topic cannot be used to move up the IBM Spectrum Scale base level.
- The procedures described below is restricted to be used for security fixes not included in any Db2 fixpacks but have been validated and supported by Db2 as documented in http://www-01.ibm.com/support/docview.wss?uid=swg21986595. Regular IBM Spectrum Scale upgrade must follow existing online or offline Db2 upgrade methods.

Procedure

1. Download the target efix files locally on each host in the cluster.
2. Log on to a host in the cluster and perform the following steps as Db2 instance owner.
   a. Stop all Db2 processes on the host. Use one of the following depending on whether the host is a member or CF:
      ```
      db2stop member <member_id>  quiesce <quiesce_timeout>
      ```
      or
      ```
      db2stop CF <cf_id>
      ```
   b. Stop the instance on the host:
      ```
      db2stop instance on <hostname>
      ```
3. Switch to the root ID on the same host and perform the following :
   a. Enter the maintenance mode on the host:
      ```
      DB2INSTANCE=<instanceName> <DB2_install_dir>/bin/db2cluster -cm -enter -maintenance
      <DB2_install_dir>/bin/db2cluster -cfs -enter -maintenance
      ```
      where <DB2_install_dir> is Db2 installation path

      Proceed to subsequent steps only if the above command returns successfully.
   b. Verify that IBM Spectrum Scale trace is not running:
      ```
      ps -ef | grep -i trace
      ```
      If the command above shows processes such as "trace" or "mmtrace", stop the IBM Spectrum Scale trace with the following:
      ```
      /usr/lpp/mmfs/bin/mmtrace stop
      ```
      Rerun "ps" command above to ensure the trace has been stopped successfully before proceeding.
   c. Unload IBM Spectrum Scale kernel and verify the result:

```
/usr/lpp/mmfs/bin/mmfsenv -u
echo $?
```

A zero return from the "echo ?" command indicates a successful unload. A non-zero value means the unload has failed. In that case, a reboot of the host must be performed before proceeding.

d. Install the target IBM Spectrum Scale efix.

   1) On AIX, any existing IBM Spectrum Scale efix must be uninstalled first. Run the following command to get the label of any IBM Spectrum Scale efix currently installed.

      ```
      /usr/sbin/emgr -P
      ```

      Sample output:

      ```
      $ /usr/sbin/emgr -P
      PACKAGE                                                 INSTALLER   LABEL
       ====================================================== =========== ==========
       gpfs.base                                               installp    g350p24e5
      ```

      Use the LABEL to remove the efix(es) that correspond with the gpfs.base package:

      ```
      /usr/sbin/emgr -r -L <GPFS_efix_label>
      ```

      Install the target IBM Spectrum Scale efix:

      ```
      /usr/sbin/emgr -e <full local path to the efix file>
      ```

      Verify the new efix has been applied, the label returned by the command below should match with one the of efixes in this link : http://www-01.ibm.com/support/docview.wss?uid=swg21986595

      ```
      /usr/sbin/emgr -l
      ```

   2) On Linux, proceed to the directory with the new IBM Spectrum Scale efix. For each package (starting with the gpfs.base-packages) run the following to install it:

      ```
      rpm -Uvh --force <rpm package>
      ```

      Verify the installation by:

      ```
      rpm -qa | grep -i gpfs
      ```

      Run the following command to compile the IBM Spectrum Scale Portability Layer (GPL) module:

      ```
      cd /usr/lpp/mmfs/src
      export SHARKCLONEROOT=/usr/lpp/mmfs/src
      make Autoconfig
      make World
      make InstallImages
      echo $?
      ```

      A zero return code from "echo $?" indicates a successful compilation. Do not proceed unless the compilation is successful.

e. Exit the maintenance mode:

   ```
   DB2INSTANCE=<instanceName> <DB2_install_dir>/bin/db2cluster -cm -exit -maintenance
   <DB2_install_dir>/bin/db2cluster -cfs -exit -maintenance
   ```

4. Switch back to the Db2 instance owner for the following steps:

   a. Start the instance on the host:

      ```
      db2start instance on <hostname>
      ```

   b. Start the member or CF on the host:

```
db2start member <member_id>
```

or

```
db2start CF <cf_id>
```

5. Repeat steps 2 to 4 above on the rest of the hosts in the same cluster.
6. Before committing to the new IBM Spectrum Scale level, allow applications to run for a short period of time to ensure the update does not introduce unexpected behaviour. When ready, switch to the root ID on any online host in the cluster to commit the new IBM Spectrum Scale level with the following command:

   ```
   <DB2_install_dir>/bin/db2cluster -cfs -commit
   ```

Upgrade Db2 High Availability Disaster Recovery (HADR) environments

For Db2 Enterprise Server Edition users upgrading from Db2 Version 10.5 Fix Pack 7 or later, and for Db2 pureScale users upgrading from Db2 Version 10.5 Fix Pack 9 or later, high availability disaster recovery (HADR) environments can now be upgraded without the need to reinitialize the standby database after performing an upgrade on the primary database. Reinitialization of the standby is still an option if the user wants, but is no longer the recommended option.

The supported HADR upgrade procedure requires the primary and standby databases both to be at a minimum of Db2 version. Any other prior level is not supported and attempts to use the documented procedure fails. For these prior level databases, the standby must be reinitialized by a backup image that is shipped from the primary after upgrade. The HADR upgrade procedure requires that one system/instance be identified to contain all primary databases and the second system/instance be identified to contain all standby databases. Where necessary, issue graceful takeover to achieve this layout of primary and standby databases.

Upgrading Db2 servers in HADR environments:

Upgrading a single partition Db2 Enterprise Server Edition V10.5 Fix Pack 6 or earlier primary database or a Db2 pureScale V10.5 Fix Pack 8 or earlier primary database, to Db2 Version 11.1 changes the database role from primary to standard.

Upgrading a single partition Db2 Enterprise Server Edition V10.5 Fix Pack 6 or earlier primary database, or a Db2 pureScale V10.5 Fix Pack 8 or earlier primary database, to Db2 Version 11.1 changes the database role from primary to standard. Upgrading a single partition Db2 Enterprise Server Edition V10.5 Fix Pack 6 or earlier standby database or a Db2 pureScale V10.5 Fix Pack 8 or earlier standby database to Db2 Version 11.1 is not supported because these databases are in rollforward pending state.

Because of these restrictions, upgrading an HADR environment to Db2 Version 11.1 requires that you stop HADR, upgrade your Db2 server where the primary database resides, and then reinitialize HADR.

Procedure

1. Stop the HADR primary or standby databases. For more information, see Stopping DB2® High Availability Disaster Recovery (HADR).
2. Upgrade the Db2 server where the primary database resides by using one of the following tasks:
   - “Upgrading a Db2 server (Windows)” on page 43

- "Upgrading a Db2 server (Linux and UNIX)" on page 52

3. Reinitialize HADR database. For more information, see Initializing high availability disaster recovery (HADR). Before you activate your standby database, you must move the log files from the previous version of Db2 out of the log path of the new version of Db2. If you do not move the log files, Db2 might try to use the old files and fail to initialize.

Upgrading Db2 servers in HADR environments (without standby reinitialization):

Use this procedure in a high availability disaster recovery (HADR) environment when you upgrade your Enterprise Server Edition Db2 Version 10.5 Fix Pack 7 or later databases to Db2 Version 11.1. This procedure maintains the database role and relies on normal log shipping and log replaying characteristics common to HADR functionality. The procedure avoids the need to stop HADR for upgrade and avoids the need to reinitialize HADR. This reduces the window where no standby database exists and eliminates the cost of sending a backup image to the standby site for reinitialization.

The procedure depends on the Db2 Version 10.5 Fix Pack 7 or later primary database having shipped all log data to the Fix Pack 7 or later standby databases and the standby databases having replayed all log data received.

The Db2 Version 11.1 standby database cannot replay log data from earlier Db2 versions. The procedure enforces this restriction through **db2ckupgrade** that is run during **db2iupgrade** or any **db2ckupgrade** invocation that establishes an exclusive connection to the database to do proper database consistency checking.

This procedure can be applied to HADR databases in Enterprise Server Edition environments that use single standby or multiple standby configurations.

Before you begin

- Review the system requirements for HADR. For more details, see System requirements for Db2 high availability disaster recovery (HADR).
- If you have two HADR databases (database A and database B) set up the following way, perform a role switch on one database so that both primaries are on the same system during the upgrade:
  - The primary for database A runs on system 1, and the standby runs on system 2
  - The primary for database B runs on system 2, and the standby runs on system 1
- Ensure that you are familiar with the steps involved in upgrading a Db2 instance and Db2 databases.
- Using High availability disaster recovery (HADR) monitoring, ensure that both the primary's log shipping functionality and the standby's log replaying functionality is working properly before running **db2ckupgrade**.
- If you are using the reads on standby feature, ensure that the database configuration parameter *logindexbuild* is turned on, on the primary database so that index recreation that is done during upgrade is sent to the standby database for replay. This allows read connections to resume post upgrade on the standby database.
- In case of failures during the HADR upgrade procedure, ensure that you are familiar with "Dealing with failures while upgrading Db2 servers in HADR environments (without standby reinitialization)" on page 93.

- All Db2 upgrades, hardware upgrades, and software upgrades should be implemented in a test environment before being applied to your production system.

About this task

This upgrade task describes the steps to upgrade HADR databases in an Enterprise Server Edition single standby environment to Db2 Version 11.1 from V10.5 Fix Pack 7 or later. Both the primary and standby database should be using the same fix pack level of Db2.

Restrictions

The primary and standby database must be from Db2 Version 10.5 Fix Pack 7 or later. It is recommended that both databases use the same fix pack level.

Procedure

To upgrade databases in an HADR environment to Db2 Version 11.1:

1. On the primary instance, monitor the database by using the **db2pd -hadr** or **MON_GET_HADR** command to ensure that primary log shipping and standby log replay are not lagging and are functioning properly. Monitoring the HADR database, reduces the chance of failures in the upgrade process. Also, ensure that replay delay is turned off on the standby to avoid significant time delay or lag.
2. Ensure that replay delay is turned off on all the standby databases by setting the database configuration parameter **hadr_replay_delay** to 0 to ensure the standby log replay position can catch up to the primary in a reasonable amount of time.
3. On the primary instance, deactivate the database by using the **DEACTIVATE DATABASE** command to ensure that log shipping is completed on the primary database and data is transferred to the standby database.
4. On the primary instance, stop the Db2 instance by using **db2stop** command. Ensure that applications do not connect so that no new log data is generated and the log replay position on the standby database eventually matches the log shipping position on the primary database.
5. On the primary instance, upgrade the Db2 instance by using the **db2iupgrade** command. The **db2iupgrade** command calls the **db2ckupgrade** command to verify that the database is ready for upgrade. The **db2iupgrade** does not run when the **db2ckupgrade** command reports errors. Check the log file if you encounter any errors. For an HADR primary database from V10.5 Fix Pack 7 or later, the **db2ckupgrade** command verifies that a valid standby database can be connected to the primary database. Once a connection is established, log shipping functionality begins and ships any pending log data, if necessary. The **db2ckupgrade** command also verifies that the log shipping position of the primary database matches the log replay position on the standby database.
6. On the standby instance, deactivate the database by using the **DEACTIVATE DATABASE** to ensure that log shipping is completed on the primary database and data is transferred to the standby database.
7. On the standby instance, stop the Db2 instance by using **db2stop** command.
8. On the standby instance, upgrade the Db2 instance by using the **db2iupgrade** command. The **db2iupgrade** command calls the **db2ckupgrade** command to verify that the standby database is ready for upgrade.

9. On the standby instance, issue the **UPGRADE DATABASE** command. This upgrades the database metadata objects and if log validation succeeded in the earlier steps the principal standby starts up and wait for a connection from the primary. The replay functionality begins in the background and wait for upgrade log data to be sent by the primary. At this point, the standby is considered upgrade in progress and the UPGRADE DATABASE command will return with SQL1103W. No user connections are allowed while in this upgrade in progress state. The progress on the standby database can be monitored using **db2pd -hadr** in conjunction with the Db2 diagnostics log. Connect attempts can also be issued and receives SQL1776N, reason code 9 as long as the standby is in upgrade in progress state.
10. On the primary instance, issue the **UPGRADE DATABASE** command to upgrade the primary database. This upgrades any database metadata files, like the database configuration, table pace definitions, log header control files, storage group files. Once these metadata files are upgraded, the primary database looks to connect to the standby database within the HADR timeout window. Once a connection is formed, upgrade processing begins and log data is sent to the standby database for replay.
11. On the primary instance, issue the **ACTIVATE DATABASE** command to start using the database in the new Db2 version.

What to do next

After upgrading the Db2 server, perform the recommended post-upgrade tasks such as resetting the diagnostic error level to its pre-upgrade value, adjusting log space size, and rebinding packages. In addition, verify that the upgrade of your Db2 server was successful. For more information see, "Post-upgrade tasks for Db2 servers" on page 102 and "Verifying upgrade of Db2 servers" on page 110.

Upgrading Db2 servers in HADR multiple standby environments (without standby reinitialization):

Upgrading your Db2 Version 10.5 Fix Pack 7 or later databases to Db2 Version 11.1 in a multiple standby environment is similar to the upgrade procedure for a single standby. This procedure maintains the database role and relies on normal log shipping and log replaying characteristics common to HADR functionality.

The procedure avoids the need to stop HADR for upgrade and avoids the need to reinitialize HADR. This reduces the window where no standby database exists and eliminates the cost of sending a backup image to all standby sites for reinitialization.

With multiple standby databases, you also have the added flexibility of upgrading all standby databases together to Db2 Version 11.1 or leaving some auxiliary standby database at Db2 Version 10.5 Fix Pack 7 or later until the primary and principal standby have completed the upgrade procedure in case of upgrade complications.

Before you begin

Ensure that you are familiar with the following tasks:
- "Upgrading Db2 servers in HADR environments (without standby reinitialization)" on page 86.

- In case of failures during the HADR upgrade procedure, ensure that you are familiar with “Dealing with failures while upgrading Db2 servers in HADR environments (without standby reinitialization)” on page 93.

About this task

This topic explains both upgrade options available to multiple standby environments.

Procedure

1. **Method 1**: Upgrading all standbys together to Db2 Version 11.1.
   a. In Db2 Version 10.5 Fix Pack 7 or later, using High availability disaster recovery (HADR) monitoring ensure that the primary database's log shipping functionality and all the standby database's log replay functionality are healthy.
   b. Ensure that replay delay is turned off on all the standby databases by setting the database configuration parameter **hadr_replay_delay** to 0 to ensure the standby log replay position can catch up to the primary in a reasonable amount of time.
   c. On the primary instance, deactivate all databases and stop the instance. Ensure that the standby database is active.
   d. Upgrade the primary instance using **db2iupgrade**. **db2iupgrade** calls **db2ckupgrade** and for HADR databases this validates the log positions between the primary database and all standby databases. If the log positions do not match for some database, then **db2iupgrade**/**db2ckupgrade** fails. If this happens, ensure the log shipping/replay functionality is still healthy for that database. If healthy, increase the hadr_timeout value for that database to give the log validation check more time for the log positions to match. If some standby is unable to catch up to the primary within enough time, then remove that standby from the **hadr_target_list**. The removed standby database needs to reinitialized post upgrade using a backup image. For a given database, if there are issues getting the log positions to match for principal and auxiliary standbys then you must proceed by stopping HADR on that database. That database needs to be upgraded then HADR reinitialized.
   e. On the standby instances, deactivate all databases and stop the instance. This can be done across all instances in parallel.
   f. Upgrade the standby instances using **db2iupgrade**. This can be done across all instances in parallel.
   g. For each database in the standby instances, starting with the principal standby, issue the **UPGRADE DATABASE** command. This upgrades the database metadata objects and if log validation succeeded in the earlier steps, the principal standby starts up and waits for a connection from the primary. The replay functionality begins in the background and waits for upgrade log data to be sent by the primary. At this point, the principal standby is considered upgrade in progress and the UPGRADE DATABASE command will return with:

      ```
      SQL1103W  The UPGRADE DATABASE command was completed successfully.
      ```

      No user connections are allowed while in this upgrade in progress state. The progress on the standby database can be monitored using **db2pd -hadr** with the Db2 diagnostics log. Connect attempts can also be issued and receives SQL1776N, reason code 9 as long as the standby is in upgrade in progress state.

You can repeat the same step for each auxiliary standby whenever convenient for you. The key is that the principal standby is brought up first and sits waiting for the primary before any auxiliary can issue the **UPGRADE DATABASE** command.

h. For each database in the primary instance, issue the **UPGRADE DATABASE** command. This upgrades the database metadata objects and if log validation succeeded in the earlier steps, attempts to form a connection with the principal standby database. Upgrade will not be able to proceed unless a connection to a valid principal standby database is available. Normal upgrade processing takes place and all log data are shipped to the principal standby database for replay. At any point once the primary is able to form a connection with an auxiliary database, log data is shipped to the auxiliary database for replay.
i. When upgrade on the primary database is complete the database can be activated and normal operations can resume.
j. When a standby database has replayed all the upgrade log data, the standby is no longer considered in upgrade in progress state and stays activated and can resume normal operations. For standby databases enabled for reads on standby that means read only connections can take place again.

2. **Method 2**: Leaving some auxiliary standby database at Db2 Version 10.5 Fix Pack 7 or later until the primary and principal standby have completed the upgrade procedure .
   a. Follow steps (a) through (e) under Method 1.
   b. Upgrade the principal standby instance using **db2iupgrade**.
   c. For each database in the principal standby instance, issue the **UPGRADE DATABASE** command. This upgrades the database metadata objects and if log validation succeeded in the earlier steps, the principal standby starts up and waits for a connection from the primary. The replay functionality begins in the background and waits for upgrade log data to be sent by the primary. At this point, the principal standby is considered upgrade in progress and the **UPGRADE DATABASE** command returns with:

```
SQL1103W  The UPGRADE DATABASE command was completed successfully.
```

   No user connections are allowed while in this upgrade in progress state. The progress on the standby database can be monitored using **db2pd -hadr** in conjunction with the Db2 diagnostics log. Connect attempts can also be issued and receive SQL1776N, reason code 9 as long as the standby is in upgrade in progress state.
   d. For each database in the primary instance, issue the **UPGRADE DATABASE** command. This upgrades the database metadata objects and if log validation succeeded in the earlier steps, attempts to form a connection with the principal standby database. Upgrade will not be able to proceed unless a connection to a valid principal standby database is available. Normal upgrade processing takes place and all log data are shipped to the principal standby database for replay.
   e. When upgrade on the primary database is complete the database can be activated and normal operations can resume.
   f. When a standby database has replayed all the upgrade log data, the standby is no longer considered in upgrade in progress state and stays activated and can resume normal operations. For standby databases enabled for reads on standby that means read only connections can take place again.

g. After confirming that everything is functioning as expected on both the primary and principal standby databases, you can repeat steps (b), (c) and (f) for each auxiliary database to upgrade them to Db2 Version 11.1.

What to do next

Post upgrade, follow the same recommendations as outlined in “Upgrading Db2 servers in HADR environments (without standby reinitialization)” on page 86.

Upgrading Db2 servers in HADR pureScale environments (without standby reinitialization):

Upgrading your supported databases to Db2 Version 11.1 in a pureScale environment is similar to the upgrade procedure for a single partition Enterprise Server Edition single standby.

For HADR environments that support upgrade without the need for standby reinitialization (V10.5 Fix Pack 9 and newer), this procedure maintains the database role and relies on normal log shipping and log replaying characteristics common to HADR functionality. The procedure avoids the need to stop HADR for upgrade and avoids the need to reinitialize HADR. This reduces the window where no standby database exists and eliminates the cost of sending a backup image to all standby sites for reinitialization.

Before you begin

Ensure you familiar with the following tasks:
- “Upgrading a Db2 pureScale server” on page 73
- “Upgrading Db2 servers in HADR environments (without standby reinitialization)” on page 86

In case of failures during the HADR upgrade procedure, ensure that you are familiar with “Dealing with failures while upgrading Db2 servers in HADR environments (without standby reinitialization)” on page 93.

Procedure

To upgrade pureScale databases in an HADR environment to Db2 Version 11.1:
1. In a supported Db2 Version 10.5 Fix Pack or later, using High availability disaster recovery (HADR) monitoring ensure that each primary member's log shipping functionality and the standby's log replay functionality are working properly.
2. Ensure that replay delay is turned off on the standby database by setting the database configuration parameter **hadr_replay_delay** to 0 to ensure the standby log replay position can catch up to the primary in a reasonable amount of time.
3. On the primary instance, deactivate all databases and stop the instance. Ensure that the standby database is active.
4. Upgrade the primary instance using Upgrading Db2 pureScale instances. As part of upgrading the instance, **db2iupgrade** calls **db2ckupgrade** and for each HADR database this validates the log positions between all primary members and the standby. If the log positions do not match for some database, then **db2iupgrade**/**db2ckupgrade** fails. If this happens, ensure the log shipping/replay functionality is still healthy for that database. If healthy, increase the **hadr_timeout** value for that database to give the log validation

check more time for the log positions to match. If there are issues getting the log positions to match then, you must proceed by stopping HADR on that database. That database needs to be upgraded then HADR reinitialized.

5. On the standby instance, deactivate all databases and stop the instance.
6. Upgrade the standby instance using Upgrading Db2 pureScale instances.
7. For each database in the standby instance, issue the **UPGRADE DATABASE** command on a single member. This upgrades the database metadata objects for all members and if log validation succeeded in the earlier steps the standby starts up and wait for a connection from the primary. The replay functionality begins in the background and wait for upgrade log data to be sent by the primary. At this point, the standby is considered upgrade in progress and the **UPGRADE DATABASE** command returns with:
   ```
   SQL1103W  The UPGRADE DATABASE command was completed successfully.
   ```
   No user connections will be allowed while in this upgrade in progress state. The progress on the standby database can be monitored by using `db2pd -hadr` with the Db2 diagnostics log. Connect attempts can also be issued and will receive SQL1776N, reason code 9 as long as the standby is in upgrade in progress state.
8. For each database in the primary instance, issue the **UPGRADE DATABASE** command on a single member. This upgrades the database metadata objects for all members and if log validation succeeded in the earlier steps attempt to form a connection with the standby database. Upgrade will not be able to proceed unless a connection to a valid standby database is available. Normal upgrade processing takes place and all log data will be shipped to the standby database for replay.
9. When upgrade on the primary database is complete the database can be activated and normal operations can resume across all members.
10. When the standby database has replayed all the upgrade log data, the standby is no longer considered in upgrade in progress state and stays activated and can resume normal operations.

What to do next

Post upgrade, follow the same recommendations as outlined in "Upgrading Db2 servers in HADR environments (without standby reinitialization)" on page 86.

Upgrading Db2 servers in an automated HADR environment:

Upgrade to Db2 Version 11.1 in an automated HADR environment from a previous Db2 Version.

Before you begin

- Ensure that you have root access on Linux and UNIX operating systems or Local Administrator authority on Windows.
- Ensure that you have SYSADM authority.
- Perform pre-upgrade tasks for Db2 servers

About this task

This topic will explain both upgrade options available to automated HADR environments.

Procedure

1. **Method 1**: Upgrading the HADR primary and standby database while maintaining the HADR role and without reinitializing the HADR pair .
   a. Export the current TSAMP resource configuration via **`db2haicu -o`** command on primary and standby hosts. Delete the domain by using the **`db2haicu -delete`** command. Deleting the cluster domain is an irreversible step. The resource model can be re-created after the upgrade is complete using the **`db2haicu -f`** command to import the configuration. The XML file generated above can be used as the input file. If the export is run on a single host (either primary or standby) and the same XML file is used to recreate the resource model after the upgrade, then the **`localHost`** and **`remoteHost`** parameters of HADRDB element will need to be modified depending on the host where the import is being performed.
   b. Follow the procedure for "Upgrading Db2 servers in HADR environments" on page 85.
2. **Method 2**: Upgrading the HADR primary and standby database through re-initialization of the HADR pair .
   a. Stop HADR by running **`db2 stop hadr`** command on the database **`dbname`**.
   b. Delete the resources for the current instance by using the **`db2haicu -delete`** command. Deleting the cluster resources is an irreversible step. Before you do this, ensure that a snapshot of the resource model for the instance is captured by using the **`lssam`** command. The resource model needs to be re-created after the upgrade is complete.
   c. Deactivate the databases on both the primary and standby servers.
   d. Stop the Db2 instance by using **`db2stop`** command.
   e. Install Db2 Version 11.1 in a new directory path.
   f. Upgrade the Db2 instance by using the **`db2iupgrade`** command.
   g. Start the Db2 instance by using the **`db2start`** command.
   h. If you are completing an upgrade on a standby server, upgrade the database by using the **`db2 upgrade dbname`** command.

What to do next

- If **Method 2** was used, reinitialize HADR.
- After the HADR pair is initialized once again, the cluster domain resources must be re-created. For more information about re-creating cluster domain resources, see: Automated cluster controlled HADR configuration setup using the IBM Db2 high availability instance configuration utility.

Dealing with failures while upgrading Db2 servers in HADR environments (without standby reinitialization):

This section describes how to deal with failures while upgrading Db2 servers in HADR environments (without standby reinitialization).

If using the procedure "Upgrading Db2 servers in HADR environments (without standby reinitialization)" on page 86, this procedure maintains the database role and relies on normal log shipping and log replaying characteristics common to HADR functionality. The procedure avoids the need to stop HADR for upgrade and avoids the need to reinitialize HADR. This reduces the window where no standby database exists and eliminates the cost of sending a backup image to the standby site for reinitialization.

During upgrade a failure can occur at any point in the procedure with any component that makes up the primary or standby. An upgrade is a scheduled event, so any failure is considered severe and having the primary or standby database available as quickly as possible is paramount. In most cases, a failure results in either the primary or standby database no longer being available to continue it's role in the HADR upgrade procedure. When this happens, the failing database must be taken out of it's role by stopping HADR, continuing upgrade as a non-HADR database, and then reinitializing HADR post upgrade.

It is difficult to document every possible failure scenario, but this topic attempts to walk you through what actions can be taken for failures at certain common points in the procedure.

Scenario 1: In Db2 Version 10.5 Fix Pack 7 or later, if the primary's log shipping functionality and the standby's log replay functionality are not healthy causing **db2iupgrade**/**db2ckupgrade** to fail.

If the issue cannot be fixed within the upgrade window, then follow the previous HADR procedure that requires the stopping of HADR and reinitialization discussed in "Upgrading Db2 servers in HADR environments" on page 85.

Scenario 2: In Db2 Version 10.5 Fix Pack 7 or later, if the primary's log shipping functionality and the standby's log replay functionality are healthy but the standby's replay position is still behind the primary's log shipping position causing **db2iupgrade**/**db2ckupgrade** to fail.

Ensure that replay delay is turned off by setting **hadr_replay_delay** to 0. Try to allow more time for the standby to catch up by increasing the **hadr_timeout** value. If neither of these options allow for the log positions to match within the upgrade window, then follow the previous HADR procedure that requires the stopping of HADR and reinitialization discussed in "Upgrading Db2 servers in HADR environments" on page 85.

Scenario 3: In Db2 Version 10.5 Fix Pack 7 or later, if the primary database becomes unavailable preventing **db2iupgrade**/**db2ckupgrade** from being run.

If the primary database cannot be brought back up within the upgrade window, switch roles on the standby and then follow the previous HADR procedure that requires the stopping of HADR and reinitialization discussed in "Upgrading Db2 servers in HADR environments" on page 85.

Scenario 4: In Db2 Version 10.5 Fix Pack 7 or later, if the standby database becomes unavailable preventing **db2iupgrade**/**db2ckupgrade** from being run.

If the standby database cannot be brought back up within the upgrade window, then follow the previous HADR procedure that requires the stopping of HADR and reinitialization discussed in "Upgrading Db2 servers in HADR environments" on page 85.

Scenario 5: In Db2 Version 11.1, if the primary database becomes unavailable preventing the upgrade procedure from continuing on the standby.

If the primary database cannot be brought back up within the upgrade window, on the standby issue **STOP HADR** followed by **ROLLFORWARD DATABASE** with the STOP option. This will turn the database into a non-HADR database. The database will now be upgrade pending and so issue the **UPGRADE DATABASE** command to continue the upgrade. Once complete refer to “Post-upgrade tasks for Db2 servers” on page 102 and “Verifying upgrade of Db2 servers” on page 110. HADR must be reinitialized.

Scenario 6: In Db2 Version 11.1, if the standby database becomes unavailable preventing the **UPGRADE DATABASE** command from starting up on the primary.

If the standby database cannot be brought back up within the upgrade window, on the primary issue **STOP HADR**. This turns the database into a non-HADR database. The database will still be upgrade pending so reissue the **UPGRADE DATABASE** command to continue the upgrade. Once complete refer to “Post-upgrade tasks for Db2 servers” on page 102 and “Verifying upgrade of Db2 servers” on page 110. HADR will have to be reinitialized.

Scenario 7: In Db2 Version 11.1, if the standby database becomes unavailable while in upgrade in progress state.

Once the **UPGRADE DATABASE** command is issued on the primary and the primary forms a connection with a standby database, the upgrade will proceed without issue on the primary and will eventually complete successfully. The concern is that there is no standby database replaying log data, which leaves an exposure to a loss of the primary. Post upgrade the primary database can still be brought up through the **START HADR** command specifying the `BY FORCE` option. At this point, all attempts should be made to resolve the issues with the standby. Once resolved, since the standby was in upgrade in progress state, the **UPGRADE DATABASE** command should be issued. The standby continues to replay the upgrade log data shipped by the primary until it completes and is no longer in the upgrade in progress state.

Scenario 8: In Db2 Version 11.1, if the **UPGRADE DATABASE** command with the `REBINDALL` option was specified on the primary and the standby database becomes unavailable while in upgrade in progress state.

The difference from Scenario 7 is that on the primary the **UPGRADE DATABASE** command was specified with the `REBINDALL` option. In this case, the **UPGRADE DATABASE** command requires and attempts a new connection to the database. If the standby database is not available during this second connection attempt, the **UPGRADE DATABASE** command returns SQL1499W. SQL1499W can be returned for many other reasons so the Db2 diagnostics log may be required to tell what failed and whether this scenario applies. If so, the primary database can still be brought up through the **START HADR** command specifying the BY FORCE option. Rebinding can still take place manually at this point. But, all attempts should be made to resolve the issues with the standby. Once resolved, since the standby was in upgrade in progress state, the **UPGRADE DATABASE** command should be issued. The standby continues to replay the upgrade log data shipped by the primary until it completes and is no longer in the upgrade in progress state.

At any time, if there are issues with the upgrade to Db2 Version 11.1, you can reverse the upgrade or fall back from Db2 Version 11.1 to a pre-Db2 Version 11.1 release. See “Reversing Db2 server upgrade” on page 120 to learn all the required steps to reverse a database upgrade.

Upgrading Db2 Text Search environments

Upgrading Db2 Text Search environments requires that you upgrade the Db2 server, instance, and all databases in the instance. Follow the steps to upgrade root installations or non-root installations that apply to your environment.

Upgrading Db2 Text Search for administrator or root installation:

To obtain the latest functionality upgrade your Db2 Text Search instance. You must upgrade the Db2 server, instance, and all databases when you are upgrading the text search instance.

Before you begin

Before you being to upgrade Db2 Text Search as administrator or root, complete the following steps:

1. Log in as the instance owner or a user with SYSADM authority.
2. Stop the Db2 database instance and the Db2 Text Search instance service.
3. Back up the Db2 Text Search configuration directory:
   - For Linux and UNIX operating systems, it is located under:
     ```
     $INSTHOME/sqllib/db2tss/config
     ```
     where *INSTHOME* represents the instance home path.
   - For Windows systems, it is located under:
     ```
     <INSTPROF>\<INSTNAME>\db2tss\config
     ```
     where *<INSTPROF>* represents the instance profile directory and *<INSTNAME>* indicates the name of the instance to be upgraded.
4. If you enabled Db2 Text Search for rich text document support, disable rich text document support. For more information about how to disable rich text document support, see the topic about disabling Db2 Text Search for rich text document support.

About this task

The following steps describe the process to upgrade Db2 Text Search Version 9.7 or Version 10.1 root installations on Linux or UNIX operating system, or for administrators on the Windows platform.

Procedure

1. Log on to the Db2 server as root on Linux and UNIX operating systems or user with Local Administrator authority on Windows operating systems. If you are upgrading a multipartitioned instance, you must perform instance upgrade from the instance-owning partition.
2. Install a new copy of V11.1 with a custom installation and make sure that Db2 Text Search is selected. Db2 Text Search is an optional component that is available only when you select a custom installation. You also can choose to install a new V11.1 copy over an earlier Db2 version by selecting `Work-With-Existing` mode and selecting Db2 Text Search as the component to be upgraded. You do not have to upgrade the Db2 instances after the installation with this approach.
3. Upgrade the Db2 Text Search server for your Db2 instances by issuing the **`configTool upgradeConfigFolder`** command. This command must be run as instance owner, and not root.
   - For Linux and UNIX operating systems:
     ```
     $DB2DIR/db2tss/bin/configTool upgradeConfigFolder
     -sourceConfigFolder $DB2DIR/cfg/db2tss/config
     -targetConfigFolder $INSTHOME/sqllib/db2tss/config
     ```

where, *INSTHOME* is the instance home directory and *DB2DIR* is the location of the newly installed V11.1 copy.

- For Windows operating systems:

```
<DB2PATH>\db2tss\bin\configTool upgradeConfigFolder
-sourceConfigFolder "<DB2PATH>\CFG\DB2TSS\CONFIG"
-targetConfigFolder "<INSTPROFDIR>\<INSTANCENAME>\DB2TSS\CONFIG"
```

  where, *<DB2PATH>* is the location of the newly installed V11.1 copy and *<INSTPROFDIR>* is the instance profile directory.

  Note: For Windows systems, if the Db2 instance was not configured previously for Db2 Text Search and the Db2 version to be upgraded is Version 9.7 Fix Pack 1 or later, you can skip this step.

The **`configTool upgradeConfigFolder`** command replaces, modifies, and merges text search configuration and data files and directories.

The config directory

The command copies the following files into the `<ECMTS_HOME>\config` directory if the files do not already exist in this directory:
- constructors.xml
- ecmts_logging.properties
- ecmts_config_logging.properties

The following files are copied and any existing files are overwritten:
- build_info.properties
- constructors.xsd
- ecmts_config_logging.properties
- mimetypes.xml
- monitoredEventsConfig.xml

The configuration settings from the following files are merged to the `configuration.xml` file. Values are added for new settings, and values are maintained for existing settings.
- config.xml
- jetty.xml

The following files are not modified:
- authentication.xml
- key.txt
- All files in the collections subdirectory

The log directory

The command does not change the contents of the existing log directory. However, when new log files are generated, those new files might replace existing log files.

The **`configTool upgradeConfigFolder`** command does not upgrade text search filters for an integrated text search server.

4. Upgrade the current Db2 instance by issuing the **`db2iupgrade`** command.
   - For Linux and UNIX operating systems, the command is located under the `$DB2DIR/instance` directory, where *DB2DIR* is the location of the newly installed Db2 database server V11.1 copy.

```
db2iupgrade  -j  "TEXT_SEARCH [[,service-name]|[,port-number]]" DB2INST
```

- For Windows operating systems, the property file is located in *<DB2PATH>*`\bin` directory, where *<DB2PATH>* is the location of the newly installed Db2 V11.1 copy.

  ```
  db2iupgrade DB2INST  /j  "TEXT_SEARCH [[,service-name]|[,port-number]]"
  ```

For more information, see the topic about **db2iupgrade** command.

Note: If you installed a new V11.1 copy with the upgrade option, and selected Db2 Text Search as a feature to be upgraded, then you can skip this step.

5. Back up the values for all configurable properties of Db2 Text Search that were used in the previous release by running the following script:
   - For Linux and UNIX operating systems:

     ```
     $DB2DIR/db2tss/bin/bkuptscfg.sh  $INSTNAME
     ```

     where, *DB2DIR* represents the location of the newly installed V11.1 copy, and *INSTNAME* represents the name of the instance to be upgraded.
   - For Windows operating systems:

     ```
     <DB2PATH>\db2tss\bin\bkuptscfg.bat <INSTANCENAME> <DB2PATH>
     ```

     where, *<DB2PATH>* represents the location of the newly installed V11.1 copy, *<INSTANCENAME>* represents the name of the instance to be upgraded.

   The backed-up configurable properties are redirected into one property file:
   - For Linux and UNIX operating systems, the property file is located in the `$INSTHOME/sqllib/db2tss/config/db2tssrvupg.cfg` directory, where *INSTHOME* represents the instance home directory.
   - For Windows operating systems, the property file is located in the `<INSTPROFDIR>\<INSTANCENAME>\db2tss\config\db2tssrvupg.cfg` directory, where *<INSTPROFDIR>* represents the instance profile directory and *<INSTANCENAME>* represents the name of the instance to be upgraded. You can find the name of the instance profile directory by issuing the **db2set DB2INSTPROF** command.

   Requirement: You must complete a backup for the values of all configurable properties of Db2 Text Search that are used in previous releases. Failure to create a backup results in a database upgrade failure.
6. Set the *DB2INSTANCE* environment variable to the current upgraded instance.
7. Upgrade the databases by issuing the **DB2 UPGRADE DATABASE** command. If the **DB2 UPGRADE DATABASE** command returns the ADM4003E error message, upgrade the Db2 Text Search catalog and indexes manually by using the SYSTS_UPGRADE_CATALOG and SYSTS_UPGRADE_INDEX stored procedures.
8. For each upgraded database, verify whether the text search server properties information in the text search SYSIBMTS.TSSERVERS catalog table is correct by comparing the property values backed up in step 7. If the value of the token or port number in the catalog table is empty or incorrect, you must update the text server information manually. For details about how to update, see the topic about updating Db2 Text Search server information.
9. Review the values for all Db2 Text Search configurable properties. Compare with the values that you backed up to ensure that they have correct values. Issue the following command to check the configuration values:

```
configTool printAll -configPath <configuration-directory>
```

10. If you disabled Db2 Text Search for rich text document support, you have to install Db2 V10.5 Accessories Suite For more information, see the topic about installing Db2 Accessories Suite.
11. Then enable rich text document support. For more information, see the topic about enabling Db2 Text Search for rich text and proprietary format support
12. Verify that the upgrade was successful by starting the Db2 Text Search instance service. If you disabled rich text document support, verify that rich text document support is enabled by issuing text search queries and compare with pre-upgrade results.

Upgrading Db2 Text Search for non-root installation (Linux and UNIX):

If you are upgrading Db2 Text Search Version 11.1, you must upgrade the Db2 server, instance, and all databases.

Before you begin

Complete the following tasks before you begin to upgrade your text search server:

1. Enable the root-based features for your user ID. You might have to ask a system administrator with root access to issue the **db2rfe** command.
2. Log in as the instance owner or as a user with SYSADM authority. Then stop the Db2 instance and the Db2 Text Search instance service.
3. Back up the old Db2 copy into a *<backup-dir>* directory.
4. If you enabled Db2 Text Search for rich text document support, disable rich text document support. For more information about how to disable rich text document support, see disabling Db2 Text Search for rich text document support.
5. Log on to the Db2 server as a non-root user. Review the database instance type to ensure it can be upgraded as a non-root installation.

Procedure

To upgrade Db2 Text Search:

1. Install a new Db2 Version 11.1 copy with the **db2nrupgrade** upgrade command. Select the Db2 Text Search component that you want to upgrade. If you specified the **-f nobackup** parameter and the Db2 database product installation failed, you must manually install the Db2 database product by selecting the Db2 Text search component from the feature tree and then upgrade the non-root instance by issuing the following command:
   ```
   db2nrupgrade -b <backup-dir> -j "TEXT_SEARCH"
   ```
   <backup-dir> specifies the directory where the configuration files from the old Db2 version are stored. For details about the upgrade non-root instance command, see **db2nrupgrade** command.
2. Back up values for all configurable properties of Db2 Text Search that is used in the previous release before the database upgrade by running the following script:
   ```
   $INSTHOME/sqllib/db2tss/bin/bkuptscfg.sh
   ```
   The backed-up configurable properties are redirected into the `$INSTHOME/sqllib/db2tss/config/db2tssrvupg.cfg` property file.
3. Upgrade the existing databases by issuing the **UPGRADE DATABASE** command.

4. For each upgraded database, verify whether the text search properties information in the text search catalog table SYSIBMTS.SYSTSSERVERS is correct by comparing the information with the property values from step 6. If the value of token or port number in the catalog table is empty or incorrect, you must update the text server information manually. For more information about the upgrading non-root instance, see updating Db2 Text Search server information.
5. Upgrade the Db2 Text Search server for your instances by issuing the **configTool upgradeInstance** command.
   - For Linux and UNIX operating systems:

     ```
     $DB2DIR/db2tss/bin/configTool upgradeConfigFolder
         -sourceConfigFolder $DB2DIR/cfg/db2tss/config
         -targetConfigFolder $INSTHOME/sqllib/db2tss/config
     ```

     INSTHOME is the instance home directory and *DB2DIR* is the location of the newly installed V11.1 copy.
6. Compare the values that you backed up in step 6 with the values for all the Db2 Text search configurable properties to ensure that all the values are correct. Issue the following command to check the configuration values:

   ```
   configTool printAll -configPath configuration-directory
   ```
7. If you disabled Db2 Text Search for rich text document support, you must install the Db2 V10.5 Accessories Suite. For information about the Accessories Suite, see installing Db2 Accessories Suite for Db2 Text Search.
8. Then enable rich text document support. For more information about enabling support, see enabling Db2 Text Search for rich text and proprietary format support.
9. Verify that the upgrade was successful by starting the Db2 Text Search instance service. If you disabled rich text document support, verify that rich text document support is enabled by issuing text search queries and compare with pre-upgrade results.

Upgrading a multi-partition instance without Db2 Text Search:

To obtain the latest functionality upgrade your Db2 Text Search instance. You need to upgrade the Db2 server, instance, and all databases when upgrading the text search instance.

About this task

Starting in Db2 Version 10.1, text search supports indexes in a partitioned database environment. The following steps describe the process to upgrade a Db2 Version 10.1 or Version 9.7 multi-partition instance for root install. Db2 Text Search should not be installed on the instances.

Procedure

1. Log in as the instance owner or a user with SYSADM authority.
2. Install a new copy of the Db2 Text Search version you are upgrading to, and perform a custom installation. Db2 Text Search is an optional component that is only available when you select a custom installation.
3. Upgrade your instances by issuing the **db2iupgrade** command:

   ```
   db2iupgrade /j "text_search [[,service-name]|[,port-number]]"
   ```
4. Upgrade the existing databases by issuing **DB2 UPGRADE DATABASE** command.

5. For each upgraded database, update the text server information manually. For more information, see the topic about updating Db2 Text Search server information.

Upgrading Db2 servers in Microsoft Cluster Server environments

Upgrading Db2 servers in Microsoft Cluster Server (MSCS) environments to Version 11.1 requires that you install Db2 Version 11.1 as a new copy in all nodes and then upgrade your MSCS instances and databases.

Microsoft Cluster Server (MSCS) provides High Availability functions to windows users. During setup of Db2 server failover support on MSCS, a server instance is transformed into an MSCS instance.

You can run the **db2iupgrade** command to upgrade your MSCS instance and to upgrade existing pre- Version 11.1 MSCS resources to Version 11.1 Db2 MSCS resources.

Before you begin

- Ensure that you have Local Administrator access.
- SYSADM authority is required.
- Review upgrade recommendations and disk space requirements. Refer to "Best practices for upgrading Db2 servers" on page 23 and "Disk space requirements for Db2 server upgrades" on page 21.
- Perform pre-upgrade tasks, especially back up your databases. Refer to "Pre-upgrade tasks for Db2 servers" on page 27 and "Backing up databases before or after upgrade" on page 32.

Restrictions

- This procedure applies only to upgrade from Db2 32-bit servers when you install the Version 11.1 32-bit database product, or from Db2 64-bit servers when you install the Version 11.1 64-bit database product.

  The instance bit size is determined by the operating system and the Db2 Version 11.1 database product that you install, see "Support changes for 32-bit and 64-bit Db2 servers" on page 22 for details.
- Use only the **Install New** option in the **Install a Product** panel to install Db2 Version 11.1. If you choose the **upgrade** action when you select the **Work with Existing** option on the **Install a Product** panel, the installation process fails.
- Additional upgrade restrictions apply. Refer to "Upgrade restrictions for Db2 servers" on page 14. Review the complete list.

Procedure

To upgrade a Db2 server in an MSCS environment to Db2 Version 11.1:

1. Log on to the Db2 server as a user with Local Administrator authority.
2. Install Db2 Version 11.1 in all of the nodes in the MSCS cluster. Run the **setup** command to launch the Db2 Setup wizard and select the **Install New** option in the **Install a Product** panel. Do not select the **Work with Existing** option.
3. Take the resource for the instance offline using the Cluster Administrator. The resource name is the same as the instance name. Ensure that all the remaining resources in the same group as the instance are online.

   For more information about using the Cluster Administrator, refer to MSCS documentation.

4. Upgrade your MSCS instances by running the **db2iupgrade** command. This command defines a new resource type called "Db2 Server", and updates all Db2 MSCS resources to use the new resource type. Having a new resource type during the upgrade eliminates conflict with existing pre-Db2 Version 11.1 MSCS resources.

   ```
   $DB2DIR\bin\db2iupgrade /u:user,password MSCS-InstName
   ```

   You must run this command from the node that owns all the instance-dependent resources.
5. Stop and restart the cluster service in all of the nodes in the MSCS cluster by using the Cluster Administrator.
6. Bring online the group of resources containing the upgraded instance by using the Cluster Administrator.
7. Optional: Upgrade your Db2 Administration Server (DAS) if you want to keep your existing DAS configuration and use new functionality available in Db2 Version 11.1.. Refer to "Upgrading the Db2 Administration Server (DAS)" on page 47.

   If you choose to create a new DAS, you have to re-configure the DAS settings for your MSCS environment.
8. Upgrade your databases. Refer to "Upgrading databases" on page 49.

What to do next

After upgrading the Db2 server, perform the recommended post-upgrade tasks such as resetting the diagnostic error level, adjusting log space size, and rebinding packages. In addition, verify that the upgrade of your Db2 server was successful. Refer to "Post-upgrade tasks for Db2 servers" and "Verifying upgrade of Db2 servers" on page 110.

Post-upgrade tasks for Db2 servers

After upgrading your Db2 servers, you should perform several post-upgrade tasks to ensure that your Db2 servers perform as expected and at their optimum level.

Procedure

Perform the following post-upgrade tasks that apply to your Db2 server:

1. If you set the **diaglevel** database manager configuration parameter to 3 or higher as recommended in the pre-upgrade tasks for Db2 servers, reset this parameter to the value set before the upgrade.
2. Existing tables that have row compression enabled from a pre-Db2 Version 11.1 database will have classic row compression enabled. If you want to use adaptive compression, it must to be enabled after the upgrade is performed. For details, see Adjusting adaptive compression settings.
3. If you changed your log space setting as recommended in the pre-upgrade tasks for Db2 servers, reset the **logfilsiz**, **logprimary**, and **logsecond** database configuration parameters to their pre-upgrade values. Ensure that the amount of log space that you allocate is adequate for your Db2 server. See "Adjusting the log space size in upgraded databases" on page 105 for details.
4. If you changed the **logindexbuild** database configuration parameter as optionally suggested in the pre-upgrade tasks for Db2 servers, reset the parameter to the pre-upgrade value.

5. Ensure that existing libraries for your external routines remain on the original location before the upgrade, if necessary, restore these libraries from the backup that you perform in “Backing up Db2 server configuration and diagnostic information” on page 34.
6. Activate your database after upgrade to start up your database and all necessary database services. See “Activating a database after upgrade” on page 106 for details.
7. Automatic storage table spaces inherit media attribute values, including overhead, device read rate and data tag attributes, from the storage group it is using by default. After upgrading to Db2 Version 11.1, the existing table spaces retain their settings and the OVERHEAD and DEVICE READ RATE attributes for the storage group are set to undefined. You can set media attributes with the ALTER STOGROUP statement. For details, see Storage group attributes.
8. Manage changes in Db2 server behavior. There are new registry variables, new configuration parameters, and new default values for registry variables and configuration parameters introduced in Db2 Version 11.1 that can impact the behavior of Db2 server. There are also changes in physical design characteristics of databases and changes to security that also have an impact. See “Managing Db2 server behavior changes” on page 107 for details.
9. If the automatic collection of statistics failed on certain system catalog tables during database upgrade, update the statistics on those system catalog tables. See “Collecting catalog statistics” in *Troubleshooting and Tuning Database Performance*.
10. If you did not use the `REBINDALL` option on the **`UPGRADE DATABASE`** command, then you can rebind packages in upgraded databases to validate packages. You can also rebind these packages to use updated statistics or new index information. This step may be skipped because the system task, **`db2pkgrb`**, will be started automatically and will rebind all invalid packages when a database starts on the catalog node. See “Rebinding packages in upgraded databases” on page 108 for details.
11. Refresh the data in existing materialized query tables by using the **`REFRESH TABLE`** statement. Materialized query tables (MQT) on unicode databases using language aware collation, where the MQT definition involves a LIKE predicate or substring function involved in a basic predicate, need to be refreshed.
12. Migrate Db2 explain tables to retain explain table information that you previously gathered. See “Upgrading explain tables” on page 109 for details.
13. If you obtained customized code page conversion tables from the Db2 support service, copy all of the files for those tables from the *`DB2OLD`*`/conv` to *`DB2DIR`*`/conv`, where *DB2OLD* is the location of your Db2 Version 10.1, or earlier copy and *DB2DIR* is the location of your Version 11.1 copy. You do not have to copy standard code page conversion tables.

    If you upgraded your existing Db2 Version 10.5 or earlier copy on Windows operating systems, you can restore the customized code page conversion tables that you backed up as part of the pre-upgrade tasks for Db2 servers to the **`DB2PATH`**`\conv` directory, where **`DB2PATH`** is the location of your Db2 Version 11.1 copy.
14. Upgrade existing target tables for event monitors that write to tables and to unformatted event (UE) tables by using the new EVMON_UPGRADE_TABLES procedure. For details, see Event monitor data retention from release to release.

15. Verify that your Db2 server upgrade was successful. Test your applications and tools to ensure that the Db2 server is working as expected. See "Verifying upgrade of Db2 servers" on page 110 for details.
16. Back up your databases after the Db2 server upgrade is complete. See "Backing up databases before or after upgrade" on page 32 for details.
17. For recoverable databases, Db2 no longer renames log files in the active log paths to the .MIG extension. Log files from previous releases are now treated like log files from the current release and regular Db2 log file management will occur. If you upgraded from Version 9.7, no log archiving will occur for log files in the original active log paths. You will be responsible for the management of these log files. The **db2cklog** utility can be used to determine what log files in the active log paths are from the pre-Version 11.1 release. If necessary, this can be used to assist in manual log file management.
18. If you have not already done so, you must migrate your SQL Replication in order to support new LSN formats.

What to do next

Perform the following post-upgrade tasks that apply to your Db2 database products or add-on features:

- If you upgraded your existing Db2 Version 10.5 or earlier copy, the database log directories will have been changed. Review the `db2diag.log` file which will have entries detailing the new log directories. If a user defined log directory is used, for example `/usr/logpath`, after upgrade the location of the log files will be `/usr/logpath/NODE0000/LOGSTREAM0000`. If the default database directory is being used, for example `/home/db2user/db2inst/NODE0000/SQL00001/SQLOGDIR`, after upgrade the location of the log files will be `/home/db2user/db2inst/NODE0000/SQL00001/LOGSTREAM0000`.
- .
- As an optional step, consider a strategy for updating statistics after version upgrades. New versions might introduce new SQL optimization, data statistics or both. To take full advantage of this, consider a strategy for updating your statistics over time. Simply relying on automatic statistics collection is one such strategy. Explicitly **RUNSTATs** invocation is another. However, again this is strictly optional. If the current performance and access plans meet your needs, you can skip this step.
- In general, version upgrades do not introduce additional reasons to reorganize all tables and indexes. That said, on occasion, functionality might be introduced that requires a reorganization for full exploitation. For example, new compression functionality might require a reorganization to be fully used. Considerations for reorganizing, if any, are provided with the documentation of the related new function.

At this point, you should resume all of your maintenance activities such as backing up databases and updating statistics. You should also remove any Db2 Version 10.5 or earlier copies that you no longer need.

Adjusting adaptive compression settings

Existing tables that have row compression enabled from a pre-Db2 Version 11.1 database will be upgraded to have classic row compression enabled. If you want to use adapative compression you must enable it after the upgrade is performed.

Before you begin

The default behaviour for compression has changed in Db2 Version 10.1, and has the syntax for enabling compression. For details, see "ALTER TABLE and CREATE TABLE statement have been changed."

About this task

Existing tables that have row compression enabled from a pre-Db2 Version 11.1 database will be upgraded to have classic row compression enabled. If you want to use adaptive compression you must enable it after the upgrade is performed.

Procedure

To take advantage of adaptive compression the following steps must be performed.

1. Estimate storage space savings by executing the administrative function **ADMIN_GET_TAB_COMPRESS_INFO**. Compare the generated estimate with the current or actual compression table savings. If the estimated compression savings that can be achieved using adaptive compression meet your requirements, proceed with enabling adaptive compression.
2. Perform **ALTER TABLE** with **COMPRESS YES ADAPTIVE** clause to enable adaptive compression. Modification of existing data rows and population of new rows will then be automatically subject to adaptive compression. Existing table rows are not immediately subject to adaptive compression as a result of issuing this ALTER statement. Any subsequent modification of existing rows or input of new rows into the table will lead to the application of adaptive compression.
3. If you want to compress all existing rows, you can perform a classic table reorganization to immediately have all existing rows compressed, in a table that has been enabled for adaptive compression. The classic table reorganization should ideally be preformed with the **RESETDICTIONARY** parameter to achieve the maximum compression possible. Subsequent reorganization for the purposes of better compressing data rows may no longer be required. If desired, use the **ADMIN_MOVE_TABLE** procedure instead of performing a classic table reorganization.

Adjusting the log space size in upgraded databases

You need to set the appropriate size for log files since it is one of the important factors in tuning your Db2 server. Also, if you increased the log files sizes as a pre-upgrade task, you can restore additional free space to your Db2 server.

Before you begin

To increase the size of table spaces and log space, you must have SYSCTRL or SYSADM authority.

Restrictions

On a partitioned database environment, you must adjust the log space size on the catalog database partition server.

Procedure

1. Connect to the database that you upgraded:

   ```
   db2 CONNECT TO sample
   ```

   where sample is the database name.

2. Restore your log file size settings to the values you had before upgrade:

   ```
   db2 UPDATE DB CFG FOR sample using LOGSECOND previous-value
   ```

   where *previous-value* is the setting that you save before upgrade and sample is the database name. In the pre-upgrade task, only the **logprimary** and the **logsecond** parameters were changed. If you change the setting for the **logfilsiz** parameter, you should restore the previous value.

   If you enabled infinite active logging, disable it by running the following commands:

   ```
   db2 UPDATE DB CFG FOR sample using LOGARCHMETH1 previous-value
   db2 UPDATE DB CFG FOR sample using LOGSECOND previous-value
   ```

   where *previous-value* is the setting that you save before upgrade and sample is the database name.
3. To support larger log record headers, increase the log space setting, by approximately 10% - 15% over what you used for Db2 Version 9.7.
4. To support larger log record headers, increase the **softmax** parameter by 10% - 15% over what you used for Db2 Version 9.7.

   ```
   db2 UPDATE DB CFG FOR sample using SOFTMAX 1.15 * previous-value
   ```

   Important: The **softmax** database configuration parameter is deprecated is deprecated in Version 10.5 and might be removed in a future release. For more information, see Some database configuration parameters are deprecated in *What's New for Db2 Version 10.5*.
5. Double the value for the **logbufsz** parameter:

   ```
   db2 UPDATE DB CFG FOR sample using LOGBUFSZ 2 * previous-value
   ```
6. Disconnect from the database that you upgraded:

   ```
   db2 CONNECT RESET
   ```

   logfilsiz changes take effect only when the database is reactivated. All applications must first disconnect from the database then deactivate and activate the database again.

Activating a database after upgrade

Activating your database allows you to ensure that all database services are running properly and to address any problems that might occur during the database activation. You can also eliminate the overhead on Db2 clients that have to wait until the database manager starts up the database to get a connection to this database.

Before you begin

Ensure that you have SYSMAINT, SYSCTRL, or SYSADM authority.

Procedure

To activate your databases after upgrade:

1. Start your database and all necessary database services with the **ACTIVATE DATABASE** command. The following example illustrates the use of this command to activate the sample database:

   ```
   db2 ACTIVATE DATABASE sample
   ```

   After this command is executed successfully your database is available for connections.

2. Review the administration notification log or the **db2diag** log files to verify that all database services are running properly and all buffer pools are activated. Address any problems that occurred during the database activation.

Results

Remember that a database, activated by the **ACTIVATE DATABASE** command, stops only when you issue the **DEACTIVATE DATABASE** command or the **db2stop** command. If the database is activated when the first connection is established, then the database is stopped when the last connection is closed.

Managing Db2 server behavior changes

The changes in Db2 registry variables, configuration parameters, and database physical design characteristics can have an upgrade impact. Review these changes to manage the upgrade impact.

About this task

After upgrading your Db2 server, compare the values of your registry variables and configuration parameters to their values before upgrade. If you find any differences, take the time to understand them because they could alter the behavior or performance of your applications. However, consider carefully whether to disable any new functionality because it provides support for new resources needed by the database manager. You should disable new functionality only if you experience negative performance or unwanted behavior.

Procedure

To manage Db2 server behavior changes:

1. Review the information about new, changed, deprecated, and discontinued registry variables, and based on the upgrade impact, choose the appropriate settings:
   - "Db2 server behavior changes" on page 17
   - Consider removing registry variables that have been deprecated or discontinued in Db2 Version 11.1 or earlier releases that can impact the upgrade of your Db2 server:
     –
     – Deprecated registry variables in Db2 Version 10.1
     – Discontinued registry variables in Db2 Version 10.1
     – Deprecated registry variables in Db2 Version 9.7
     – Discontinued registry variables in Db2 Version 9.7
2. Set your Db2 global profile registry variables. The variables that you set at the global profile level, using the **db2set** command with the **-g** option, are not upgraded. The global profile variables apply to all instances pertaining to a specific Db2 copy. Therefore, after upgrading your instances, use the configuration information that you saved in the pre-upgrade tasks to restore the values of your global profile registry variables for every Db2 Version 11.1 copy.
3. Review the information about new, changed, and deprecated database manager configuration parameters, and based on the upgrade impact, choose the appropriate settings:
   - "Db2 server behavior changes" on page 17

- There are no database manager configuration parameters that have been deprecated or discontinued in this release. However, if you are upgrading from Db2 Version 10.1 or earlier, consider removing database manager configuration parameters that have been deprecated in pre-Db2 Version 11.1 releases:
  - Deprecated database manager configuration parameters in Db2 Version 10.1
  - Deprecated database manager configuration parameters in Db2 Version 9.7
4. Review the information about new, changed, deprecated, and discontinued database configuration parameters, and based on the upgrade impact, choose the appropriate settings:
   - "Db2 server behavior changes" on page 17
   - Review the topic for further details on functionality that has been deprecated or discontinued in this release. If you are upgrading from Db2 Version 10.1 or earlier, consider removing database manager configuration parameters that have been deprecated or discontinued in pre-Db2 Version 11.1 releases:
     - Deprecated and discontinued database configuration parameters in Db2 Version 10.1
     - Deprecated and discontinued database configuration parameters in Db2 Version 9.7
5. Review the changes in database physical design characteristics and security, and based on the upgrade impact, modify database objects accordingly:
   - "Db2 server behavior changes" on page 17

What to do next

If you change the settings of any database manager configuration parameters that are not dynamic, you might need to restart the instance so the new settings take effect.

Rebinding packages in upgraded databases

During database upgrade, all packages for user applications and routines are marked as invalid. You may consider explicitly rebinding invalidated packages to take advantage of changes in the Db2 server and new statistics before allowing users access to the database. Alternately, you can allow an automatically started system task to rebind all invalid and inoperative (if **auto_reval** is set to anything other than DISABLED) packages after a database starts on the catalog node. The system task, **db2pcsd**, makes a single pass of all invalid packages and attempts to rebind them before terminating. See **db2pscd** in the Database EDU list for more information. Any access to a package that has not yet been rebound will be implicitly rebound upon first SQL request by an application.

Before you begin

Ensure that you have DBADM authority.

About this task

After you upgrade your database, any package not yet rebound by **db2pcsd** will be implicitly rebound when an application uses it for the first time. You may also choose to explicitly rebind invalid packages yourself . However, you must explicitly rebind inoperative packages. Alternatively, you can specify the **REBINDALL** option on the **UPGRADE DATABASE** command in Upgrading Databases.

This procedure applies only to C, C++, COBOL, FORTRAN, and REXX embedded SQL database applications.

Procedure

To explicitly rebind packages in upgraded databases:
1. Log on as a user with DBADM authority.
2. Rebind all invalid packages in each database:
   - From the CLP, run the **db2rbind** command, as follows:
     ```
     db2rbind database-name -l logfile -u userid -p password
     ```
     Review the log file specified by *logfile*, and address any issues. The **all** parameter should not be specified as it may attempt to rebind packages already bound by **db2pscd**.
   - From IBM Data Studio, open the task assistant to rebind packages.
3. Verify that your Db2 server upgrade was successful. For details, see Verify your Db2 server upgrade. Test your applications and tools to ensure that the server is working as expected. For details, see "Verifying upgrade of Db2 servers" on page 110 .

Results

After all the database packages have been rebound, you can automatically take advantage of optimizer improvements. Refer to "Upgrade essentials for database applications" on page 138 for details on the optimizer improvements available in this release.

Upgrading explain tables

If you must maintain explain table information that you gathered in your Db2 copies from previous releases, upgrade your explain tables to Db2 Version 11.1.

Before you begin

Ensure that you have DBADM authority. For additional authorization details, see *Command Reference*.

About this task

You can manually upgrade your explain tables after you upgrade your database, or you can re-create the explain tables and gather new information.

Procedure

To upgrade the explain tables, run the **db2exmig** command, as follows:

```
db2exmig -d dbname -e explain_schema -u userid password
```

where:
- *dbname* represents the database name. This parameter is required.
- *explain_schema* represents the schema name of the explain tables that you are migrating. This parameter is required.
- *userid* and *password* represent the current user's ID and password. These parameters are optional.

Results

The explain tables are upgraded. The **db2exmig** command renames the original explain tables, creates a new set of tables by using the `EXPLAIN.DDL` file, and copies the contents of the original explain tables to the new tables. Finally, the tool drops the original explain tables. The **db2exmig** command preserves any user-added columns in the explain tables.

What to do next

Use the **db2expln** command to see the access plan information in the upgraded explain tables.

Verifying upgrade of Db2 servers

When you upgrade your Db2 server, it is a good measure to run some tests on the new environment to verify that the Db2 server is working as expected. These tests can consist of batch programs that you usually run against the Db2 server or any programs or scripts that you run for benchmarks.

If you have Db2 command scripts with SQL statements, you can use the **db2batch** benchmark tool command to execute the statements in these scripts, and gather performance information details and statistics such as CPU time and elapsed time. This tool can work in both a single partition database and in a multiple partition database.

Before you begin

Ensure that you have the same authority level that is required to run the SQL statements in your script.

Procedure

To verify that your Db2 server upgrade was successful:

1. Log on to the Db2 server as a user with the same authority level that is required to run the SQL statements in the script.
2. Prepare a script with SQL statements that you frequently run. If you installed the sample files, you can also run any of the sample CLP scripts.
3. Run your script using the **db2batch** command. The following example shows you how to run this tool with the `testdata.db2` sample script:

   ```
   cd samplefile-dir-clp
   db2batch -d sample -f testdata.db2 -o r 0 p 3
   ```

   where *samplefile-dir-clp* is *DB2DIR*`/samples/clp` on Linux and UNIX and *DB2DIR*`\samples\clp` on Windows, *DB2DIR* represents the location for your Db2 Version 11.1 copy, sample is the database name, and the option **-o r 0 p3** indicates to print 0 fetched rows to the output and to report elapsed time, CPU time, and summary of monitoring information for each statement in the `testdata.db2` script.

   The following text is an extract of the summary table output generated by the command in the previous example:

   ```
    Summary Table:

   Type      Number Total Time Min Time  Max Time  Arithmetic Mean Geometric Mean
   --------- ------ ---------- --------  --------  --------------- --------------
   Statement      1 0.281284   0.281284  0.281284  0.281284        0.281284
   Statement      2 0.073158   0.073158  0.073158  0.073158        0.073158
   ```

```
Statement      3 0.000823   0.000823  0.000823  0.000823        0.000823
Statement      4 0.155366   0.155366  0.155366  0.155366        0.155366

* Total Entries:           4
* Total Time:              0.510630 seconds
* Minimum Time:            0.000823 seconds
* Maximum Time:            0.281284 seconds
* Arithmetic Mean Time:    0.127658 seconds
* Geometric Mean Time:     0.040271 seconds
```

Adopting new Version 11.1 functionality in upgraded databases

After you upgrade your Db2 server, enhance the functionality and improve the performance of your upgraded databases by adopting new Version 11.1 functionality.

Before you begin

You must upgrade your Db2 server to Version 11.1.

Procedure

Perform any of the following steps to adopt the specified Version 11.1 functionality in your upgraded Db2 environment:
- Use index that includes an expression in its key definition to enhance the performance of queries that contain expressions.
- Use column-organized tables in non-partitioned or partitioned environments to add columnar capabilities, improve the storage and query performance of databases. For more information, see Column-organized tables or Column-organized tables in partitioned database environments.
- Use large row size support so that the row length can exceed the maximum record length for the page size of the table space.
- Exclude NULL keys from indexes to reduce the size of sparse indexes and therefore improve storage and optimization.
- Use HADR solution in Db2 pureScale environments.

What to do next

If you upgraded your Db2 server from Db2 *Version 9.7*, adopt functionality that is introduced in *Version 11.1* in your upgraded Db2 environment. See the following topic for details:
- "Adopting new Version 11.1 functionality in upgraded databases."

Migrating Db2 functionality to Db2 database product features

Migrating Db2 functionality to specific Db2 database product features requires that you understand how the product feature works and how to implement equivalent functionality using a product feature.

The following migration tasks provides guidelines on how to implement workload management and XML data store features:
- "Migrating from Db2 Governor to Db2 workload manager" on page 112

Migrating from Db2 Governor to Db2 workload manager

Migrating from Db2 Governor to Db2 workload manager (WLM) requires that you set up your database for coexistence of Db2 Governor and Db2 WLM, re-examine your goals, and implement a workload management solution.

Before you begin

- Review your overall approach to workload management in light of the Db2 WLM capabilities provided to determine the best implementation. Refer to Workload management roadmap for a number of resources that are available to get you started with Db2 WLM, including "Best Practices: Db2 Workload Management."
- Review the Chapter 11. Db2 Governor in *Db2 Workload Manager for Linux, UNIX, and Windows* available at http://www.redbooks.ibm.com/redpieces/abstracts/sg247524.html for details about migration from Db2 Governor to Db2 WLM.
- If your existing workload management solution includes Query Patroller, also review Migrating from Query Patroller to Db2 workload manager. Query Patroller has been discontinued in Version 10.1.

About this task

There is no tool to automatically migrate your Governor configuration to Db2 WLM because the type of controls and mechanisms available are different between the two. When a query is running, the Governor watches for certain thresholds during the query execution which can trigger certain events. In Db2 WLM, a number of control mechanisms are available, in addition to the control of thresholds, which enable you to approach the same workload management problems in different but more effective ways.

This task provides guidelines to implement an efficient workload management solution and assist users migrating from Db2 Governor to Db2 WLM.

Important: With the workload management features introduced in Db2 Version 9.5, the Db2 governor utility was deprecated in Version 9.7 and might be removed in a future release. It is not supported in Db2 pureScale environments. For more information, see "Db2 Governor and Query Patroller have been deprecated" at http://www.ibm.com/support/knowledgecenter/SSEPGG_9.7.0/com.ibm.db2.luw.wn.doc/doc/i0054901.html.

Procedure

To migrate from Db2 Governor to Db2 WLM:

1. Upgrade the data server where the Governor is installed to Db2 Version 11.1 so that you have an environment where Db2 WLM and the Governor can coexist. Use one of the following tasks:
   - "Upgrading a Db2 server (Windows)" on page 43
   - "Upgrading a Db2 server (Linux and UNIX)" on page 52

   After the upgrade, there is a default workload created to identify all the user database activities and the workload is mapped to the default user service class which defines an execution environment. The Governor **ACTION NICE** rule clause is managed in only the default user service class. You cannot use the Governor to alter the priority of agents in user-defined service superclasses and subclasses. However, all other governor rules are enforced for all user-defined service classes.

2. Limit the use of Db2 WLM to control work in the default user service class to avoid potential conflicts between the Governor and Db2 WLM.
3. Re-examine your workload management goals. Understanding them is critical to implement a workload management solution.
4. Identify the work that runs on the data server and maps to your goals. Take advantage of the additional identification options at your disposal in Db2 WLM.
5. Manage the work that you identified by assigning resources and imposing controls to meet your goal metrics. Using any of the following approaches might result in a more simple and effective implementation:
   - Use Db2 service classes to separate and isolate competing workloads from each other or group database activities. Then change the agent, buffer pool, and prefetch priority options each service class receives to affect their individual response times. Try this approach first instead of creating concurrency thresholds.
   - Take note of the AUTHID and APPLNAME parameter values in the Governor control file and create a workload specifying the SESSION_USER and APPLNAME connection attributes using the AUTHID and APPLNAME parameter values.
   - If you cannot separate work by its source using workloads, map all incoming work to a common service super class and use a Db2 work action set to separate work by different characteristics and assign it to different service sub classes. At this point, manipulate the resources available to each service class to achieve your goals.
   - If you do not achieve the desired results by setting the priority options each service class receives alone, selectively apply other features of Db2 WLM as needed until you achieve your goals, such as the application of Db2 thresholds.
   - When you use Db2 thresholds, ensure that the threshold violations event monitor is created and activated; otherwise, you will not know when and what thresholds are being violated.
   - If you create thresholds to map to the same workloads the Governor was watching, consider all the thresholds available in Db2 WLM. Some of the Db2 Governor reactive rules will find a direct functional equivalent in Db2 workload management thresholds, like those controlling maximum execution time, the maximum number of rows returned, or the maximum connection idle time.
   - Other rules are unique to workload management or to the Db2 Governor and require you to rethink your approach to controlling work in current workload management terms. Note that Db2 Governor rules can apply to already running queries, whereas changes to Db2 WLM thresholds apply only to new queries.

     Consider all the different threshold actions available in Db2 WLM. You can choose a more forgiving action when a resource threshold is exceeded than ending the activity, such as letting the threshold continue execution or remapping it to a service subclass with different resource controls, and you can use the information logged in the threshold violations event monitor to further investigate the activity.
   - For the rowssel limit, you can create a threshold using the SQLROWSRETURNED condition to indicate what action should be taken when the limit of number of data rows returned to the application is exceeded.

- For the rowsread limit, you can create a threshold using the SQLROWSREAD or SQLROWSREADINSC condition to indicate what action should be taken when the limit of number of data rows read during query evaluation is exceeded.
- For the cpu limit, you can create a threshold using the CPUTIME or CPUTIMEINSC condition to indicate what action should be taken when the limit for the amount of combined user and system CPU time consumed by an activity is exceeded.
- For the idle limit, you can create a threshold using the CONNECTIONIDLETIME condition to indicate what action should be taken when the maximum connection idle time is exceeded.
- For the uowtime limit, you can create a threshold using the UOWTOTALTIME condition to indicate the length of time a unit of work is allowed to run.
- If you are using connection pooling, Db2 WLM has the client attributes available for proper identification and management of queries. The application at the middle tier could either call the sqleseti API or WLM_SET_CLIENT_INFO procedure to set one of the client attributes before it issues the SQL.
- If your data server runs on the AIX operating system, consider using AIX WLM for a more granular control of processor resource.

6. Monitor options to ensure that you are meeting your goals.

Recovering through a Db2 server upgrade

For recoverable databases, as part of the upgrade procedure to Db2 Version 11.1, you must develop a recovery plan in case issues arise during the upgrade or post upgrade.

Database upgrade is a recoverable operation so if you have a backup image and log files from a pre-Db2 Version 11.1 release and log files from Db2 Version 11.1 then post upgrade you can restore and roll forward to a point in time before any failure.

Db2 Version 11.1 cannot:
- restore a pre-Db2 Version 11.1 online backup image.
- roll forward through log files created in a pre-Db2 Version 11.1 release.

This means that the pre- Version 11.1 backup image must be restored in, and the roll forward command must be issued from Db2Version 10.5 Fix Pack 7 or later. This requires you to reinstall the previous release before the restore and roll forward can take place.

The roll forward will replay all log records created in the previous release until it comes to the end of release. At this point, you can choose to:
- Stay in the previous release, considered reversing or fall back of upgrade. If so, see “Reversing Db2 server upgrade” on page 120.
- Move to Db2 Version 11.1 and continue the roll forward through the upgrade.

If you intend to roll forward to a point in time post upgrade in Db2 Version 11.1 then you must re-install Db2 Version 11.1 and continue the roll forward to the point in time you want. Once the database comes out of roll forward pending state then all post upgrade tasks should be reexamined to make sure that the database is ready to accept new workload. For example, this could involve rebinding of packages. If the database configuration parameter **logindexbuild** was not on during the initial upgrade, then during roll forward, catalog indexes are marked bad. First connect recreates these indexes, which might create a delay before data can be accessed.

It is highly suggested that a new online database backup be created at your earliest convenience so that you do not have to rely on this recovery procedure again.

Before you begin

- Ensure that you have root access on Linux and UNIX operating systems or Local Administrator authority on Windows.
- Ensure that you have SYSADM authority.
- Perform the following steps before you upgrade your Db2 server:
  - Review upgrade recommendations and disk space requirements.
  - Before upgrade, ensure that you have a valid backup image and valid log files for all databases taken in the previous release.
  - After upgrade, ensure that you have valid log files for all databases taken in Db2 Version 11.1.
  - Back up all database manager configuration parameter values for each instance and all database configuration parameter values for each database.
  - Perform other pre-upgrade tasks that apply to your environment.

Restrictions

- This procedure applies only to Db2 server upgrade. It does not include Db2 clients.
- This procedure is not supported in partitioned database environments. Take an offline full backup of all databases that you are going to upgrade.
- This procedure does not work if the previous database version is Db2 Version 9.7, Version 10.1 or Version 10.5 Fix Pack 7 or earlier.
- This procedure should only be used for failures that occur after upgrading to Db2 Version 11.1 and before a new full database online backup completes.
- This procedure should only be used to recover a database that has successfully been upgraded to Db2 Version 11.1 and where user workload transactions have been performed that would be lost otherwise.
- This procedure should only be used on a major failure that can only be recovered restoring the most recent backup, for example, storage failures or data corruptions.

About this task

This procedure could also be used in a cold standby system to recover a database after an upgrade and before a new full database online backup completes. As recommended, the cold standby system should remain at Db2 Version 10.5 Fix Pack 7 or later until a new full database online backup completes on the main system. For this reason, the recovery procedure on the cold standby system starts with the step that restores the pre-Db2 Version 11.1 backup.

Recovery in a multiple database scenario when only a subset requires recovery: In case only a subset of the databases require recovery the procedure below does not change. In this case, all databases will take an outage while the subset that needs recovery are recovered. The steps can be optimized by moving only once to the previous release then doing the recovery for all databases to the end of release, followed by moving up to Db2 Version 11.1 once, and then finishing the recovery for all databases.

In the previous release, the restore can consist of rebuilding a database. It is highly recommended to choose all table spaces in the database. If choosing a subset of table spaces, be aware that once you continue the roll forward in Db2 Version 11.1 any table space in restore or rollforward pending can not be recovered. Those table spaces will have to be dropped and re-created.

Choose the procedure that matches your environment:
- "Recovering through a Db2 single partition server upgrade (Windows)"
- "Recovering through a Db2 single partition server upgrade (Linux and UNIX)" on page 117
- "Recovering through a Db2 pureScale server upgrade (Linux and UNIX)" on page 118

What to do next

Take a new full online database backup in Db2 Version 11.1. Once this completes and is verified this procedure is not necessary anymore.

Recovering through a Db2 single partition server upgrade (Windows)

Use this procedure to recover one or more databases through a Db2 single partition Windows server upgrade.

Before you begin

- Ensure that you have read and are familiar with the details and restrictions discussed in "Recovering through a Db2 server upgrade" on page 114.
- Make sure that the backup image that is used to perform recovery is accessible and previously validated.

Procedure

1. IMPORTANT: As instance owner, protect the log files as in the next step you will be required to drop the database (in multi-database environments, apply this to all databases that require recovery). This step is necessary as some or all of the log files in the active or mirror log paths might not have been archived and the drop of the database deletes all of those log files. To get the current value of these log paths, use:
   ```
   db2 get db cfg for <database_name> | grep -i log
   ```
   Note: This copy must be retained until this recovery procedure is completed and a new online full database backup is completed and verified, which includes a copy of the log files.
2. Log on to the Db2 server as a user with **SYSADM** authority, issue list database directory to view the database path where the database is restored to (in multi-database environments, apply this to all databases that require recovery).
3. Log on to the Db2 server as a user with **SYSADM** authority, drop all databases that need recovery in Db2 Version 11.1 by running the **DROP DATABASE** command.
4. As instance owner, stop the instance using **db2stop**.

5. Uninstall your Db2 copy through `Control Panel > Programs`. This command does not remove the database files.
6. Re-create your Db2 copy by running **db2setup**. Then restore the database manager configuration parameter values for each instance by using the **UPDATE DATABASE MANAGER CONFIGURATION** command.
7. As instance owner, start the instance using **db2start**.
8. For each database that needs recovery, restore your databases from a pre-Db2 Version 11.1 backup by running the **RESTORE DATABASE** command to the database paths listed in step 2.
9. Roll forward all databases restored in step 8 to a point in time before the failure. Ensure that you supply the log files saved off in step 1 to the **ROLLFORWARD DATABASE** command. This can be done by copying the log files back into the active log paths or by supplying the **OVERFLOW LOG PATH** parameter on the **ROLLFORWARD DATABASE** command. If log archiving is being used ensure that the archive location is available. When the roll forward hits SQL2463N or SQL2464N that means the end of release is reached.
10. As instance owner, stop the instance using **db2stop**.
11. Uncatalog all databases by using the **UNCATALOG DATABASE** command.
12. Upgrade the instance.
13. As instance owner, start the instance using **db2start**.
14. Catalog all databases using the **CATALOG DATABASE** command.
15. For all databases that need recovery, continue the roll forward done in step 9 until complete.

What to do next

See "Recovering through a Db2 server upgrade" on page 114.

Recovering through a Db2 single partition server upgrade (Linux and UNIX)

Use this procedure to recover one or more databases through a Db2 single partition Linux or UNIX server upgrade.

Before you begin

- Ensure that you have read and are familiar with the details and restrictions discussed in "Recovering through a Db2 server upgrade" on page 114.
- Make sure that the backup image that is used to perform recovery is accessible and previously validated.

Procedure

1. IMPORTANT: As instance owner, protect the log files as in the next step you will be required to drop the database (in multi-database environments, apply this to all databases that require recovery). This step is necessary as some or all of the log files in the active or mirror log paths might not have been archived and the drop of the database deletes all of those log files. To get the current value of these log paths, use:

   ```
   db2 get db cfg for <database_name> | grep -i log
   ```

   Note: This copy must be retained until this recovery procedure is completed and a new online full database backup is completed and verified, which includes a copy of the log files.

2. Log on to the Db2 server as a user with **SYSADM** authority, issue list database directory to view the database path where the database is restored to (in multi-database environments, apply this to all databases that require recovery).
3. Log on to the Db2 server as a user with **SYSADM** authority, drop all databases that need recovery in Db2 Version 11.1 by running the **DROP DATABASE** command.
4. As instance owner, stop the instance using **db2stop**.
5. Drop your Db2 Version 11.1 instance by running the **db2idrop** command. This command does not remove the database files.
6. Re-create your instance in the pre-Db2 Version 11.1 by running the **db2icrt** command. Then restore the database manager configuration parameter values for each instance using the **UPDATE DATABASE MANAGER CONFIGURATION** command.
7. As instance owner, start the instance by using **db2start**.
8. For each database that needs recovery, restore your databases from a pre-Db2 Version 11.1 backup by running the **RESTORE DATABASE** command to the database paths listed in step 2.
9. Roll forward all databases restored in step 8 to a point in time before the failure. Ensure that you supply the log files saved off in step 1 to the **ROLLFORWARD DATABASE** command. This can be done by copying the log files back into the active log paths or by supplying the **OVERFLOW LOG PATH** parameter on the **ROLLFORWARD DATABASE** command. If log archiving is being used ensure that the archive location is available. When the roll forward hits SQL2463N or SQL2464N that means the end of release is reached.
10. As instance owner, stop the instance by using **db2stop**.
11. Uncatalog all databases by using the **UNCATALOG DATABASE** command.
12. Log on the Db2 server as root on Linux and UNIX operating systems and move the prior release's `sqllib` directory.
13. As root on Linux and UNIX operating systems, upgrade the instance to Db2 Version 11.1 by using **db2iupgrade**.
14. As instance owner, start the instance using **db2start**.
15. Catalog all databases by using the **CATALOG DATABASE** command.
16. For all databases that need recovery, continue the roll forward done in step 11 until complete.

What to do next

See “Recovering through a Db2 server upgrade” on page 114.

Recovering through a Db2 pureScale server upgrade (Linux and UNIX)

Use this procedure to recover one or more databases through a Db2 pureScale Linux or UNIX server upgrade.

Before you begin

- Ensure that you have read and are familiar with the details and restrictions discussed in “Recovering through a Db2 server upgrade” on page 114.
- Make sure that the backup image that is used to perform recovery is accessible and previously validated.
- IBM Spectrum Scale levels stay at the levels used by Db2 Version 11.1.

- IBM Tivoli® System Automation for Multiplatforms (Tivoli SA MP) levels stay at the levels used by Db2 Version 11.1.

Procedure

1. IMPORTANT: As instance owner, protect the log files as in the next step you will be required to drop the database (in multi-database environments, apply this to all databases that require recovery). This step is necessary as some or all of the log files in the active or mirror log paths might not have been archived and the drop of the database deletes all of those log files. To get the current value of these log paths, use:
   ```
   db2 get db cfg for <database_name> | grep -i log
   ```
   Note: This copy must be retained until this recovery procedure is completed and a new online full database backup is completed and verified, which includes a copy of the log files.
2. Log on to the Db2 server as a user with **SYSADM** authority, issue list database directory to view the database path where the database is restored to (in multi-database environments, apply this to all databases that require recovery).
3. Log on to the Db2 server as a user with **SYSADM** authority, drop all databases that needs recovery in Db2 Version 11.1 by running the **DROP DATABASE** command.
4. As instance owner, stop the instance by using the following command.
   ```
   db2stop
   ```
5. Log on the Db2 server as root on Linux and UNIX operating systems and run the following command to determine your IBM Spectrum Scale file system name:
   ```
   <db2dir>/bin/db2cluster -cfs -list -filesystem
   ```
   where `DB2DIR` represents the installation location of your Db2 copy. The output of this command must be similar to the following:
   ```
   FILE SYSTEM NAME        MOUNT_POINT
   ----------------        -----------
   db2fs1                 /db2sd_20091027220651
   ```
6. Drop your Db2 Version 11.1 instance by running the **db2idrop** command. This command does not remove the database files nor IBM Spectrum Scale.
   ```
   <db2dir>/instance/db2idrop -g <instance_name>
   ```
7. Recreate your instance in the pre-Db2 Version 11.1 by running the **db2icrt** command. Use the same set of options to **db2icrt**, which was used for creating Db2 Version 11.1. Use the `MOUNT_POINT` output identified in step 5 as a parameter to `-instance_shared_dir`.
   ```
   <DB2DIR>/instance/db2icrt -d -m <member hostname> -mnet <member netname> -cf <CF hostname> -cfnet <cfnet cfnetname> -
   ```
8. Add additional members and an additional CF by using the **db2iupdt** command. To add additional members:
   ```
   <DB2DIR>/instance/db2iupdt -add -m <member hostname> -mnet <member netname> <instancename>
   ```
   To add another CF:
   ```
   DB2DIR>/instance/db2iupdt -add -cf <secondary CF hostname> -mnet <cf hostname> <instancename>
   ```
9. Restore the database manager configuration parameter values for each instance by using the **UPDATE DATABASE MANAGER CONFIGURATION** command.
10. As instance owner, start the instance. To start the Db2 instance:
    ```
    db2start
    ```

11. For each database that needs recovery, restore your databases from a pre-Db2 Version 11.1 backup by running the **RESTORE DATABASE** command to the database paths listed in step 2.
12. Roll forward all databases restored in step 11 to a point in time before the failure. Ensure that you supply the log files saved off in step 1 to the **ROLLFORWARD DATABASE** command. This can be done by copying the log files back into the active log paths or by supplying the **OVERFLOW LOG PATH** parameter on the **ROLLFORWARD DATABASE** command. If log archiving is being used ensure that the archive location is available. When the roll forward hits SQL2463N or SQL2464N that means the end of release is reached.
13. As instance owner, stop the instance. To stop the Db2 instance:
```
db2stop
```
14. Uncatalog all databases by using the **UNCATALOG DATABASE** command.
15. Log on the Db2 server as root on Linux and UNIX operating systems and move the prior release's `sqllib` and `sqllib_shared` directory.
16. As root on Linux and UNIX operating systems, upgrade the instance to Db2 Version 11.1 by using **db2iupgrade**:
```
<DB2DIR>/instance/db2iupgrade -g -u <fenced id> <instance owner name>
```
Then rebuild the contents of the network resiliency condition and response resources:
```
<DB2DIR>/bin/db2cluster -cfs -repair -network_resiliency -all
```
17. As instance owner, start the instance. To start the Db2 instance:
```
db2start
```
18. Catalog all databases using the **CATALOG DATABASE** command.
19. For all databases that need recovery, continue the roll forward done in step 12 until complete.

What to do next

See "Recovering through a Db2 server upgrade" on page 114.

Reversing Db2 server upgrade

Reversing Db2 server upgrade involves creating a plan using the steps in this procedure to fall back to the Db2 release from which you upgraded your server. There is no utility to fall back to a previous release of Db2 database after upgrading your Db2 server.

Performing an upgrade in a test environment will help you identify any issues with the process and avoid having to reverse the upgrade.

Before you begin

- Ensure that you have SYSADM authority, as well as root on Linux and UNIX operating systems or Local Administrator authority on Windows operating systems.
- Perform the following steps before upgrading your Db2 server:
  - Review upgrade recommendations and disk space requirements. See "Best practices for upgrading Db2 servers" on page 23 and "Disk space requirements for Db2 server upgrades" on page 21.
  - Take an offline full backup of all databases that you are going to upgrade. See "Backing up databases before or after upgrade" on page 32.

  - Back up all database manager configuration parameter values for each instance and all database configuration parameter values for each database. See "Backing up Db2 server configuration and diagnostic information" on page 34.
  - Perform other pre-upgrade tasks that apply to your environment. See "Pre-upgrade tasks for Db2 servers" on page 27.
- Keep your existing pre- Version 11.1 copy during server upgrade. To do this, select the **Install New** option to create a new copy when installing Version 11.1. Do not select the **Work with an existing** option and then choose a pre- Version 11.1 copy with the **upgrade** action that is available on Windows operating systems.
- Save all log files in the active log paths to another directory in case you want to rollforward through these log files after reversing the upgrade.
- Use the **db2cklog** utility to determine what log files in the active log paths are from the pre- Version 11.1 release. Record this information for later use if rolling forward after reversing the upgrade.

Restrictions

- This procedure applies only to Db2 server upgrade. It does not include Db2 clients.
- In partitioned database environments you must perform this procedure on all participating database partition servers. If you have several database partitions on a partition server, execute tasks at the database level, such as backup and restore, on each database partition.
- Additional upgrade restrictions apply. See "Upgrade restrictions for Db2 servers" on page 14. Review the complete list.

Procedure

To reverse a Db2 server upgrade, you need to perform the following steps:

1. Log on to the Db2 server as a user with SYSADM authority.
2. For all recoverable databases, save all log files in the active log paths (primary and mirror if defined) in case you want to rollforward through these log files after reversing the upgrade. This is required to protect log files during the **DROP DATABASE** command needed in the ensuing steps in the procedure.
3. Drop all databases in Db2 Version 11.1 by running the **DROP DATABASE** command.
4. Log on to the Db2 server as root on Linux and UNIX operating systems or a user with Local Administrator authority on Windows operating systems.
5. Drop your Db2 Version 11.1 instances by running the **db2idrop** command. This command does not remove the database files; you need to drop your databases before dropping your instances.
6. If you upgraded your pre- Version 11.1 instances to Version 11.1, re-create your instances in the pre- Version 11.1 by running the **db2icrt**. Then restore the database manager configuration parameter values for each instance using the **UPDATE DATABASE MANAGER CONFIGURATION** command.
7. For each pre- Version 11.1 instance, log on to the server as the instance owner and restore your upgraded databases from a pre- Version 11.1 offline full backup by running the **RESTORE DATABASE** command. You cannot upgrade your databases from Version 11.1 to pre- Version 11.1 release.

If you recreated the instances using the same instance owner they had before upgrade and you did not upgrade a database to a Db2 Version 11.1 instance, the database is still in pre-Db2 Version 11.1 release and you can access it by just re-cataloging it.

8. If you have recoverable databases and you want to rollforward through the log files you had before the upgrade, issue the **ROLLFORWARD DATABASE** command. Ensure that you supply the log files saved off in step 2 to the **ROLLFORWARD DATABASE** command. This can be done by copying the log files back in to the active log paths or by supplying the **OVERFLOW LOG PATH** parameter on the **ROLLFORWARD DATABASE** command.
   - If using Db2 Version 10.5 Fix Pack 7 or later, the **ROLLFORWARD DATABASE** command may return success or SQL2463N. Reissue the command with the STOP option to bring up the database at the end of the pre-Version 11.1 release.
   - If using Db2 Version 10.5 Fix Pack 6 or earlier, the ROLLFORWARD DATABASE command may return success or SQL1263N. Rename the error S*.LOG log file to a .NEW extension and reissue the command. If using a multi-partition or multi-member environment, you may need to repeat the same steps for each partition or member. If using log archiving additional steps, such as manually retrieving log files to disk, may be required to ensure that the rollforward utility does not replay the log file from Db2 Version 11.1.

Upgrade Db2 clients

Upgrading your Db2 database product to a new release might require upgrading your Db2 client.

Upgrading a client involves installing a Db2 Version 11.1 client copy and then upgrading the client instance. A client instance allows you to connect your application to a database and keeps the information about your client configuration, your cataloged nodes, and your cataloged databases.

The current level of client that you have installed determines the way to proceed with upgrade to Db2 Version 11.1. You can directly upgrade to Db2 Version 11.1 clients from Version 10.1, or Version 9.7. If you have Version 9.5 or earlier clients, upgrade to any Version 9.7 or Version 10.1 client first.

Note: It is recommended that the clients be at the same fix pack level as the server.

Review "Upgrade essentials for clients" for details about upgrade support and options available for clients.

Upgrade essentials for clients

Upgrading clients to Db2 Version 11.1 requires an understanding of upgrade concepts, upgrade options, upgrade restrictions, upgrade recommendations, and connectivity between clients and Db2 servers.

After you have a complete understanding of what upgrading your clients involves, you can create your own plan to successfully upgrade your clients to Db2 Version 11.1.

In the upgrading client topics, the term *pre-Db2 Version 11.1 clients* refers to Version 10.1, and Version 9.7 clients.

Upgrade options for clients

The upgrade options vary depending on the type of client that you want to install. The following table describes the upgrade options for each type of Db2 Version 11.1 client:

Table 15. Upgrade options for Db2 Version 11.1 clients

Upgrading from	**Upgrading to**	**Upgrade support details**
• Version 10.1 Data Server Client • Version 9.7 Data Server Client (Windows)	Db2 Version 11.1 Data Server Client(Windows)	You have two options: • Install the Db2 Version 11.1 Data Server Client, and choose a pre-Db2 Version 11.1 client copy with the **upgrade** action in the **Work with Existing** window. The client instance is then automatically upgraded for you. • Install a new copy of the Db2 Version 11.1 Data Server Client, and then manually upgrade existing client instances.
• Version 10.1 Data Server Runtime Client • Version 9.7 Data Server Runtime Client (Windows)	Db2 Version 11.1 Data Server Runtime Client(Windows)	• Install the Db2 Version 11.1 Data Server Runtime Client as a new copy, and then manually upgrade your existing client instance.
All Version 10.1, or Version 9.7 clients (Linux or UNIX)	All Db2 Version 11.1 clients (Linux or UNIX)	• Install a new copy of any Db2 Version 11.1 client, and then manually upgrade your existing client instance.

When you upgrade a client instance, the bit size is determined by the operating systems where you installed the Db2 Version 11.1 client. Refer to Table 12 on page 23 for details.

Upgrade restrictions for clients

Review "Upgrade restrictions for Db2 servers" on page 14 for information regarding instance upgrade and operating system support. These restrictions also apply to clients and can impact their upgrade.

Also, the trusted context capability supports only the TCP/IP protocol. Any connections to upgraded databases that you cataloged using a local node are unable to use this capability unless you recatalog the nodes using the TCP/IP protocol.

Connectivity support between clients and Db2 servers

In Db2 Version 11.1, the following support for connectivity between clients and Db2 servers is available:

Table 16. Db2 Version 11.1 connectivity support

Client	Db2 server	Client connectivity support
32-bit or 64-bit Db2 Version 11.1 clients	32-bit or 64-bit Db2 Version 11.1 server	Version 11.1 clients other than the IBM Data Server Driver for JDBC and SQLJ can establish 32-bit or 64-bit connections. For the IBM Data Server Driver for JDBC and SQLJ: • With type 4 connectivity, a 32-bit or 64-bit Java application can connect to a 32-bit or 64-bit server. • With type 2 connectivity – A 32-bit or 64-bit Java application can make a remote connection to a 32-bit or 64-bit server. – A 64-bit Java application can make a local connection to a 32-bit or 64-bit server. – A 32-bit Java application can make a local connection only to a 32-bit server.
32-bit or 64-bit Db2 Version 9.7 clients	32-bit or 64-bit Db2 Version 11.1 server	Only Db2 Version 9.7 or earlier functionality is available.
32-bit or 64-bit Version 10.1 clients	32-bit or 64-bit Db2 Version 11.1 server	Only Db2 Version 10.1 or earlier functionality is available.

Connections to Db2 Version 9.1 servers from a Version 11.1 client is supported. However, Db2 Version 9.1 reached end of support on April 30, 2012. For more support lifecycle information, see http://www-01.ibm.com/software/data/support/lifecycle/. For continued Version 9.1 support, a service extension is required.

Besides connectivity support, if you issue Db2 commands or SQL statements from a client to a Db2 server with a different version, you must be aware of incompatibilities between releases that can arise from changes in default behavior or restrictions lifted for these commands or SQL statements.

For example, if you issue the DESCRIBE command with the INDEXES FOR TABLE parameter from a Db2 Version 11.1 client, a pre-Db2 Version 11.1 server lists only relational indexes while a Db2 Version 11.1 Db2 server lists indexes over XML data and text search indexes in addition to relational indexes. Refer to “Upgrade impact from Db2 command changes” on page 141 and “Upgrade impact from SQL statement changes” on page 142 for details.

Best practices for upgrading clients

Consider the following best practices when planning your client upgrade.

Determine whether to upgrade clients or Db2 servers first

In general, upgrade clients after you upgrade your Db2 servers is the traditional approach. Supported pre-Db2 Version 11.1 clients can connect to Db2 Version 11.1 servers. However, the functionality introduced in releases after the pre-Db2 Version 11.1 client release is not available. If you plan to use this functionality in your applications, upgrade your clients to Db2 Version 11.1 or install new Db2 Version 11.1 client copies. See “Supported combinations of client and server versions” in *Installing IBM Data Server Clients* for details.

You can upgrade your clients before you upgrade your Db2 servers. However, you must ensure that your applications are able to manage any incompatibilities between releases. Review the following topics to determine if any incompatibilities apply to your application, and take necessary actions to manage these incompatibilities:

- "Upgrade essentials for database applications" on page 138 for changes to Db2 APIs, Db2 commands, and SQL statements
- "Db2 server behavior changes" on page 17 for changes on default values for existing registry variables, database and database manager configuration parameters
- "Deprecated or discontinued functionality that affects Db2 server upgrades" on page 20 for discontinued functionality not supported by Db2 Version 11.1 clients
- "Changed functionality" in Db2 Version 11.1 for additional changes between releases

Upgrade your clients in a test environment

Upgrading clients in a test environment allows you to determine if the upgrade can be successful and to address any problems that might occurred during the upgrade process. You can also test your database applications and determine if you must upgrade them to run successfully in Db2 Version 11.1.

If you are upgrading your clients first, upgrading clients in a test environment allows you to determine and manage any incompatibilities between releases to successfully run your applications on pre-Db2 Version 11.1 servers using Db2 Version 11.1 clients

Install a new client copy instead of upgrading existing client

If you have software that requires a pre-Db2 Version 11.1 client, install the Db2 Version 11.1 client as a new copy and keep your existing client copy to satisfy the software requirement. Then create a Db2 Version 11.1 client instance and keep your existing client instance with its configuration. You can select the option to create a new client instance during the installation, or you can manually create the client instance after installation.

Perform pre-upgrade and post-upgrade tasks

Perform the pre-upgrade and post-upgrade tasks for clients to ensure a successful upgrade.

Pre-upgrade tasks for clients

Before you upgrade your clients, you must complete certain tasks to help ensure that your upgrade is successful.

Procedure

Prepare for the upgrade of your clients by performing the following tasks:

1. Review the upgrade essentials for clients to determine which factors might impact your client upgrade.

   For more details, see "Upgrade essentials for clients" on page 122.
2. Review the supported and non-supported client configurations.
3. Plan your upgrade strategy.

For more information, see “Plan your Db2 environment upgrade” on page 1. For example, you might need to upgrade your Db2 server first, then your clients.

4. Optional: Upgrade your Db2 servers.

   For more information, see “Upgrade Db2 servers” on page 9.
5. Back up your client configuration information.

   For more information, see “Backing up client configuration information.”
6. On Windows operating system, identify the system or user ODBC data sources that are registered in the Microsoft ODBC Data Source Administrator by using the **db2cli** interactive tool.

   For user ODBC DSN, issue the following command:
   ```
   db2cli registerdsn -list
   ```
   For 64-bit ODBC DSN, issue the following command:
   ```
   db2cli registerdsn -list -system
   ```
   For 32-bit ODBC DSN, issue the following command:
   ```
   db2cli32 registerdsn -list -system
   ```
7. Optional: Upgrade your clients in a test environment to identify upgrade issues and to verify that applications, scripts, tools, and routines work as expected before you upgrade your production environment.

   For more information, see “Upgrading clients in a test environment” on page 127.

Backing up client configuration information

Before you upgrade your client, back up the database manager configuration parameter settings of your client instance and the information details about all of your cataloged databases. With this information, you can restore your previous client configuration and cataloged databases after upgrade, if necessary.

Before you begin

Ensure that you have SYSADM or SYSCTRL authority to run the **db2cfexp** command.

Restrictions

This procedure describes how to back up the configuration information for only one client. If you have different configuration settings on each client, you must back up the configuration information for each client.

Procedure

To back up your client configuration information:

1. Back up your database manager configuration parameter settings. Use the **GET DATABASE MANAGER CONFIGURATION** command to list your settings for the parameters and redirect the command output to a file as shown in the following example:
   ```
   db2 GET DBM CFG > D:\upgrade\dbm_client.cfg
   ```
2. Back up the information of cataloged databases to export your configuration profile.

Upgrading clients in a test environment

Upgrading clients in a test environment before you upgrade them in your production environment allows you to address problems during the upgrade process more effectively and to evaluate the impact of changes introduced in Db2 Version 11.1.

Before you begin

- You must have root user authority on Linux and UNIX operating systems or Local Administrator authority on Windows. You must also have SYSADM authority.

Restrictions

- On Linux and UNIX operating systems, you must not setup the instance environment for the root user. Running the **db2iupgrade** or the **db2icrt** command for a root user is not supported.

Procedure

To duplicate your production environment in a test environment, perform the following tasks:

1. Install the same client and version that you have in your production environment in a test system.
2. Re-create your client instance by running the **db2icrt** command with the **-s** option:

Operating system	Db2 command
Windows	`"%DB2PATH%"\bin\db2icrt -s client` *InstName*
Linux and UNIX	`$`*DB2DIR*`/instance/db2icrt -s client` *InstName*

   where **DB2PATH** and *DB2DIR* are set to the location of the client copy that you installed in the previous step, and *InstName* is the name of the instance.
3. Perform the pre-upgrade tasks that apply to your client.
4. Install a Db2 Version 11.1 client that you can upgrade to depending on the client that you are upgrading from. Select the **Install New** option to install a new copy. Refer to Table 15 on page 123 to determine what client product to install.
5. Upgrade your client instance by running the **db2iupgrade** command:

Operating system	Db2 command
Windows	`"%DB2PATH%"\bin\db2iupgrade` *InstName*
Linux and UNIX	`$`*DB2DIR*`/instance/db2iupgrade` *InstName*

   where **DB2PATH** and *DB2DIR* are set to the location of the Db2 Version 11.1 client copy that you installed in the previous step, and *InstName* is the name of the instance.
6. If you found any issues upgrading your test client instance, resolve these issues and add the tasks to resolve these issues to your upgrade plan.
7. Perform post-upgrade tasks that apply to your client.
8. Verify that the client upgrade was successful.
9. Test your applications, scripts, tools, and maintenance procedures by using the Db2 Version 11.1 client.

Upgrading to Data Server Client (Windows)

Upgrading an existing client copy to Db2 Version 11.1 requires that you install a Version 11.1 Data Server Client copy and then upgrade your client instance to retain your client configuration and to connect to all your previously cataloged databases.

Before you begin

- Ensure that you have SYSADM, SYSCTRL, or SYSMAINT authority and Local Administrator authority to run the **db2iupgrade** and the **db2icrt** commands.
- Review supported connectivity between Db2 clients and Db2 servers in upgrade essentials for Db2 clients.
- Perform pre-upgrade tasks for Db2 clients.

  Refer to “Pre-upgrade tasks for clients” on page 125.

About this task

When you install a Db2 Version 11.1 Data Server Client, you can choose to automatically upgrade an existing pre- Version 11.1 client copy.

Your existing client instances are upgraded to a new Db2 Version 11.1 Data Server Client copy and the existing pre- Version 11.1 client copy is removed.

You can also choose to install a new copy of Db2 Version 11.1Data Server Client and then manually upgrade your existing client instance after installation.

Restrictions

- The bit size of the client instance is determined by the operating system where you install a Db2 Version 11.1 client. The instance is 32-bit only in 32-bit Windows on x86 or x64. The instance is 64-bit only in 64-bit Windows on x64. Refer to Table 12 on page 23 for details.

Procedure

To upgrade from an existing client copy to a Db2 Version 11.1 Data Server Client on Windows:

1. Install Db2 Version 11.1 Data Server Client by running the **setup** command to launch the Db2 Setup wizard. You have three choices:
   - Select the **Work with Existing** option on the **Install a Product** panel. Then in the **Work with an existing** Db2 copy window, select a client copy name with action **upgrade**. The selected Db2 copy is removed and your client instance is upgraded. You can choose this option if you have an existing copy of Version 9.7 Data Server Client or Version 10.1 Data Server Client
   - Select the **Install New** option in the **Install a Product** panel. You should choose this option to create a new copy of Db2 Version 11.1 Data Server Client and keep your existing client copy. After installation, you must manually upgrade the client instance to run on the newly installed Data Server Client copy:
     - Log on to the system as a user with Local Administrator authority.
     - Run the **db2iupgrade** command:

       ```
       "%DB2PATH%"\bin\db2iupgrade InstName
       ```

where **DB2PATH** is set to the location that you specified during the Db2 Version 11.1 Data Server Client installation and *InstName* is the name of the instance.

- Select the **Work with Existing** option on the **Install a Product** panel. Then in the **Work with Existing** window, choose the client copy name with the **upgrade** action. Finally, in the **Select the installation, response file creation, or both** window, select the **Save my installation setting in a response file** option to create a response file for a response file installation. The response file has the required **UPGRADE_PRIOR_VERSIONS** keyword, the client copy name to upgrade, and the installation path.

  The result of the response file installation will be the same as in the first choice, all your client instances running on the selected client copy are automatically upgraded to the Db2 Version 11.1 Data Server Client copy. Using a response file installation to upgrade your clients can help you automate the upgrade process when you have a large number of clients.

2. If you want your applications to use the Db2 Version 11.1 Data Server Client copy through the default interface set the Db2 Version 11.1 Data Server Client copy as the Db2 default copy. See "Changing the default Db2 and default IBM database client interface copy after installation" in *Installing Db2 Servers*.
3. Optional: You can create a new Db2 Version 11.1 client instance instead of upgrading the existing client instance. You only need to create a new Version 11.1 client instance when you want to keep multiple client copies running on the same machine, or create a testing environment. To create a new client instance, run the **db2icrt** command with the option **-s**:

   ```
   "%DB2PATH%"\bin\db2icrt -s client InstName
   ```

   To create the same client connectivity environment you had, including the database manager configuration parameter and Db2 profile registry settings, run the **db2cfimp** command with the configuration profile that you save in the pre-upgrade tasks.
4. Compare the upgraded database manager configuration parameter values with the pre-upgrade values to ensure the changed values are compatible with your database applications.

What to do next

After upgrading your client, perform the recommended post-upgrade tasks for Db2 clients, especially verifying upgrade for clients to ensure that your client upgrade was successful. Refer to "Post-upgrade tasks for clients" on page 136 and "Verifying your client upgrade" on page 137.

Upgrading to Data Server Runtime Client (Windows)

Upgrading an existing Runtime Client copy to Version 11.1 requires that you install a Db2 Version 11.1 Data Server Runtime Client copy and then upgrade your client instance to retain your client configuration and to connect to all your previously cataloged databases

After you install a Db2 Version 11.1 Data Server Runtime Client copy, you can manually upgrade your existing client instance from a Version 10.1, or earlier Data Server Runtime Client.

Before you begin

- Ensure that you have SYSADM, SYSCTRL, or SYSMAINT authority and Local Administrator authority to run the **db2iupgrade** and the **db2icrt** commands.

- Review supported connectivity between clients and Db2 servers in "Upgrade essentials for clients" on page 122.
- Perform pre-upgrade tasks for clients.

  Refer to "Pre-upgrade tasks for clients" on page 125.

Restrictions

- The bit size of the client instance is determined by the operating systems where you install Db2 Version 11.1 client. The instance is 32-bit only in 32-bit Windows on x86 or x64. The instance is 64-bit only in 64-bit Windows on x64. Refer to Table 12 on page 23 for details.

Procedure

To upgrade from a Version 10.1, or earlier Db2 Runtime Client copy to Version 11.1 Data Server Runtime Client on Windows:

1. Install Db2 Version 11.1 Data Server Runtime Client. See "Installing IBM data server clients (Windows)" in *Installing IBM Data Server Clients*. Run the Db2 Setup wizard for all languages.
2. If you want your applications to use the Db2 Version 11.1 Data Server Runtime Client copy through the default interface or if you upgraded your existing Version 8 client copy, set the Version 9.7 copy as the default copy. See "Changing the default Db2 and default IBM database client interface copy after installation" in *Installing Db2 Servers*.
3. Log on to the system as a user with Local Administrator authority.
4. Upgrade your existing client instance by running the **db2iupgrade** command:
   ```
   "%DB2PATH%"\bin\db2iupgrade InstName
   ```
   where **DB2PATH** is set to the location that you specified during the Db2 Version 11.1 Data Server Runtime Client installation and *InstName* is the name of the instance.
5. Optional: You can create a new Db2 Version 11.1 client instance instead of upgrading an existing client instance. You only need to create a new Db2 Version 11.1 client instance when you want to keep multiple client copies running on the same machine. To create a new Db2 Version 11.1 client instance, run the **db2icrt** command with the option **-s**:
   ```
   "%DB2PATH%"\bin\db2icrt -s client InstName
   ```
   To create the same client connectivity environment you had, including the database manager configuration parameter and Db2 profile registry settings, run the **db2cfimp** command with the configuration profile that you saved in the pre-upgrade tasks.
6. Compare the upgraded database manager configuration parameter values with the pre-upgrade values to ensure the changed values are compatible with your database applications.

What to do next

After upgrading your client, perform the recommended post-upgrade tasks for clients, especially verifying upgrade for clients to ensure that your client upgrade was successful. Refer to "Post-upgrade tasks for clients" on page 136 and "Verifying your client upgrade" on page 137.

Upgrading clients (Linux and UNIX)

Upgrading existing clients to Db2 Version 11.1 requires that you install a Db2 Version 11.1 client copy and then upgrade your existing client instances to retain your client configuration and to connect to all your previously cataloged databases.

Before you begin

- Ensure that you have root user authority.
- Ensure that you have SYSADM, SYSCTRL, or SYSMAINT authority and root access to run the **db2iupgrade** and the **db2icrt** commands.
- Ensure that you meet the installation requirements for Db2 database products. Some operating systems require a 64-bit kernel.
- Review supported connectivity between clients and Db2 database servers in "Upgrade essentials for clients" on page 122.
- Perform pre-upgrade tasks for clients. Refer to "Pre-upgrade tasks for clients" on page 125.

Restrictions

- You can only upgrade from a Version 10.1, or Version 9.7 to a Version 11.1 Data Server Client.
- You can only upgrade from a Version 10.1, or Version 9.7 to a Version 11.1 Data Server Runtime Client.
- On Linux and UNIX except for Linux on x64, your existing 32-bit or 64-bit client instances are upgraded to Db2 Version 11.1 64-bit client instances. The bit size of the client instance is determined by the operating system where you install the Db2 Version 11.1 client. Refer to Table 12 on page 23 for details.
- On Linux and UNIX operating systems, you must not setup the instance environment for the root user. Running the **db2iupgrade** or the **db2icrt** command for a root user is not supported.

Procedure

To upgrade existing clients to Db2 Version 11.1 clients:

1. Install the appropriate Db2 Version 11.1 client as a new copy by running the **db2setup** command and select **Install New** on the Install a Product panel:
   - If you are upgrading from a Version 10.1, or earlier Data Server Client, install a new Version 11.1 Data Server Client.
   - If you are upgrading from a Version 10.1, or earlier Data Server Runtime Client, install a new Version 11.1 Data Server Runtime Client copy.
2. Log on to the system with root user authority.
3. Upgrade your existing client instances by running the **db2iupgrade** command:
   ```
   $DB2DIR/instance/db2iupgrade InstName
   ```
   where
   - *DB2DIR* is set to the location that you specified during the Db2 Version 11.1 client installation. The default installation path for UNIX is `/opt/IBM/db2/V10.5` and for Linux is `/opt/ibm/db2/V10.5`.
   - *InstName* is the login name of the client instance owner.
4. Optional: You can also create a new Db2 Version 11.1 client instance instead of upgrading the existing client instance. You only need to create a new Version

11.1 client instance when you want to keep multiple client copies running on the same machine. To create a new Version 11.1 client instance, run the **db2icrt** command with the option **-s**:

```
$DB2DIR/instance/db2icrt -s client InstName
```

where

- *DB2DIR* is set to the location that you specified during the Db2 Version 11.1 client installation.
- *InstName* is the login name of the instance owner.

To create the same client connectivity environment you had, including the database manager configuration parameter and Db2 profile registry settings, run the **db2cfimp** command with the configuration profile that you backed up in the pre-upgrade tasks.

5. Compare the upgraded database manager configuration parameter values with the pre-upgrade values to ensure that the changed values are compatible with your database applications.

What to do next

After upgrading your client, perform the recommended post-upgrade tasks for clients, especially verifying upgrade for clients to ensure that your client upgrade was successful. Refer to “Post-upgrade tasks for clients” on page 136 and “Verifying your client upgrade” on page 137.

Upgrading a network mounted client

When you upgrade a network-mounted client, you must apply the upgrade operations on a staging client workstation.

Before you begin

Upgrade the client environment on the staging client workstation that you used when you created the network-mounted client environment. If you use the same staging client workstation, the client configurations that you specified when you created the environment are transferred to the upgraded environment. However, if the client workstation that you used to create the network-mounted environment is not available, you can copy the `sqllib` directory on the network path to the `$HOME/sqllib` directory of a new staging client workstation.

If you do not want to replace binary files during the client upgrade, you must copy the new `sqllib` directory to a different shared network path. You can use the upgraded environment by setting the **DB2_NET_CLIENT_PATH** system environment variable to the path address where the new `sqllib` directory is located.

The staging client workstation that you use to create your client configurations must have the same operating system as the clients that are part of the network-mounted client environment.

Note: For C shell (csh) environments, use the **setenv** command instead of the **export** command to set environment variables. For more information about other command shells, see the topic "Setting environment variables outside the profile registries on Linux and UNIX operating systems" in the Db2 Knowledge Center.

Note: Do not run any instance commands in this instance-less client environment.

Procedure

To upgrade a network mounted client environment:

1. On the staging client workstation:
   a. Log in with a user ID that does not have root authority, but does have write access to the network directory.
   b. Use one of the following approaches:
      - If you are applying a new client fix pack, issue the **installFixPack** command. For more information about applying fix packs as a non-root user, see Applying fix packs to a non-root installation.
      - If you are upgrading to a new Db2 client version, issue the **db2setup** command. For more information about upgrading a Db2 client as a non-root user, see Upgrading non-root installations.

   c. Strip all setuid bits from the client installation by using the `chmod` command:

      ```
      chmod -R -s $HOME/sqllib
      ```

      For HP-UX, you must enable the variables LD_LIBRARY_PATH and SHLIB_PATH to search for Db2 executable files. Enable the variables by running the following **chatr** commands against the files in the `$HOME/sqllib/adm`, `$HOME/sqllib/bin`, and `$HOME/sqllib/security` directories:

      ```
      chmod u+w $HOME/sqllib/adm/*
      chatr +s enable $HOME/sqllib/adm/*
      chmod u-w $HOME/sqllib/adm/*

      chmod u+w $HOME/sqllib/bin/*
      chatr +s enable $HOME/sqllib/bin/*
      chmod u-w $HOME/sqllib/bin/*

      chmod u+w $HOME/sqllib/security/*
      chatr +s enable $HOME/sqllib/security/*
      chmod u-w $HOME/sqllib/security/*
      ```

   d. Configure the client environment, such as CLI parameters in the `db2cli.ini` file, the LDAP directory, updating the `db2dsdriver.cfg` file, Db2 registry variables, and database manager configuration parameters.
   e. Test your client configuration to ensure that it is working as you expect.
   f. Copy the `$HOME/sqllib` directory that you created to a shared network location on the code server. This location must be accessible to all client workstations in the network mounted client environment. You must use the `-R` and -Loptions with the `cp` command to copy all files in subdirectories to the network path and materialize all links as real files and directories:

      ```
      cp -R -L $HOME/sqllib <upgraded_network_dir>/sqllib
      ```

   g. (Optional) Grant a user group ownership of the `sqllib` network directory. You can change ownership of the `sqllib` directory by using the `chown` command.

      ```
      chown -R owner:group <upgraded_network_dir>/sqllib
      ```

      Where *owner* is the userid and *group* is the group that is known to all client hosts that share this `sqllib` directory.

2. For each user account that uses the upgraded network-mounted client:
   a. Set the **DB2_NET_CLIENT_PATH** environment variable to the shared network location where you copied the new `sqllib` directory by using the following command:

      ```
      export DB2_NET_CLIENT_PATH=<upgraded_network_dir>/sqllib
      ```

b. Use the `db2profile` and `db2cshrc` files in the path that is specified by **DB2_NET_CLIENT_PATH** to configure system environment variables that are used by the Db2 client. You can access these files by using the following commands:

```
. $DB2_NET_CLIENT_PATH/db2profile
```

Or on C shell, enter:

```
source $DB2_NET_CLIENT_PATH/db2cshrc
```

c. Set the **DB2_APPL_DATA_PATH** environment variable to a new local directory to store user-specific data and diagnostic files. For example, to set the application data path to the local directory `$HOME/db2`, run the following commands:

```
mkdir $HOME/db2
export DB2_APPL_DATA_PATH=$HOME/db2
```

d. If a user set the **DB2_APPL_CFG_PATH** to a local directory, run the **db2ccprf** command to copy the upgraded global configuration to a local directory. For example, if the local directory is `$HOME/db2`, run the following command:

```
db2ccprf -t $HOME/db2
```

e. Update **DB2_APPL_CFG_PATH** with the address of the local directory you specified in the **db2ccprf** command. For example, if the local directory is `$HOME/db2`, run the following command:

```
export DB2_APPL_CFG_PATH=$HOME/db2
```

f. (Optional) To apply all the environment variable settings detailed above automatically, define them in the .profile file.

Migrating and Upgrading to IBM Data Server Driver Package

Migrating and Upgrading to IBM Data Server Driver Package requires that you install a Db2 Version 11.1 DSDRIVER and optionally set the default client interface.

Before you begin

- Review supported connectivity between Db2 clients and Db2 servers in “Upgrade essentials for clients” on page 122.

About this task

There are two tasks in this topic. If you are using LDAP functionality in an instance based client environment, follow the steps in Migrating to IBM Data Server Driver Package from an instance based client. Otherwise, follow the steps in Upgrading to IBM Data Server Driver Package.

Upgrading to IBM Data Server Driver Package

Procedure

1. Install a Db2 Version 11.1 DSDRIVER copy.

   See “Installation methods for IBM data server clients” in *Installing IBM Data Server Clients* for details.

   - If there is no existing DSDRIVER installed, then install the latest version of the DSDRIVER. The new DSDRIVER will be installed to a new copy.
   - If there is one existing copy of the DSDRIVER:

    - If there is an existing DSDRIVER and a copy name is not provided for the new install, the default behavior is to install the DSDRIVER on top of that copy and upgrade it to the current level.
    - If there is an existing DSDRIVER and a copy name is provided in the install command line or in the response file (for the silent install) the DSDRIVER will be installed to that copy, whether it is a new copy, or an existing DSDRIVER copy.
  - If there are 2 or more existing DSDRIVER copies:
    - If one of the existing DSDRIVER copies is set as the default Db2 client interface copy:
      - If no copy name is provided during install, the DSDRIVER will be installed on top of the default client interface copy.
      - If a copy name is provided during install, the DSDRIVER will be installed to that copy, whether it is an existing copy or a new one.
    - If none of the existing DSDRIVER copies is set as the default Db2 client interface copy:
      - If no copy name is provided during install, the DSDRIVER install will be stopped with message DBI20006E Installing the IBM Data Server Driver Package failed because the installer could not determine whether to install a new copy or to upgrade an existing copy because no copy name was specified.
      - If a copy name is provided during install the DSDRIVER will be installed to that copy, whether it is an existing copy or a new one.

   Note: The installer will handle the case where the release level of the existing copy is higher than the current.
2. If you want your applications to use the Db2 Version 11.1 DSDRIVER copy through the default interface, set the Db2 Version 11.1 DSDRIVER copy as the Db2 client interface default. See "Changing the default Db2 and default IBM database client interface copy after installation" in *Installing Db2 Servers*.

Migrating to IBM Data Server Driver Package from an instance based client

Procedure

To migrate to IBM Data Server Driver Package from an instance based client, complete the following steps:

1. If you do not have an LDAP-enabled environment, skip this step. If you do have an LDAP-enabled environment and the LDAP server is used for storing database entries, refresh all the local database and node entries and add new entries from the LDAP server by issuing the following command:
   ```
   db2 refresh ldap immediate all
   ```
   Running this command ensures that any LDAP database information is now refreshed in the local catalog of the instance based client.
2. Create the DSDRIVER configuration file and populate it with the existing instance based client catalog information by issuing the following command:
   ```
   db2dsdcfgfill -i instance_name -o output_path
   ```
   where:
   - *instance_name* is the name of the database manager instance. To determine the name of the instance, run the **db2ilist** command. For more information, see db2ilist - List instances command.

- *output_path* is the path where the configuration file is saved. Choose any path other than the path where the instance based client is installed. This file is used later when the IBM Data Server Driver Package is installed

The **db2dsdcfgfill** command is found in one of the following paths:

- On AIX, HP-UX, Linux, or Solaris operating systems `$HOME/sqllib/bin`, where *$HOME* is the home directory of the instance owner.
- On Windows operating systems `DB2DIR/bin` directory, where *DB2DIR* is the installation location of the Db2 copy.

Starting with Db2 Version 11.1 Fix Pack 5, information about the LDAP databases stored in the local catalog is also automatically populated to the `db2dsdriver.cfg` file when you run the **db2dsdcfgfill** command, as in the following example:

```
<configuration>
<dsncollection>
   <dsn alias="EC205" name="STLEC1" host="INEC005.vmec.svl.ibm.com" port="446"/>
   <dsn alias="EC206" name="STLEC1"  host="INEC006.vmec.svl.ibm.com" port="446" ldap="1"/>
   <dsn alias="EC207" name="STLEC1"  host="INEC007.vmec.svl.ibm.com" port="446"/>
</dsncollection>
   <databases>
   <database name="STLEC1" host="INEC005.vmec.svl.ibm.com" port="446"/>
   <database name="STLEC1" host="INEC006.vmec.svl.ibm.com" port="446"/>
   <database name="STLEC1" host="INEC007.vmec.svl.ibm.com" port="446"/>
   </databases>
</configuration>
```

Note: The dsn alias EC206 is an LDAP cataloged database on an instance based client. Therefore the entry in the example above contains `ldap="1"`

3. Uninstall the Data Server client copy. For more information, see Uninstalling an IBM data server client.
4. Install a Db2 Version 11.1 Fix Pack 5 DSDRIVER copy. For more information, see IBM Data Server Drivers.
5. Copy the configuration file created in step 2 to the DSDRIVER configuration file location.

   To determine the DSDRIVER configuration file location, use the **db2cli validate** command.

   The DSDRIVER configuration file, `db2dsdriver.cfg`, is now created using the configuration file path you chose with the **-o** in the **db2dsdcfgfill** command from step 2 and is ready to connect.
6. Verify the connection to the LDAP directory by issuing the following command: `db2cli validate -dsn` *`dsn name`* `-connect -user` *`username`* `-passwd` *`password`* `.` For more information, see Validating IBM Data Server Driver Package.
7. If you have an LDAP-enabled environment, your `db2dsdriver.cfg` file contains an `<ldapserver>` section. To refresh the contents of the `db2dsdriver.cfg` file at any time, issue the following command: `db2cli refreshldap`.

What to do next

After migrating your IBM Data Server Driver Package, perform only the post-upgrade tasks for Db2 clients that apply. Refer to “Post-upgrade tasks for clients.”

Post-upgrade tasks for clients

After upgrading your clients, you should perform some post-upgrade tasks to ensure that your clients perform as expected and at their optimum level.

Procedure

Perform the following post-upgrade tasks that apply to your clients:

1. Manage changes in Db2 server behavior by modifying your settings where required. There are new registry variables, new configuration parameters, and new default values for registry variables and configuration parameters introduced in Db2 Version 11.1 that can impact the behavior of your application.

   For more details, see "Managing Db2 server behavior changes" on page 107.
2. Verify that upgrading your clients was successful.

   For more details, see "Verifying your client upgrade."

Verifying your client upgrade

When the upgrade of your client is complete, it is a good practice to run some tests in the new upgraded environment to verify that your client is working as expected. These tests can consist of running batch programs that connect to databases in a Db2 server or any programs or scripts that you use for benchmarking.

Before you begin

- Ensure that you have network connectivity from the client to the Db2 server.
- Ensure that the Db2 servers and instances are up and running.

Procedure

To verify that your client upgrade is successful:

1. Test connecting to all cataloged databases. The following example tests a connection to a remote database by issuing the **CONNECT** command:
   ```
   db2 CONNECT TO sample USER mickey USING mouse

   Database Connection Information

   Database server        = DB2/AIX64 10.5
   SQL authorization ID   = MICKEY
   Local database alias   = SAMPLE
   ```
   You need to specify a user ID and password when connecting to a remote database.
2. If you experience problems connecting to your cataloged database, use the **db2cfimp** tool and the configuration profile that you saved by performing the saving Db2 clients configuration pre-upgrade task to re-create the same client connectivity environment you had before upgrade.
3. Run your client database applications or scripts that connect to your databases to ensure they are working as expected.

Upgrade database applications, routines, tools, and scripts

Upgrading your Db2 database product to a new release might require upgrading your database applications, routines, tools, and scripts if changes in new release impacts them.

Upgrading your applications and routines involves the following actions:

- Test whether your applications and routines perform as expected in a Db2 Version 11.1 testing environment. You do not need to upgrade your applications and routines if they run successfully.
- If your applications or routines have errors that run in Db2 Version 11.1, you must do the following:
  - Review upgrade essentials for database applications to identify any changes in Db2 Version 11.1 that can impact your applications.
  - Review upgrade essentials for routines to identify any changes in Db2 Version 11.1 that can impact your routines.
  - Plan how to modify your applications and routines to handle these changes. Determine the steps that you must perform by reviewing the Upgrading database applications or Upgrading routines tasks.
  - Modify your applications and routines according to your plan.
  - Test your applications and routines in a Db2 Version 11.1 testing environment.
- Verify that your applications and routines perform as expected in your Db2 Version 11.1 production environment before deploying them.

If your applications and routines use any functionality that is deprecated in Db2 Version 11.1, you must remove the use of this functionality from your application code. The deprecated functionality will not be supported in Db2 Version 11.1 and may be discontinued in a future release.

Also, you must consider adopting new functionality available in Db2 Version 11.1 to enhance functionality and improve performance.

Upgrade essentials for database applications

Changes in application development support, new functionality, discontinued functionality, and deprecated functionality might impact your database applications, scripts and tools after you upgrade them to Version 11.1.

Operating system support

A complete list of supported operating systems is available at "Installation requirements for Db2 database products" in *Installing Db2 Servers*. If your current version of operating system is unsupported, you must upgrade it before you install Version 11.1.

In UNIX operating systems, only 64-bit kernels are supported. Your 32-bit instances are upgraded to Version 11.1 64-bit instances.

If you upgrade to the latest version of your operating system or you install a 64-bit kernel, rebuild all database applications and external routines after you upgrade to Version 11.1 so that they use the new runtime libraries in the operating system.

Development software support

Development software support has also changed. To improve performance and avoid technical support issues, rebuild your applications with the latest version of your development software. Review the changes in support for development software requirements. See "Support for elements of the database application development environment" in *Getting Started with Database Application Development*

Application drivers

The IBM Data Server Driver for JDBC and SQLJ includes the `db2jcc.jar` class file for applications that use JDBC 3.0 methods or earlier and the `db2jcc4.jar` class file for applications that use JDBC 4.0 or later methods or JDBC 3.0 or earlier methods. To manage the behavioral differences between the driver that supports JDBC 4.0 or later in Version 9.7 and previous releases of this driver, upgrade Java applications that use IBM Data Server Driver for JDBC and SQLJ. See "Upgrading Java applications that use IBM Data Server Driver for JDBC and SQLJ" on page 150 for details.

The Db2 JDBC Type 2 driver is discontinued in Version 10.1. You should modify your Java applications and external routines to use the IBM Data Server Driver for JDBC and SQLJ with type 2 connections.

To manage the behavioral differences between the version of IBM Data Server Driver for JDBC and SQLJ that support JDBC 3.0 and the Db2 JDBC Type 2 driver, upgrade your Java applications that use Db2 JDBC Type 2 driver. See Upgrading Java applications that use Db2 JDBC Type 2 driver for details.

See "Db2 and IBM Data Server Driver for JDBC and SQLJ levels" in *Installing Db2 Servers* for details about the versions of IBM Data Server Driver for JDBC and SQLJ that are delivered with every Db2 database product version and fix packs. See "JDBC differences between versions of the IBM Data Server Driver for JDBC and SQLJ" for details about the differences between those drivers.

CLI applications, Db2 CLP interface, and .Net Data Provider clients support Secure Sockets Layer (SSL). The IBM Global Security Kit (GSKit) provides encryption services for the Secure Sockets Layer (SSL) support. Refer to "Configuring Secure Sockets Layer (SSL) support in non-Java Db2 clients" in *Database Security Guide* for details about how to enable SSL in a client including how to download and install the GSKit.

Db2 APIs and Db2 commands

Review the following topics to determine if you have applications and scripts that are impacted by changes to Db2 APIs and Db2 commands in Version 11.1:

- Db2 API functions
- Db2 command line processor (CLP) and system commands

SQL statements

Review the changes to SQL statements in Version 11.1 to determine if you have applications and scripts that are impacted by these changes and how to manage these changes. Introduction of new functionality such as an untyped NULL keyword in expressions and a DEFAULT keyword in procedure parameters requires that you modify your applications to adapt to these changes.

System catalog views and built-in administrative routines and views

After database upgrade to Version 11.1, the system catalog views under the SYSCAT schema remain compatible with catalog views that you defined in previous releases. However, there are new columns, increases in column length, or columns with changed data types in some of the system catalog views.

SQL administrative routines include changes such as new parameters and new columns returned. Also, some routines are replaced with built-in administrative routines and views. In addition, all of the built-in table functions with names that start with SNAPSHOT_ are discontinued.

Review the following topics to determine if you have applications and scripts that are impacted by changes to system catalog views and built-in administrative routines and views:

- Some administrative routines are discontinued
- System catalog
- "Deprecated built-in administrative routines and their replacement routines or views" in *Administrative Routines and Views*

Optimizer and query execution plans

Rebind any statically bound packages after upgrade to take advantage of optimizer improvements.

Database packages

When you upgrade a database, all packages for user applications and routines are placed into an invalid state. Packages are also placed into an invalid state if they depend on database objects that you dropped, such as tables, views, aliases, indexes, triggers, referential constraints, and table check constraints. If you drop a UDF, your package is placed into an inoperative state.

Although invalid packages are automatically rebound by the database manager the first time that an application needs to access them, rebind your database packages to control when rebinding occurs and resolve any possible issues. See the Optimizer enhancements section for additional advantages of manually rebinding your database packages.

Db2 server behavior

In general, the Db2 server behavior is compatible between releases. However, there are changes in behavior to support new functionality or improve the performance of existing functionality. Review "Db2 server behavior changes" on page 17 to determine the impact of these behavior changes on your applications.

After upgrading your Db2 server, compare your registry variable and configuration parameter values to your values before upgrade, and change any values according to the needs of your applications.

Client connectivity support

Your applications can use pre-Version 11.1 clients to access databases in Version 11.1 servers. However, your applications are restricted to the functionality available for that client. Review "Upgrade essentials for clients" on page 122 to learn details about client connectivity and to identify changes in support that can impact your Db2 clients.

Upgrade of applications from Db2 Version 9.7

If you are upgrading from Db2 Version 9.7 , review changes in application driver support, 32-bit and 64-bit Db2 server support, and discontinued functionality between pre-Version 11.1 releases that might also impact your applications and scripts:

- Changes between Db2 Version 10.1 and Db2 Version 9.7 that impact applications.

Upgrade impact from Db2 API changes

The changes in Version 11.1 to Db2 APIs can impact your existing applications after you upgrade to Version 11.1.

The changes to Db2 APIs include new parameters, modifications to existing parameters, and deprecated or discontinued APIs. The following table lists the changes that impact your existing applications:

Table 17. Changes to Db2 APIs

Db2 API	Summary of changes with upgrade impact
db2DatabaseUpgrade	The COBOL and FORTRAN language support for the db2DatabaseUpgrade API is deprecated. For more information, see COBOL and FORTRAN language support for the db2DatabaseUpgrade API is deprecated for details.

Upgrade impact from Db2 command changes

The changes in Version 11.1 to Db2 command line processor (CLP) and system commands can impact your existing applications and scripts after you upgrade to Version 11.1.

The changes to commands include new parameters, modifications to existing parameters, deprecated or discontinued parameters, and modifications to command output. The following table lists the changes that impact applications and scripts:

Table 18. Changes to Db2 CLP and system commands

Command	Summary of changes with upgrade impact
db2cat, **db2exfmt**, **db2expln**	The output for the **db2cat**, **db2exfmt** **db2expln** commands now display information for random ordering of index keys. For more information, see Db2 command and SQL statement changes summary for details.
db2pd	The **-apinfo** parameter now displays additional information about current and past activities of the UOW. For more information, see Db2 command and SQL statement changes summary for details.
db2look	The **db2look** command now generates DDL statements to create column-organized tables in addition to row-organized tables. For more information, see Db2 command and SQL statement changes summary for details.
db2support	The **-d** parameter now supports collection of information from multiple databases. To specify multiple databases, separate the database names with a comma. For more information, see Db2 command and SQL statement changes summary for details.

Table 18. Changes to Db2 CLP and system commands (continued)

Command	Summary of changes with upgrade impact
db2IdentifyType1	The **db2IdentifyType1** command is discontinued. For more information, see Db2 command and SQL statement changes summary for details.
LOAD	For column-organized tables, automatic statistics collection occurs by default during the **LOAD REPLACE** command. To explicitly disable automatic statistics collection, specify the `STATISTICS NO` parameter. For more information, see Db2 command and SQL statement changes summary for details.
STATISTICS YES parameter of the **LOAD** command	The **STATISTICS YES** parameter of the **LOAD** command is discontinued. For more information, see Db2 command and SQL statement changes summary for details.

Upgrade impact from SQL statement changes

The changes to SQL statements in Version 11.1 can impact your existing applications and scripts after you upgrade to Version 11.1.

The changes to SQL statements include new default behaviors and modifications to statement output. In addition, some statements are changed, deprecated, or discontinued. The following table lists the changes that impact applications and scripts:

Table 19. Changes to SQL statements

SQL statement	Summary of changes with upgrade impact
CREATE TABLE statement	Because the default table organization that is specified by the **dft_table_org** database configuration parameter is ROW, there is no impact to the upgrade. After the upgrade, if you change the default organization to COLUMN or set the **DB2_WORKLOAD** registry variable to ANALYTICS before creating the database, scripts or applications that use the CREATE TABLE statement without the ORGANIZE BY COLUMN and ORGANIZE BY ROW clauses create tables with column table organization. Ensure to include the explicit clause to indicate the table organization as a good practice.

Refer to the *SQL Reference Volume 2* guide for details about any of the statements.

Upgrade impact from system catalog changes

In Version 11.1, system catalog objects are modified to support new functionality. These changes can impact your existing applications and scripts after you upgrade to Version 11.1.

System catalog views

The following system catalog views have changed in Version 11.1. Most modifications to catalog views consist of new columns, changed descriptions, changed column data types, and increased column lengths.

- SYSCAT.ATTRIBUTES catalog view
- SYSCAT.CHECKS catalog view
- SYSCAT.COLUMNS catalog view
- SYSCAT.CONTROLS catalog view
- SYSCAT.DATATYPES catalog view
- SYSCAT.INDEXCOLUSE catalog view
- SYSCAT.INDEXES catalog view
- SYSCAT.PACKAGES catalog view
- SYSCAT.ROUTINEPARMS catalog view
- SYSCAT.ROUTINES catalog view
- SYSCAT.ROWFIELDS catalog view
- SYSCAT.SERVICECLASSES catalog view
- SYSCAT.STOGROUPS catalog view
- SYSCAT.TABDEP catalog view
- SYSCAT.TABLES has a new column that is called TABLEORG to indicate the table organization
- SYSCAT.TABLESPACES catalog view
- SYSCAT.TRIGGERS catalog view
- SYSCAT.VARIABLES catalog view
- SYSCAT.VIEWS catalog view
- SYSSTAT.COLUMNS catalog view
- SYSSTAT.INDEXES catalog view
- SYSSTAT.TABLES catalog view

For more information, see .Some system catalog views, built-in functions and global variables, built-in administrative routines and views have been added and changed for details.

Built-in administrative views and routines

The following administrative views and routines have changed in Version 11.1. Most modifications consist of new columns, new values, changed column data types, and increased column lengths:

- ADMINTABINFO administrative view and ADMIN_GET_TAB_INFO table function
- ENV_SYS_INFO administrative view
- MON_BP_UTILIZATION administrative view
- MON_CONNECTION_SUMMARY administrative view
- MON_CURRENT_SQL administrative view
- MON_DB_SUMMARY administrative view
- MON_FORMAT_XML_COMPONENT_TIMES_BY_ROW table function
- MON_FORMAT_XML_METRICS_BY_ROW table function
- MON_FORMAT_XML_TIMES_BY_ROW table function

- MON_GET_APPL_LOCKWAIT table function
- MON_GET_BUFFERPOOL table function
- MON_GET_CONNECTION table function
- MON_GET_CONNECTION_DETAILS table function
- MON_GET_PKG_CACHE_STMT table function
- MON_GET_SERVICE_SUBCLASS table function
- MON_GET_SERVERLIST table function
- MON_GET_TABLE table function
- MON_GET_TABLESPACE table function
- MON_GET_TABLE_USAGE_LIST table function
- MON_TBSP_UTILIZATION administrative view
- MON_GET_UNIT_OF_WORK
- MON_GET_UNIT_OF_WORK_DETAILS
- MON_GET_WORKLOAD table function
- MON_GET_WORKLOAD_DETAILS table function
- MON_WORKLOAD_SUMMARY administrative view

For more information, see . Some system catalog views, built-in functions and global variables, built-in administrative routines and views have been added and changed for details.

In addition, all of the administrative routines with names that start with SNAPSHOT have been deprecated since Db2 Version 9.1. For more information, see . Some system catalog views, built-in functions and global variables, built-in administrative routines and views have been added and changed for details.

Review the list of the deprecated administrative routines and their replacement routines or views in “Deprecated SQL administrative routines and their replacement routines or views” in *Administrative Routines and Views* to determine additional changes that might impact your applications and scripts.

System catalog changes between pre-Version 9.7 releases

If you are upgrading from Db2 Version 9.7, the following additional system catalog changes between pre-Version 11.1 releases can also impact your applications and scripts:
- System catalog changes between Db2 Version 10.1 and Db2 Version 9.7.

Upgrade essentials for routines

Upgrade essentials describe changes in application development support, changes to support new functionality, unsupported functionality, and deprecated functionality that might impact your routines.

The changes described in “Upgrade essentials for database applications” on page 138 could also impact your routines.

Development software support
The information about development software support in “Upgrade essentials for database applications” on page 138 applies to external stored procedures and user-defined functions (UDFs).

Implicit casting

After function invocation, the database manager must decide which function in a group of like-named functions is the "best fit". A comparison of the data types of the arguments with the defined data types of the parameters of the functions under consideration forms the basis for this decision. An untyped parameter marker or an untyped NULL constant argument accepts any parameter type as a best fit.

This change to support implicit casting impacts function resolution that involves modified system built-in functions and any new functions that you create using these arguments.

XML data is passed by reference in SQL routines

In SQL routines, when you assign XML data to input and output parameters of XML type or local variables of XML type, the XML data is now passed by reference. In previous releases, the XML data was passed by value in SQL procedures. Therefore, some operations using XML data in SQL procedures can return results that are different from the results returned by the same operations in previous releases.

Unfenced external routines

During database upgrade to Db2 Version 11.1 on Linux and UNIX operating systems, all external unfenced routines that have no dependency on the Db2 engine libraries (`libdb2e.a` or `libdb2apie.a`) are altered to FENCED and NOT THREADSAFE so you can safely run these routines under the new multithreaded database manager. Running external routines defined as NOT FENCED and THREADSAFE in the new multithreaded database manager that are not thread safe can yield incorrect results, database corruption, or abnormal termination of the database manager. Refer to “Upgrading C, C++, and COBOL routines” on page 154 for details about how to manage this change.

31-bit external routines (Linux on zSeries)

All upgrade considerations for 32-bit external routines also apply to 31-bit external routines running on a Db2 database on Linux on zSeries.

Java external routines

The IBM Software Developer's Kit (SDK) for Java 1.4.2 is deprecated and might be discontinued in a future release.

For Db2 Version 9.7 or later, the default JDBC driver to run JDBC routines is the IBM Data Server Driver for JDBC and SQLJ. See “Upgrading Java routines” on page 156 for details on how to manage this change.

Pre-upgrade tasks for database applications and routines

Before you upgrade your database applications and routines, you should perform certain tasks to help you ensure a successful upgrade.

Procedure

Prepare for the upgrade of your database applications and routines by performing the following tasks:

1. Review upgrade essentials for database applications to determine which changes might impact your database applications.

   For more details, see “Upgrade essentials for database applications” on page 138.

2. Review upgrade essentials for routines to determine which changes might impact your routines.

   For more details, see “Upgrade essentials for routines” on page 144.
3. Plan your upgrade strategy.

   For more details, see “Plan your Db2 environment upgrade” on page 1.
4. Upgrade your operating system to a supported level if necessary.
5. Upgrade your development software to a supported level if necessary.
6. Perform benchmark tests on your database applications and routines in your production environment and save these baseline results to compare with benchmark test results after the upgrade.
7. Optional: Upgrade your client or install a Db2 Version 11.1 application driver if your application requires one.

   For more details, see “Upgrade Db2 clients” on page 122.

   Although Db2 Version 11.1 server provides connectivity support for earlier clients, using a Db2 Version 11.1 client eliminates any limitations and incompatibilities between releases.
8. Test your database applications in a Db2 Version 11.1 testing environment. If testing is successful, you do not need to upgrade your applications. However, review the upgrading database applications task and consider performing any steps that can help you improve performance.

   For more details, see “Upgrading Db2 servers in a test environment” on page 40 and “Upgrading database applications.”
9. Test your routines in a Db2 Version 11.1 testing environment. If testing is successful, you do not need to upgrade your routines. However, review the upgrading routines task and consider performing any steps that can help you improve performance.

   For more details, see “Upgrading Db2 servers in a test environment” on page 40 and “Upgrading routines” on page 153.

Upgrading database applications

Upgrading your existing database applications to Db2 Version 11.1 involves managing the changes between Version 11.1 and previous releases that impact these applications and verifying that these applications function as expected. Managing these changes might require that you modify your applications code and rebuild your applications.

You only need to modify your application code to manage changes in Db2 Version 11.1 that impact your applications, to remove the use of deprecated or discontinued functionality in Db2 Version 11.1, or to use new functionality.

Before you begin

- Ensure that you have access to a Db2 Version 11.1 server, including instances and databases. The Db2 server can be part of a testing environment.
- Ensure that you meet the installation requirements for Db2 database products.
- Ensure that the development software is at a version level that is supported by Db2 database products.
- Perform the pre-upgrade tasks for database applications. See “Pre-upgrade tasks for database applications and routines” on page 145.

Restrictions

This procedure only applies to database applications programmed in C, C++, COBOL, FORTRAN, Java, Perl, PHP, REXX, and .NET languages.

Procedure

To upgrade your database applications to Db2 Version 11.1:

1. If you identified changed Db2 commands, changed SQL statements, and changed system catalog views and built-in functions that impact your applications, edit your application code or scripts to modify:
   - Db2 CLP and system command syntax
   - SQL statements syntax
   - SQL statements using catalog views and SQL Administrative views and routines
   - SQL statements using target tables for write-to-table event monitors
   - User defined routine names that are not fully qualified with a schema name
   - Db2 API calls
   - Application programming interface calls such as JDBC, ODBC and CLI
   - If your applications or scripts read from the command output, modify them to read the changed output format.

   See “Upgrade impact from Db2 command changes” on page 141, “Upgrade impact from SQL statement changes” on page 142, and “Upgrade impact from system catalog changes” on page 142.
2. If you identified changes specific to the development environment that impact your applications, modify them to support these changes. See “Upgrade essentials for database applications” on page 138. Upgrade your:
   - Embedded SQL applications. See “Upgrading embedded SQL applications” on page 148.
   - CLI applications. See “Upgrading CLI applications” on page 149.
   - Java applications that use the IBM Data Server Driver for JDBC and SQLJ. See “Upgrading Java applications that use IBM Data Server Driver for JDBC and SQLJ” on page 150.
   - ADO and .NET applications. See “Upgrading ADO.NET applications” on page 151.
   - Scripts that use Db2 CLP commands and SQL statements. See “Upgrading scripts” on page 152.
3. Rebuild all changed database applications programmed in C/C++, COBOL, FORTRAN, and REXX, using the appropriate Db2 build file and specifying the appropriate Db2 shared library path.
4. Test your database applications to verify your changes and to ensure that they run as expected using Db2 Version 11.1.

What to do next

After upgrading your database applications, perform the recommended post-upgrade tasks for database applications to ensure that your upgrade was successful. See “Post-upgrade tasks for database applications and routines” on page 158.

Upgrading embedded SQL applications

Upgrading your existing embedded SQL applications to Db2 Version 11.1 involves managing the changes between Db2 Version 11.1 and previous releases that impact these applications and verifying that these applications function as expected.

Before you begin

- Ensure that you have access to a Db2 Version 11.1 server, including instances and databases. The Db2 server can be part of a testing environment.
- Ensure that the C, C++, COBOL, FORTRAN, or REXX development software is at a version level that is supported by Db2 database products.
- Perform previous steps in the upgrading database applications task. See "Upgrading database applications" on page 146.

Restrictions

This procedure only applies to database applications programmed in C, C++, COBOL, FORTRAN, and REXX.

Procedure

To upgrade your embedded SQL applications to Db2 Version 11.1:

1. If you modified the library path environment variables, ensure that those variables include the correct Db2 shared library path for your applications . The environment variables listed in this table specify additional paths to enable your applications to find the appropriate Db2 shared library at runtime (in most cases).

   On the Linux operating system: if you link an application using the RPATH link option without also specifying the RUNPATH link option, the **`LD_LIBRARY_PATH`** environment variable will be ignored at application run time, which can cause your application to fail.
2. Test your embedded SQL applications in a Db2 Version 11.1 testing environment. If testing is successful, you do not need to perform any additional steps.
3. If you bound your embedded applications using the **`BIND`** command with the **`BLOCKING ALL`** or **`BLOCKING UNAMBIGIOUS`** clause to enable the blocking of cursors for LOB columns, ensure that the **`instance_memory`** or **`database_memory`** database configuration parameters are set to `AUTOMATIC` or increase their numeric value to account for the extra memory usage. If you cannot increase these database configuration parameters, you have the following options:
   - Rebind them using the **`BIND`** command specifying **`BLOCKING NO`** or precompile them using the **`PRECOMPILE`** command specifying the **`SQLRULES STD`** command parameter. The **`BLOCKING NO`** clause disables blocking of all cursors in the application. The **`SQLRULES STD`** command parameter might have other effects than disabling blocking cursors.
   - Modify the application source code and declare the cursor with the FOR UPDATE clause to disable blocking.
4. To explicitly specify the correct Db2 shared library path for your applications, do one of the following:
   - If the application source code is available, rebuild the application. Specify the required Db2 shared library path. This is the best option.
   - Create a wrapper script to run your application. In the wrapper script, explicitly set the library path environment variable to the required Db2 shared library path.

- If you do not have the original source code available, run the **db2chglibpath** command to update the embedded runtime library path within the binary code of your application. This command is provided as-is and should therefore be considered a last resort.

What to do next

After upgrading your embedded SQL applications, perform the remaining steps in the upgrading database applications task. See "Upgrading database applications" on page 146.

Upgrading CLI applications

Upgrading your existing CLI applications to Db2 Version 11.1 involves managing the changes between Db2 Version 11.1 and previous releases that impact these applications, such as operating system support changes, development software support changes, the bit-width of the application, and the bit-width of the Db2 instance on which you deploy the applications.

Before you begin

- Ensure that you have access to a Db2 Version 11.1 server, including instances and databases. The Db2 server can be part of a testing environment.
- Ensure that the C and C++ development software is a version that is supported by Db2 database products. For details, see "C and C++ development software".
- Perform previous steps in the "Upgrading database applications" on page 146 task.

Restrictions

This procedure only applies to database applications programmed in C or C++ using the CLI interface.

Procedure

To upgrade your CLI applications to Db2 Version 11.1:

1. If you modified the library path environment variables, ensure that those variables include the correct Db2 shared library path for your applications, as shown in "Upgrade essentials for database applications" on page 138. You can use the environment variables listed in this table to specify additional paths that enable your applications to find the appropriate Db2 shared library at run time (in most cases).

   On Linux operating systems only: If you link an application using the RPATH link option without also specifying the RUNPATH link option, the **LD_LIBRARY_PATH** environment variable is ignored at application run time, which can cause your application to fail.
2. If you have set the **CLISchema** configuration keyword in your `db2cli.ini` file, set the **SysSchema** configuration keyword instead. The **CLISchema** configuration keyword is discontinued since Db2 Version 9.5.
   ```
   SysSchema = alternative schema
   ```
3. Test your CLI applications in a Db2 Version 11.1 testing environment. If testing is successful, you do not need to perform the remaining steps.
4. If you set the **BlockLobs** CLI configuration keyword to 1 and your application gets the error message SQL0973N, perform one of the following actions:
   - Set the **database_memory** configuration parameter to `AUTOMATIC`. This is the best option.

   - Reset the **BlockLobs** CLI configuration keyword to 0.
   - Bind LOB values directly to buffers instead of using LOB locators.

   Your client requires more memory to receive LOB data because this cursor blocking setting using the **BlockLobs** keyword sends all the LOB values immediately to your client after the row data is sent.
5. Review "CLI and ODBC function summary" in *Call Level Interface Guide and Reference Volume 2* to determine if you are using any of the deprecated functions in ODBC 3.0 and modify your application to use the replacement function instead. Although this version of CLI continues to support these functions, using the replacement functions ensures that your applications conform to the latest standards.
6. Explicitly specify the correct Db2 shared library path for your applications by performing one of the following actions:
   - If the application source code is available, rebuild the applications. Specify the required Db2 shared library path as shown in "Upgrade essentials for database applications" on page 138. This is the best option.
   - Create a wrapper script to run your applications. In the wrapper script, explicitly set the library path environment variable to the required Db2 shared library path as shown in "Upgrade essentials for database applications" on page 138.
   - If you do not have the original source code available, run the **db2chglibpath** command to update the embedded runtime library path within the binary code of your applications. This command is provided as-is and should therefore be considered a last resort.

What to do next

After upgrading your CLI applications, perform the remaining steps in the "Upgrading database applications" on page 146 task.

Upgrading Java applications that use IBM Data Server Driver for JDBC and SQLJ

Upgrading Java applications that use previous releases of the IBM Data Server Driver for JDBC and SQLJ involves managing the changes between different releases of this driver and the changes in Db2 Version 11.1 that can impact these applications.

Before you begin

- Review the upgrade essentials for applications to identify key changes that might impact your Java database applications. See "Upgrade essentials for database applications" on page 138.
- Ensure that you have access to a Db2 Version 11.1 server, including instances and databases. The Db2 server can be part of a testing environment.
- Ensure that the Java application development software and IBM Data Server Driver for JDBC and SQLJ are at a version level that is supported by Db2 database products.
- Perform the previous steps in the upgrading database applications task. See "Upgrading database applications" on page 146.

Restrictions

- This procedure applies only to Java applications using the IBM Data Server Driver for JDBC and SQLJ.

Procedure

To upgrade your Java database applications using the IBM Data Server Driver for JDBC and SQLJ to Db2 Version 11.1:

1. Install the version of the IBM Data Server Driver for JDBC and SQLJ that corresponds to the version and fix pack level of your Db2 copy. See "Java software support for Db2 products" in *Installing Db2 Servers* for a complete list of supported drivers.
   - If you use methods in JDBC 4.0 or earlier specifications in your applications, install IBM Data Server Driver for JDBC and SQLJ Version 4.13 or later.
   - If you use methods in JDBC 3.0 or earlier specifications in your applications, install IBM Data Server Driver for JDBC and SQLJ Version 3.63 or later
2. Adjust your applications to manage the differences between the current version of the IBM Data Server Driver for JDBC and SQLJ and the previous versions.
3. If you changed your Java application source code, rebuild your Java application. Refer to one of the following tasks in *Developing Java Applications* for details on how to rebuild them:
   - Building JDBC applications
   - Building SQLJ applications

Results

Upon completion of this task, your Java application should perform successfully using Db2 Version 11.1.

What to do next

After upgrading your Java applications, perform the remaining steps in the upgrading database applications task. See "Upgrading database applications" on page 146.

Upgrading ADO.NET applications

Upgrading your existing ADO.NET applications to Db2 Version 11.1 involves managing the changes between Db2 Version 11.1 and previous releases that impact these applications and verifying that these applications function as expected.

Before you begin

You do not have to upgrade ADO.NET applications that use the OLE DB .NET Data Provider or the ODBC .NET Data Provider to run with Db2 Version 11.1. However, upgrading these applications to the Data Server Provider for .NET can be beneficial for the following reasons:

- The Data Server Provider for .NET has a far more extensive set of APIs than the OLE DB and ODBC .NET data providers.
- Access to the Db2 database development productivity tools integrated with Visual Studio.
- Use of the Data Server Provider for .NET can bring significant performance improvements.
- Ensure that you have access to a Db2 Version 11.1 server, including instances and databases. The Db2 server can be part of a testing environment.
- Ensure that a supported version of the Microsoft .NET Framework software is installed on the Db2 database client computer.See "Supported .NET development software" in *Developing ADO.NET and OLE DB Applications*.

- Perform the previous steps in the "Upgrading database applications" on page 146 task.

Procedure

To upgrade your ADO.NET applications to Db2 Version 11.1:

1. Review the support for the Data Server Provider for .NET and how to program your applications to use the Data Server Provider for .NET and determine what changes to make on your ADO.NET applications.
2. Rebuild your ADO.NET applications to use the Data Server Provider for .NET.

What to do next

After upgrading your ADO.NET applications, perform the remaining steps in the "Upgrading database applications" on page 146 task.

Upgrading scripts

Upgrading your existing scripts that use Db2 command line processor (CLP) commands, Db2 system commands or SQL statements involves managing the changes between Version 11.1 and previous releases related to SQL statements, Db2 CLP and system commands, SQL Administrative views and routines, built-in functions, and catalog views.

Before you begin

- Ensure that you have access to a Db2 Version 11.1 server, including instances and databases.
- Ensure that a Db2 Version 11.1 client is installed.
- Perform the previous steps in the upgrading database applications task.

Restrictions

This procedure only applies to scripts that use Db2 CLP commands, Db2 system commands or SQL statements.

Procedure

To upgrade your scripts with Db2 CLP commands to Db2 Version 11.1:

1. Run your scripts to detect any incompatibilities with Db2 Version 11.1. If your scripts run successfully, you do not need to perform any additional steps. However, consider performing the remaining steps to remove deprecated functionality in Db2 Version 11.1 before they become discontinued or to use new command functionality.
2. Remove the Db2 CLP and system commands that display or update registry variables and configuration parameters that are deprecated or discontinued:
   - Deprecated and discontinued registry variables in "Deprecated and discontinued registry variables" on page 18
   - Deprecated and discontinued database manager configuration parameters in "Deprecated and discontinued database manager configuration parameters" on page 18
   - Deprecated and discontinued database configuration parameters in "Deprecated and discontinued database configuration parameters" on page 19

3. If your scripts perform snapshot or event monitoring, you need to modify your scripts to remove references to discontinued monitor elements or use a new name when they have been replaced by a new monitor element.
4. Determine the upgrade impact from system catalog changes. See "Upgrade impact from system catalog changes" on page 142. Using the changed views and routines requires that you:
   - Change the view names on your queries.
   - Change column names in your queries for columns that have been renamed in the view or routine.
   - Remove column names from your queries for columns that are not available in the view or result sets from routines.
   - Replace * in your queries for a specific list of column names that you want to receive as a result set because the changed view result set has additional columns.
   - Change routines names and parameter names, and indicate new additional parameters.
   - Modify your script to process additional columns in a result set when calling a changed routine or querying a changed view that returns additional columns.
5. Test your scripts to ensure that they run as expected using Db2 Version 11.1.

What to do next

After upgrading your scripts, perform the remaining steps in the upgrading database applications task. See "Upgrading database applications" on page 146.

Upgrading routines

Upgrading your existing routines to Db2 Version 11.1 involves managing the changes between Db2 Version 11.1 and previous releases that impact these routines and verifying that they function as expected. Managing these changes might require that you modify your routine code, rebuild your external routines, re-create your external routines in the database, and re-create SQL routines.

Test your routines in a Db2 Version 11.1 testing environment. If they run successfully, you are not required to change them. You only need to modify your routines to manage any changes between releases, to remove the use of discontinued or deprecated functionality in Db2 Version 11.1, or to use new functionality.

Before you begin

- Review upgrade essentials for routines to identify any changes that apply to your routines. See "Upgrade essentials for routines" on page 144.
- Ensure that you have access to upgraded Db2 Version 11.1 databases. These can be test databases.
- Ensure that you meet the installation requirements for Db2 database products. See "Installation requirements for Db2 database products" in *Installing Db2 Servers*.
- Ensure that the development software is at a version level that is supported by Db2 database products.
- Perform the pre-upgrade tasks for routines. See "Pre-upgrade tasks for database applications and routines" on page 145.

- Ensure that you have the necessary authorizations and privileges to use the ALTER FUNCTION or ALTER PROCEDURE statements. The authorizations allowed are listed in the *SQL Reference Volume 2*.

Restrictions

This procedure only applies to SQL routines and external routines programmed in C/C++, COBOL (procedures only), Java, and .NET languages.

Procedure

To upgrade your routines to Db2 Version 11.1 databases:

1. If you identified changes in Db2 Version 11.1 that impact your routines, edit your routine code and modify:
   - SQL statement syntax
   - SQL statements using SQL Administrative views and routines, built-in routines, and catalog views
   - User defined routine names that are not fully qualified with a schema names
   - Application programming interface calls such as JDBC and CLI
2. If you identified changes specific to the development environment that impact your routines, modify them to support these changes. Upgrade your:
   - C, C++, and COBOL routines. See "Upgrading C, C++, and COBOL routines."
   - Java routines. See "Upgrading Java routines" on page 156.
   - .NET CLR routines. See "Upgrading .NET CLR routines" on page 157.
3. Rebuild all changed external routine libraries or if you performed operating system or development software upgrades.
4. Test your routines to verify your changes and to ensure that the routines run as expected using Db2 Version 11.1.

What to do next

After upgrading your routines, perform the recommended post-upgrade tasks for routines. See "Post-upgrade tasks for database applications and routines" on page 158.

Upgrading C, C++, and COBOL routines

Upgrading your existing C, C++, or COBOL routines to Db2 Version 11.1 involves managing the changes between Db2 Version 11.1 and previous releases that impact these routines and verifying that they function as expected.

Before you begin

- Ensure that you have access to a Db2 Version 11.1 server, including instances and databases. The Db2 server can be part of a testing environment.
- Ensure that the C, C++, or COBOL routine development software are at a version level that is supported by Db2 database products by reviewing the following requirements:
  - "Support for external routine development in C" in *Administrative Routines and Views*
  - "Support for external routine development in C++" in *Administrative Routines and Views*

- "Support for external procedure development in COBOL" in *Administrative Routines and Views*

- Ensure that you have the necessary authorizations and privileges to use the ALTER FUNCTION or ALTER PROCEDURE statements. The authorizations allowed are listed in the *SQL Reference Volume 2*.
- Perform the previous steps in the upgrading routines task. See "Upgrading routines" on page 153.

Restrictions

This procedure only applies to external routines programmed in C/C++, and COBOL (procedures only).

Procedure

To upgrade a C, C++, or COBOL routine to Db2 Version 11.1, do the following:

1. If you upgraded to a Db2 Version 11.1 64-bit instance, change your routine libraries or routine definitions according to the following table:

Table 20. Upgrading C, C++, and COBOL routines to a Db2 Version 11.1 64-bit instance

Routine definition	**Action**
unfenced 32-bit routine library that use the Db2 engine library	Rebuild the routine source code into a 64-bit library using the Db2 Version 11.1 `bldrtn` script and redeploy the library to the Db2 server. If LOB locators are referenced in the routine, you must rebuild your routines. You can determine most of the routines that reference lob locators by executing the following query: `SELECT DISTINCT a.routineschema, a.routinename, a.specificname FROM syscat.routines a, syscat.routineparms b WHERE a.specifIcname = b.specificname AND b.locator = 'Y' AND a.fenced = 'N'` An advantage of this approach is that using a 64-bit library results in better routine runtime performance than using a 32-bit library.
fenced 32-bit routine library	• Rebuild the routine source code into a 64-bit library using the Db2 Version 11.1 `bldrtn` scripts and redeploy the library to the Db2 server. • If you cannot rebuild your routines, define the routine as not threadsafe using the ALTER PROCEDURE or ALTER FUNCTION statement with the NOT THREADSAFE clause.

   If none of the previously mentioned situations apply, you do not need to change your routine libraries or routine definitions.
2. If you are using the cursor blocking and found any differences in the behavior of your C, C++, or COBOL routines, review the "Upgrading embedded SQL applications" on page 148 task to learn how to manage those differences.
3. For routines that you did not rebuild but that you modified, rebind the routine packages to the target Db2 database. See "Rebinding packages in upgraded databases" on page 108.
4. Determine if the external routines that were altered during database upgrade or the external routines that use the Db2 engine libraries can safely run as NOT FENCED and THREADSAFE. If you have external unfenced routines in your database, the **UPGRADE DATABASE** command performs the following actions:
   - Returns the SQL1349W warning message and writes the ADM4100W message to the administration notification log.

- Redefines all your external unfenced routines that have no dependency on the Db2 engine library as FENCED and NOT THREADSAFE.
- Creates a CLP script called `alter_unfenced_`*`dbname`*`.db2` in the directory specified by the **`diagpath`** database manager configuration parameter to redefine the affected routines as NOT FENCED and THREADSAFE.

If you can safely run the external routines altered by database upgrade as NOT FENCED and THREADSAFE, you can redefine them as NOT FENCED and THREADSAFE using the original CLP script or a modified version with just specific routines that you want to redefine. If you can run them as FENCED and NOT THREADSAFE and the performance degradation that you experience is acceptable, you do not need to redefine your routines .

What to do next

After upgrading your C, C++, or COBOL routines, perform the remaining steps in the upgrading routines task. See "Upgrading routines" on page 153.

Upgrading Java routines

Upgrading your existing Java routines to Db2 Version 11.1 involves managing the changes between Db2 Version 11.1 and previous releases that impact these routines and ensure that these routines function as expected.

Before you begin

The following prerequisites must be met to perform this task:

- Ensure that you have access to a Db2 Version 11.1 server, including instances and databases. The Db2 server can be a test system.
- Ensure that the Java routine development software is at a version level that is supported by Db2 database products. See "Supported Java routine development software" in Developing User-defined Routines (SQL and External).
- Ensure that you are using supported Db2 drivers for JDBC and SQLJ APIs. See "Supported drivers for JDBC and SQLJ" in Developing Java Applications.
- Ensure that you have the necessary authorizations and privileges to use the ALTER FUNCTION or ALTER PROCEDURE statements. The authorizations allowed are listed in the *SQL Reference Volume 2*.
- Perform the previous steps in the upgrading routines task.

Procedure

To upgrade your Java routines:

1. Ensure the **`jdk_path`** database manager configuration parameter specifies the installation path of the IBM Software Developer's Kit (SDK) for Java that is installed on your Db2 server. Determine the current value of this parameter by issuing the following command:

   ```
   db2 GET DBM CFG
   ```

   By default the **`jdk_path`** database manager configuration parameter value is set during instance upgrade to the values shown in "Upgrade essentials for routines" on page 144 which are the installation path of SDK for Java 7.

   If you must use an SDK for Java other than the one installed in your Db2 Version 11.1 copy, set this configuration parameter to the installation path of an SDK for Java with the same bit width as the Db2 instance by updating the **`jdk_path`** parameter:

```
db2 UPDATE DBM CFG USING jdk_path SDKforJava-path
```

However, setting the **jdk_path** parameter to the installation path of SDK for Java 1.4.2 is not recommended because SDK for Java 1.4.2 is deprecated and might be discontinued in a future release.

2. Test your Java routines in your Db2 Version 11.1 database. If testing is successful and your Java routine perform as expected, you do not need to perform any additional steps.
3. If you found any differences in the behavior of your Java routines, review "Upgrading Java applications that use IBM Data Server Driver for JDBC and SQLJ" on page 150 to learn how to manage those differences.
4. If the pre-upgrade value of the **jdk_path** parameter was the installation path of SDK for Java 6 or Java 1.4.2, manage any differences in behavior between the SDK specified by the **jdk_path** parameter and the SDK for Java 7.
5. Explicitly define your Java routines as fenced using the ALTER FUNCTION or ALTER PROCEDURE statement with the FENCED clause. All Java routines run as fenced, regardless of how you defined them, but defining your Java routine definitions as fenced improves routine manageability and maintenance.
6. Optional: If your Java routine class is included within a JAR file that has been installed into a Db2 instance using a specific JAR file ID, ensure that the Java class is resolved more quickly by the Db2 database manager by specifying the JAR file ID as part of the EXTERNAL NAME clause in the routine definition. Use the ALTER PROCEDURE or ALTER FUNCTION statement to update the EXTERNAL NAME clause if required.
7. If you created projects in the Development Center to develop your Java routines, upgrade any existing projects to the IBM Data Studio using the upgrade wizard.

What to do next

After upgrading your Java routines, perform the remaining steps in the upgrading routines task.

Upgrading .NET CLR routines

Upgrading your existing .NET CLR routines involves managing the changes between Db2 Version 11.1 and previous releases that impact these routines and verifying that they function as expected.

Before you begin

- Review the "Upgrade essentials for routines" on page 144 to identify key changes that might apply to your .NET CLR routines.
- Ensure that you have access to a Db2 Version 11.1 server, including instances and databases. The Db2 server can be part of a testing environment.
- Ensure that a supported version of the Microsoft .NET Framework software is installed on the Db2 server.
- Perform the previous steps in the "Upgrading routines" on page 153 task.

Procedure

To upgrade your .NET CLR routines to Db2 Version 11.1:

1. Connect to the Db2 Version 11.1 database in which you defined the .NET CLR routines.
2. If you created your .NET CLR routines with execution control mode UNSAFE and you are upgrading from pre-Db2 Version 11.1 32-bit instance to Db2

Version 11.1 64-bit instance, rebuild their source code using the compile and link options specified in `bldrtn.bat`, the Db2 sample script for building .NET CLR routines.

   If you upgraded your .NET Framework, you should also rebuild your .NET CLR routines.
3. Deploy the routine assembly to the Db2 server in the same location specified by the EXTERNAL clause in the routine definition. The routines should function successfully, with no differences in between previous releases and Db2 Version 11.1.

What to do next

After upgrading your .NET CLR routines, perform the remaining steps in the "Upgrading routines" on page 153 task.

Post-upgrade tasks for database applications and routines

After upgrading your database applications and routines, you should perform several post-upgrade tasks to ensure that your database applications and routines perform as expected and at their optimum levels.

Procedure

Perform the following post-upgrade tasks that apply to your database applications and routines:
1. Perform benchmark tests on your database applications and routines in your production environment and compare with the baseline results that you saved before the upgrade.
2. Tune your database applications. Review important guidelines related to:
   - Character conversion
   - Optimization class
   - Isolation level
   - Locks and concurrency
   - Parallel processing for applications
   - Query optimization

   See related concepts for information about additional factors that can affect application performance.
3. Tune your routines. Review important guidelines related to:
   - Stored procedures
   - SQL procedures

   In addition, review guidelines on improving the performance of database applications that also apply to routines, such as the guidelines on optimization classes, locks, concurrency, and query tuning.
4. Remove dependencies on functionality that is deprecated in Db2 Version 11.1 in your database applications and routines before that functionality becomes discontinued.

   For more details, see "Deprecated or discontinued functionality that affects Db2 server upgrades" on page 20.
5. Adopt new Db2 Version 11.1 functionality in database applications, where appropriate, to improve performance or add new functionality. Check the Sample files to understand how the new functionality works.

For more details, see "Adopting new Version 11.1 functionality in database applications and routines."

Adopting new Version 11.1 functionality in database applications and routines

After upgrading to Version 11.1, enhance the functionality and improve the performance of your database applications by adopting new Version 11.1 functionality.

Before you begin

You must upgrade your Db2 server to Version 11.1.

Procedure

For applications that access upgraded databases, perform any of the following steps to adopt the specified Version 11.1 functionality:

- Enable the batch CALL statement support in CLI applications to optimize network flow by specifying an array size with the SQL_ATTR_PARAMSET_SIZE statement attribute and provide argument data in form of an array. For more details, see Calling stored procedures from CLI applications.
- Use NOT ENFORCED primary key and unique constraints to avoid performance costs during insert, update, and delete operations on a table and space requirements that are associated with enforcing a unique constraint when it is known that the data already conforms to the unique constraint. It also provides the same potential performance benefits for queries. Fore more details, see Informational constraints.

What to do next

If you upgraded from Db2 Version 9.7, adopt functionality introduced in Version 9.7 in your database applications and routines. See Adopting new Db2 Version 9.7 functionality in database applications and routines in the *Upgrading to Db2 Version 9.7* guide for details.

Index