IBM DB2 Universal Database

Administering Satellites Guide and Reference

Version 6

GC09-2821-00

IBM DB2 Universal Database

Administering Satellites Guide and Reference

Version 6

GC09-2821-00

Before using this information and the product it supports, be sure to read the general information under "Appendix G. Notices" on page 253.

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

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

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

© Copyright International Business Machines Corporation 1999. All rights reserved.
US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

© Copyright IBM Corp. 1999

About This Book

This book provides information necessary to use and administer the year 2000 ready DB2 Satellite Edition product, including information about:

- Setting up and maintaining a satellite environment
- Installing and migrating to DB2 Satellite Edition.

For information that supplements this book, refer to the following Web site:

`www.software.ibm.com/data/db2/library/satellite`

Who Should Use This Book

This book is intended primarily for administrators who implement and maintain a satellite environment. It can also be used by programmers who create applications that run on the satellites, as well as support personnel.

How This Book is Structured

This book is divided into three parts. The first part contains information about the satellite environment:

- "Chapter 1. The Satellite Environment" on page 3 provides an overview of the satellite environment, and how the different components of this environment interact.
- "Chapter 2. Batches" on page 15 describes how to create and modify the batches that set up and modify the configuration on the satellites.
- "Chapter 3. Authentication in the Satellite Environment" on page 45 describes how authentication works in the satellite environment.
- "Chapter 4. Cataloging Instances and Databases" on page 51 describes how to catalog the different DB2 targets against which the satellites must authenticate for synchronization.
- "Chapter 5. Setting up and Testing Your Satellite Environment" on page 63 describes how to set up and test the satellite environment.
- "Chapter 6. Working with the Model Office" on page 87 describes how you can use the model office during the different phases of the deployment, from the development phase through the post-deployment phase.
- "Chapter 7. DB2 Data Replication" on page 95 describes how to implement data replication between the satellite and a target DB2 system.

© Copyright IBM Corp. 1999

- "Chapter 8. Building Applications" on page 107 describes using the application programming interfaces that are specific to the satellite environment.
- "Chapter 9. Recovering the Satellite Environment" on page 129 describes the recovery considerations that apply to the satellite environment.
- "Chapter 10. Performing a Mass Deployment" on page 145 describes how to perform a mass roll out of your production satellites.
- "Chapter 11. Problem Determination" on page 161 describes typical problems, and how to recover from them.

The second part contains information about the installation and setup of DB2 Satellite Edition:

- "Chapter 12. An Introduction to Planning and Installation for the Satellite Environment" on page 179 describes the minimum tasks that you need to perform to set up the satellite environment.
- "Chapter 13. DB2 Control Server Planning and Installation" on page 181 describes how to install the DB2 control server, including prerequisites, and the steps that you perform to set up the DB2 control server after you install it.
- "Chapter 14. Planning for DB2 Satellite Edition Installation" on page 187 describes information that you should be familiar with before installing DB2 Satellite Edition.
- "Chapter 15. Installing DB2 Satellite Edition" on page 195 describes how to install DB2 Satellite Edition.
- "Chapter 16. DB2 Satellite Edition Distributed Installation" on page 199 describes how to create a response file and perform a distributed install.
- "Chapter 17. Migrating DB2 Satellite Edition from Previous Versions and Releases" on page 207 describes how to migrate previous versions of DB2 to DB2 Satellite Edition.

The third part contains the appendixes:

- "Appendix A. How the DB2 Library Is Structured" on page 225 describes the DB2 library.
- "Appendix B. DB2 Satellite Edition Differences" on page 241 describes the differences between DB2 Satellite Edition and DB2 Personal Edition.
- "Appendix C. db2sync - Start DB2 Synchronizer" on page 243 describes the **db2sync** command.
- "Appendix D. Administering DB2 Satellite Edition from the Control Center" on page 245 describes the Control Center functions that you can use to administer DB2 Satellite Edition.
- "Appendix E. Naming Rules" on page 247 describes the conventions for naming objects in a DB2 environment.

- "Appendix F. National Language Support (NLS)" on page 251 describes the code page and language support provided by DB2.

Part 1. Satellite Administration

© Copyright IBM Corp. 1999

Chapter 1. The Satellite Environment

The *satellite environment* is an environment in which you can administer large numbers of DB2 servers from a central control site.

In the satellite environment, you set up collections of DB2 servers. Each collection is known as a *group*. Each DB2 server that belongs to a group is called a *satellite*. You use groups to organize satellites that have shared characteristics into single entities. The characteristics that many satellites could share are the end-user application that runs on them, and the database definition that supports the application.

Note: For this document, *database definition* is the entire setup of DB2, including, but not limited to, the instance, the database manager configuration parameter values, the database design, and the database configuration parameter values.

In a group, the satellites are similar in terms of their database definition, usage, and purpose. For example, assume that there are two groups in your organization, Sales and Support. The Sales group would require one end-user application and database definition, while the Support group would require a different end-user application and database definition.

By grouping the satellites together, you administer small numbers of groups, which may contain hundreds, if not thousands of satellites, rather than having to administer each satellites individually. If you acquire additional DB2 servers to perform the same function as the satellites of an existing group, you can add them to that group. The administration solution provided by the satellite environment is fully scalable.

In the satellite environment, the setup and maintenance of any database definition is accomplished by sets of scripts known as *batches*. Because the database definition can be different for each group, each group has its own set of batches. The batches that are specific to a group are known as *group batches*.

© Copyright IBM Corp. 1999

Within a group, satellites may run different versions of the end-user application, and each version of the end-user application may require its own database definition and data. Group batches are always associated with a particular *application version*. The application version represents the database definition and data that support a particular version of the end-user application.

Each DB2 server that belongs to a group is known as a satellite. A satellite, because it belongs to a group, will have the same end-user application as the other satellites in the group. Depending on its version of the end-user application, however, it may share its database definition and data with only a subset of the satellites that belong to its group.

Information about the satellite environment is stored in a central database known as the *satellite control database*. This database records, among other things, which satellites are in the environment, the group each satellite belongs to, and which version of the end-user application a satellite is running. It also records the group batches that the satellites execute. This database is on a DB2 server that is known as the *DB2 control server*.

To set up and maintain its database definition, each satellite connects to the satellite control database to download the batches that correspond to its version of the end-user application. The satellite executes these batches locally, then reports the results back to the satellite control database. This process of downloading batches, executing them, then reporting the results of the batch execution is known as *synchronization*. A satellite synchronizes to maintain its consistency with the other satellites that belong to its group and that run the same version of the end-user application.

In the satellite environment, DB2 can be hidden from your end users: they do not need to learn about database administration. Instead, you can administer the satellite environment centrally from the *Satellite Administration Center*, which is a set of graphical tools that is available from the Control Center.

The DB2 Control Server

The DB2 control server contains the satellite control database, which records information about the satellite environment, and manages the synchronization process for satellites. You can use any supported DB2 server on any of the supported operating systems as a DB2 control server. This means that you can use tools such as the Control Center to administer it. You can also use the audit and trace facilities on this DB2 server.

Note: Currently, only servers running DB2 Universal Database Enterprise Edition on the Windows NT or AIX platform can function as a DB2 control server.

On Windows NT, the DB2 control server is created automatically when you install the DB2 Control Server component on your DB2 server as part of a custom installation. On AIX, you create the DB2 control server, after installing the DB2 Control Server component, by using the normal procedure for creating an instance. For more information about installation, see "Chapter 13. DB2 Control Server Planning and Installation" on page 181.

When the DB2 control server is created on Windows NT, the satellite control database, SATCTLDB, is also automatically created. On AIX, you create this database after creating the DB2 control server instance.

The Satellite Control Database

The information that configures and maintains satellites is stored in relational tables in the satellite control database, SATCTLDB. Because these tables are in an ordinary DB2 database, this information is accessible to authorized users, and can be maintained (for example, backed up and restored) using standard DB2 utilities. For information about these utilities (and other administrative issues), refer to the *Administration Guide*.

This book does not describe the tables in the satellite control database in detail. For information about these tables, refer to the following Web site:

```
www.software.ibm.com/data/db2/library/satellite
```

Although the tables in the satellite control database contain the information about satellites and the satellite environment, you should not attempt to maintain the environment by updating the tables directly. Instead, maintain the satellite environment by using the Satellite Administration Center, which is available from the Control Center.

Groups

Every satellite in the satellite environment must belong to a group. A group is a collection of satellites that share one or more characteristics. The shared characteristics will be things such as the business function of the satellite users, for example, selling life insurance, and the end-user application that is used to support the business function.

A satellite can only be associated with a single application version. However, within a group, not all satellites need to be associated with the same application version. This characteristic allows you to stage the deployment of the application. The different versions of the end-user application are supported by the group batches that are associated with different application versions. For additional information, see "Application Versions, Batches, and Scripts" on page 9.

Note: The term *group*, as used in the satellite environment, is *not* associated with operating system or security groups.

Satellites

A satellite is any DB2 server that is both a member of a group and synchronizes with a satellite control server to maintain its database definition and data.

Along with DB2, the satellite will run the business application that your end users require. The hardware on which DB2 and your application run can be any laptop or desktop computer on which any of the supported operating systems is running. Depending on how you decide to install DB2, the end user need not be aware that DB2 is installed on the satellite.

Note: Currently, only servers running DB2 Satellite Edition can be used as satellites.

There are two types of satellites in the satellite environment: *test satellites* and *production satellites*. You use test satellites to test the group batches that set up and maintain the database definition that supports the end-user application. When the group batches are fully tested, and produce satisfactory results, you promote them for use by the production satellites of the group.

Typically, you assign test satellites to your development environment. That is, the test satellites will not manage any active data that is required for business operations.

You assign production satellites to the users that you support. Unlike test satellites, production satellites are not used for testing purposes. Instead, these satellites execute the tested group batches that have been put into production, which results in them having a stable database definition to support the end-user application. Because production satellites have a stable database definition, they manage the active data that is required for business operations.

If you determine that the database definition on the production satellites is no longer adequate to support the requirements of your users, you modify the database definition on the test satellites to address the problem that is being reported. When you are satisfied with the results of the modification in your development environment, you make the modification available to the production satellites.

When you create a satellite, it is, by default, a production satellite. You must decide which satellites that you want to use as test satellites. This allows you to exercise full control over testing. For information about how to define a satellite as a test satellite, refer to the online help that is available from the Satellite Administration Center.

Note: Because test satellites are not used to maintain active data, you should not change them to production satellites during the lifetime of the application.

The Model Office

A model office is a special member of the test satellites in the group. Typically, you will have one model office for each version of the end-user application that you have deployed in the group. You can use a model office for a variety of purposes:

- To model your initial deployment of the group.

  When the model office is set up, you can create a response file that is based on the model office. You can use the response file to install large numbers of satellites. For more information, see "Chapter 10. Performing a Mass Deployment" on page 145.
- To test the changes that are required to the database definition and data that supports a version of the end-user application that is already in production.

  The model office provides a representative database, which you can work with using tools such as the Control Center and the Satellite Administration Center. Using the model office, you can generate scripts from the different SmartGuides, notebooks, and windows available in the Control Center. You verify the behavior of the scripts by submitting them to execute against the model office. This means that the production environment is almost entirely isolated from the changes that you make to the model office. When you are satisfied with the database definition changes that the scripts produce on the model office, you can promote the scripts to production. The changes will then be executed by the group's production satellites.

- To provide a representation of a typical satellite in the group.

  Because the model office represents a typical satellite, it can be useful for problem determination. If an end-user experiences a problem, you can use the model office, or a copy of it, to reproduce the problem and determine how to fix it.
- To deploy a new version of your end-user application. You can use the model office to verify that the installation of the new version of the application provides the correct results, and that the setup, update and cleanup batches produce the expected database definition and data for the end-user application. For more information, see "Create a Test System to Test the Deployment of the New Application" on page 157.

Figure 1 provides an overview of how the Control Center, the DB2 control server, and the model office can interact.

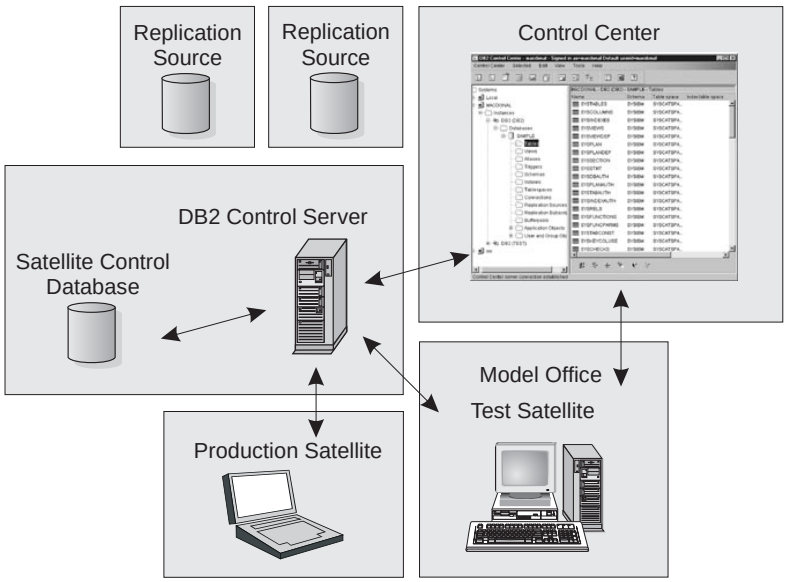


Figure 1. Model Office

For more information, see "Chapter 6. Working with the Model Office" on page 87.

Application Versions, Batches, and Scripts

Although the satellites of a group run the same end-user application, they do not have to run the same version of this application. Each version of the end-user application may require a database definition that is different from the other versions of the same application. To set up and maintain the database definition and the data to support a particular version of the end-user application, you use group batches for the application version. Each application version of a group is associated with its own batches.

Each batch is a collection of one or more batch steps. You create these batch steps to set up and maintain both the database definition and the data for the application version. The batch step is executed on the satellite when the satellite synchronizes.

A batch step is made of the following components:

- A *script*. The script can be a DB2 command, an SQL statement, or an operating system command.
- An *execution target*. The scripts that you create can execute against a DB2 instance, DB2 database, or on the operating system on the satellite. The DB2 instance, DB2 database, or operating system against which the script executes is known as an execution target.
- *Authentication credentials*. Before a script can execute against a DB2 instance or DB2 database, it must be authenticated. That is, the script requires the combination of a user ID and password so that the satellite can attach to the instance or connect to the database. This combination of user ID and password is known as authentication credentials.
- A *success code set*. The execution of a script is considered to be successful if its return code is within a set of return codes that you predefine for that script. This set of codes is known as a success code set.

The batch steps within a batch are always executed in the sequence in which they appear in the batch. When one batch step within a batch is executed successfully, the next batch step is executed. If, as defined by the success code set, an error occurs when a satellite is executing a batch step, that satellite stops executing its group batches, and reports an error back to the DB2 control server. When the error is fixed, the satellite can continue executing from the batch step that caused the error.

The application version is set at the satellite, typically during the installation and configuration of the end-user application on the satellite. When a satellite synchronizes, it reports its application version to its DB2 control server before it downloads and executes the scripts associated with the group batches for the application version.

For more information about batches, batch steps, and application versions, see "Chapter 2. Batches" on page 15.

Satellite Synchronization

The satellites of a particular group need to be in a state consistent with the other satellites of its group. The consistent state can be accomplished with synchronization.

Figure 2 provides a high-level view of how a satellite synchronizes. Before the satellite can synchronize, it must connect to the network on which the DB2 control server resides. The satellite can connect with the network in a variety of ways, such as by a dial-up program, by docking with the network, or by remaining permanently connected to the network.

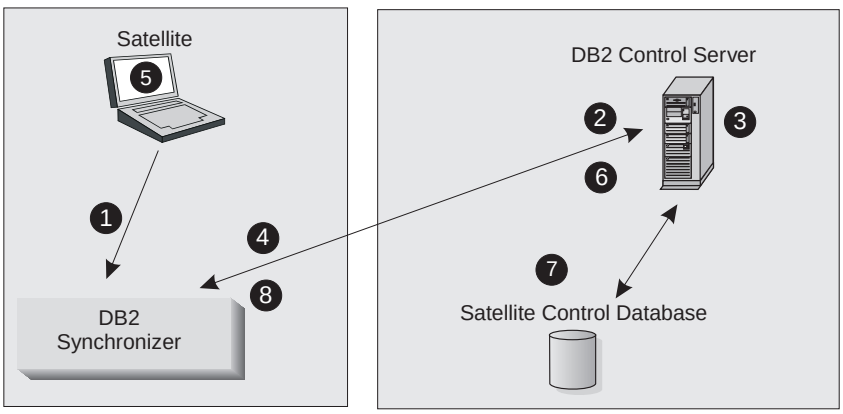


Figure 2. Satellite Synchronization

The satellite synchronizes as follows:

1. The end-user of the satellite invokes the synchronizer function. The invocation can be from your user application (if it calls the db2SyncSatellite API), or from the DB2 Synchronizer application that is provided with DB2. For information about the APIs that an application can call for synchronization, see "Synchronizing a Satellite" on page 117. Also refer to the *Administrative API Reference*. For information about the DB2 Synchronizer, see "Using the DB2 Synchronizer Application" on page 127.

   When the synchronizer function is invoked, steps 2 on page 11 through 8 occur automatically. No manual intervention is required.

   The satellite is only connected to the satellite control database for step 3 on page 11, and for step 7 on page 11.

2. The satellite connects to the satellite control database, where it is authenticated.
3. After authentication occurs, the DB2 control server checks which group the satellite belongs to, and the version of the application that the satellite is executing. The DB2 control server uses this information to determine which batches the satellite should execute, and which batch steps should be executed, if any. At this time, other events may also occur:
   a. If the satellite could not upload the results of its previous synchronization session to the satellite control database, the results are written at this time.
   b. If any of the scripts in a batch to be downloaded are parameterized, the DB2 control server instantiates the script with the values that are appropriate for the satellite. For more information about parameterized scripts, see "Parameterizing Scripts" on page 40.
   c. When steps 3a and 3b are complete (if required), the DB2 control server releases the scripts that the satellite is to execute, and the satellite downloads them. When this occurs, the tables in the satellite control database are updated to indicate that the satellite has obtained the batches that apply to it.
4. The synchronizer function drops the connection with the satellite control database.
5. The satellite executes the batches that it downloaded.
6. After executing the batches, the synchronizer function again connects to the satellite control database.
7. The synchronizer function updates the log information in the satellite control database with the results of the execution of the different steps of the batches. The log information provides details about the execution of the batch steps. For information about these logs, refer to the online help that is available from the Satellite Administration Center.
8. The synchronizer function drops the connection with the satellite control database.

After the synchronization session is complete, the satellite can be disconnected from the network, if required.

Group Administration

Within a group, satellites run the same end-user application, have a common interest in the same data, have similar database definitions, and perhaps the same execution environment. Typically, within a group, the satellites are used by people with the same occupation, such as selling life insurance.

The group can contain satellites that run a different version of the end-user application. Because the database definition and data that support each version of the application are set up and maintained by the batches of a specific application version, you can deploy different versions of the end-user application. This enables you to stage the deployment of a new version of the end-user application within the group.

Because satellites are organized by group, you administer at the group level, and not at the individual satellite level. This greatly simplifies administration. Instead of having to manage hundreds, if not thousands, of satellites separately, you manage the group to which they belong. The group batches that maintain the database definition and data for a particular version of an end-user application are associated with the application version. These group batches are organized into application versions for each version of the end-user application running on the satellites of the group.

When you create new satellites for the satellite environment, you add them to the group that is already running the end-user application that the new satellites will run. When these satellites synchronize for the first time, they will download and execute the group batches that apply to the version of the end-user application that they are running. You do not have to perform any special tasks to integrate these satellites into the environment. This means that the administration model that you use to set up and maintain the satellite environment is fully scalable. The groups that you set up can contain as many satellites as your business requires.

The Satellite Administration Center

The Satellite Administration Center is a collection of graphical tools that is available from the Control Center. You use the Satellite Administration Center to set up and maintain satellites, groups, and the batches that the satellites execute when they synchronize.

Before you can use the Satellite Administration Center, ensure that you add the DB2 control server and the satellite control database to the Control Center object tree. To perform this task, see “Chapter 4. Cataloging Instances and Databases” on page 51.

You can use the Satellite Administration Center when the Control Center detects that any of the instances available on it contain a satellite control database. You can open the Satellite Administration Center from the pop-up menu associated with the instance that contains the satellite control database, or from the pop-up menu associated with the satellite control database. You can also open it from the Control Center tool bar. For information about using the Satellite Administration Center, refer to the online help provided with it.

Note: You should only administer the satellite environment from the Satellite Administration Center.

The Satellite Environment Setup

Figure 3 shows a possible setup of the satellite environment. In the example, the development environment, which includes the Control Center, the DB2 control server and satellite control database, as well as the model office and test satellites, is almost entirely separate from the production environment. You use the development environment to both create and test the batches that you want the production satellites to execute.

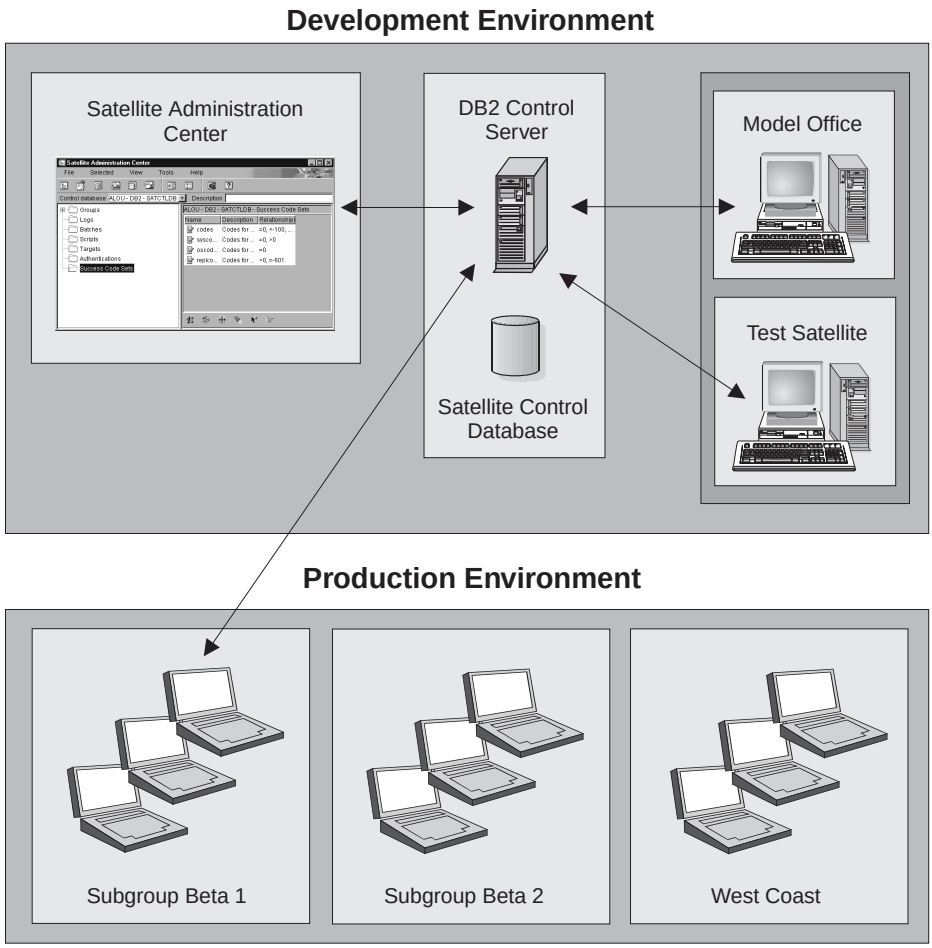


Figure 3. Satellite Environment

In Figure 3 on page 13, all the production satellites in the production environment belong to the same group, but belong to different subgroups. You can specify that a satellite belongs to a specific subgroup when you either create or edit the satellite with the Satellite Administration Center. You can use subgroups to stage the deployment of the first version of the end-user application.

When rolling out the first version of the end-user application, you stage the deployment to control which satellites can synchronize (that is, which satellites can execute the group batches). You also stage the deployment to test whether the database definition and data is appropriate for the end-user application in the production environment. While the group batches may produce correct results on the model office and test satellites of the development environment, the active data of the production environment may indicate that the group batches have to be modified. For example, in Figure 3 on page 13, the subgroup `Beta 1` is the first stage of the deployment, that is, only the `Beta 1` subgroup can synchronize with the DB2 control server. Assume that you receive reports from the `Beta 1` users that the performance of the end-user application is not satisfactory. You can address the application-performance problem, then, when the problem is resolved for the `Beta 1` subgroup, continue by rolling out the `Beta 2` subgroup. Because the `Beta 1` and `Beta 2` subgroups are running the same version of the end-user application, they execute the same group batches of the same application version. This means that the `Beta 2` subgroup is not likely to report the same problem as the `Beta 1` subgroup. To stage the deployment of the first version of the application by subgroups, you enable the satellites, subgroup by subgroup, to execute the group batches.

You can also use subgroups to stage the deployment of the next version of the end-user application. For example, assume that the `Beta 2` and `West Coast` subgroups are running the first version of the end-user application, and that you have tested the second version of the application on a model office or test satellite, then installed the new version of the application on the `Beta 1` subgroup. In this situation, all the subgroups will be enabled to synchronize, and all will be maintaining active data. The difference is that when the `Beta 1` subgroup synchronizes, it executes the group batches associated with the second application version, while `Beta 2` and `West Coast` execute the group batches of the first application version. In this situation, you can use the `Beta 1` subgroup both to determine whether the new version of the end-user application is appropriate for your business requirements in the production environment, and whether the group batches that the `Beta 1` subgroup executes produce satisfactory results.

Chapter 2. Batches

You use batches to ensure that the satellites in a group remain as similar as possible. All the group satellites execute the same group batches that set up and maintain the database definition and data for their version of the end-user application. They also execute the batch steps of these batches in the same order. Because the group satellites execute the same group batches and batch steps, each satellite in the group will be similar.

You can also use batches to fix satellites that either report problems, or require an adjustment. The function of a batch depends on its mode. For more information, see "Batch Modes".

A batch is an ordered set of *batch steps*. A batch step is the combination of a script, the target against which the script is to execute, the authentication credentials required for the script to execute against a DB2 instance or database, and a success code set to indicate whether the script completed successfully or not. The script can be a DB2 command, SQL statement, or an operating system command that you want a satellite to run. For more information, see "Components of a Batch Step" on page 36.

Batch Modes

There are three different modes of batches in the satellite environment:

Group To set up and maintain the database definition and data on satellites, you use group batches. A group batch is associated with a particular application version. In addition, each satellite is associated with an

© Copyright IBM Corp. 1999

application version. When a satellite synchronizes, it downloads and executes the group batches for its particular application version. The satellites always execute the batch steps of the group batches in the same order. Before the group satellites execute the batch steps, they are all similar. When all the satellites have executed all the batch steps of all their group batches, they remain similar.

Because the satellites execute the group batches to maintain their database definition and data, you only have to work with the batches, instead of individually maintaining the hundreds, if not thousands of group satellites.

A group batch will be either a setup, update, or cleanup batch. For more information, see "Group Batches" on page 17. You work with group batches by using the Edit Application Version window in the Satellite Administration Center. You can also create an unassigned batch using the Create Batch window in the Satellite Administration Center, then assign it to a group when you edit the application version. For more information about these windows, refer to the online help for the Satellite Administration Center.

Fix In any environment, problems will occur. A fix batch is one that is created to fix a problem on one or more satellites. This is in contrast to the purpose of group batches, which is to set up and maintain the database definition and data for a specific version of the end-user application. Because of this difference of purpose, fix batches are not assigned to a specific group or to an application version. For more information, see "Fix Batches" on page 43.

Unassigned

An unassigned batch is one that maintains its mode until it is assigned to either:

- An application version, to be executed by a group as a setup, update, or cleanup batch
- A satellite, that will execute the batch as a fix batch.

You can modify an unassigned batch in any fashion, and you can also delete it. If you assign an unassigned batch to an application version, the batch becomes a group batch. If you assign the batch as a fix batch, which you use to make changes to a specific satellite, the unassigned batch becomes a fix batch. In both cases, the change in how the batch is used permanently changes the mode of the batch. That is, the batch cannot be changed back to an unassigned batch.

You work with unassigned batches by using the Create Batch and Edit Batch windows. For more information about these windows, refer to the online help for the Satellite Administration Center.

Group Batches

Group batches are associated with an application version, and you use these batches to set up and maintain the database definition and data on the satellites that are running a particular version of an end-user application. For information about the different types of batches that are associated with an application version, see "Types of Group Batches".

Even though the satellites in the group may execute the batch steps at different times, each satellite of the group will execute the same set of batch steps, and will execute them in the same order. This ensures the consistency of the satellites that belong to the group. It also simplifies the management of large numbers of satellites. You know that the satellites are similar because they have all executed the same set of steps in the same order. If the satellites started out similar before executing the batches of an application version, they will remain similar after they have all executed the batches. For more information, see "Application Versions" on page 19.

Types of Group Batches

Three types of group batches can be executed by satellites when they synchronize: setup, update, and cleanup. You will typically use the setup batch to set up the database definition for the satellite, the update batch to maintain the data on the satellite, and the cleanup batch to perform cleanup activities on the satellite. Using group batches allows you to maintain consistency among the satellites of a group without having to maintain each satellite separately. Only one of each type of group batch can be associated with an application version.

When the satellite synchronizes for the first time, it executes, in order, the batch steps of the setup batch to configure itself, the batch steps of the update batch to populate its tables, and the batch steps of the cleanup batch to perform any cleanup activities. After a satellite synchronizes for the first time, it will execute any new batch steps that are appended to the setup batch to modify its database definition (if required). The satellite then executes all the batch steps of the update batch to maintain its data. Finally, the satellite will execute any batch steps that are appended to the cleanup batch.

Details about the different types of group batches are as follows:

Setup A setup batch is an ordered set of batch steps that is executed first, before any other batches. Each batch step in a setup batch is run only once by a satellite. If you add a new batch step to the setup batch, and the setup batch has already been executed by a given satellite, only the new batch step will be executed by that satellite. You can use the setup batch to set up the database definition on the satellite,

including schemas, tables, indexes, and any other database objects that you require. You can also use the setup batch to set configuration parameter values and to set up data replication.

Update

An update batch is an ordered set of batch steps, and each step is executed every time that the satellite synchronizes. This type of batch is run after a setup batch is run, and before a cleanup batch is run. The batch steps in an update batch are those that can be repeatedly executed. A typical update batch will consist of a data synchronization batch step. If you are using DB2 data replication, this batch step will be an operating system script that runs the Capture and Apply programs by invoking the **asnsat** command.

The steps in an update batch are considered to be idempotent, in the sense that they can be repeatedly executed without changing the current state or database definition of the satellite to a different state with each invocation of the step. For example, a table can be replicated multiple times without resulting in a change in the replication configuration. In contrast, the batch step in the setup batch that originally created the table would be executed only once.

Cleanup

A cleanup batch is an ordered set of batch steps that is executed last, after the update batch. Each batch step in a cleanup batch is run only once by a satellite. If you add a new batch step to the cleanup batch, and the cleanup batch has already been executed by a given satellite, only the new batch step will be executed by that satellite.

A typical cleanup batch is one that contains a batch step that updates the database statistics.

Depending on your requirements, you may not need to create all three types of group batches. For example, you may decide to use a different mechanism to set up the database definition than the setup batch. If a specific type of batch does not exist when a satellite synchronizes, that batch type is bypassed.

You can create group batches from the Edit Application Version window. You can also create unassigned batches from the Create Batch window, then assign them as group batches when you edit the application version. For more information about these windows, refer to the online help for the Satellite Administration Center.

Application Versions

At some point in time, you will need to deploy a new version of the end-user application. Typically, a new version of an end-user application will require a different database definition than the previous version. Consequently, the batches and the associated scripts used to maintain the new database definition will be different.

If you have a large number of satellites in a group, you will probably want to stage the deployment of a new version of the end-user application within that group. That is, you will want to keep most satellites of the group at one version of the end-user application, and use a subset of the group satellites to determine whether the new version of the application meets your business requirements. To stage the deployment, however, you may have to support more than one set of batches for a group of satellites. You will have one set of batches for each version of the end-user application that is used by the group. That is, you will have one set for the original version, and one set for the new version to be deployed. The satellite administration environment supports this requirement through the implementation of application versions.

Each collection of setup, update, and cleanup batches that maintains the database definition and data for the end-user application is associated with an application version. For every version of the end-user application that is used in a group, you require a different application version, and its associated batches. Every group, therefore, must have at least one application version.

When you create an application version on the DB2 control server, you supply a unique identifier for the application version. After the application version is created, you edit it to associate a setup, update and cleanup batch with it.

A satellite will run a particular application to fulfill a specific business requirement. The version of the application on the satellite is identified by the application version that is set on the satellite. You usually set the application version on the satellite using DB2 interfaces during the installation of the end-user application. When you set the value on the satellite for the end-user application, use the same identifier that you used on the DB2 control server. For more information about setting the application version on the satellite, see “Setting the Application Version” on page 71.

When a satellite synchronizes, it uploads its application version to the DB2 control server. The DB2 control server uses this information, in conjunction with the group that the satellite belongs to, to determine which of the group batches the satellite will execute. The DB2 control server only allows the satellite to download and execute the group batches that correspond to the satellite’s application version.

You create an application version on the DB2 control server using the Create Application Version window of the Satellite Administration Center, and the Edit Application Version window to maintain it. For more information about these windows, refer to the online help for the Satellite Administration Center. You can set the application version on the satellite system using a DB2 API or command; you can also set it during the installation of DB2 Satellite Edition on a satellite. For more information about setting the application version on the satellite, see "Setting the Application Version" on page 71.

Levels of an Application Version

Assume that you have a large group of users who will all sell life insurance using the first version of the end-user application. Because the group is large, it may not be practical to perform the deployment at one time. Because of this, you stage the deployment.

To stage the deployment, you enable satellites that have a shared characteristic, such as a common subgroup, to begin executing the group batches. As you deploy more and more satellites, scaling the environment larger and larger, you may find that the database definition and data that seemed appropriate for your application in the earlier stages of the deployment is no longer adequate. For example, you may be receiving reports that performance of the end-user application is becoming worse for many users. It may be that the amount of data that is maintained by each satellite is larger than you originally anticipated. This situation may not necessarily result in satellites returning errors and requiring fix batches, but you need to make some changes to the database definition. In this situation, adding indexes to one or more tables will improve the performance of the end-user application. Rather than creating a new application version, which is appropriate only if you are going to change the version of the application that the end users run, you would create a new level of the *existing* application version. You then modify this new level of the application version to change the database definition.

A new level of an application version is a copy of the setup, update and cleanup batches of the previous level of the application version. You can add one or more additional batch steps that you design to modify the database definition or data. Each level of an application version is associated with a particular number. For example, the first level is level 0, the second level is level 1, and so on. In addition, a level can be a test level for execution by test satellites, a production level for execution for production satellites, or an obsolete level that is not executed by any satellite. For more information, see "States of an Application-Version Level" on page 22

In the new copy of the setup and cleanup batches, you can only append new batch steps. You cannot modify the contents or the order of any existing batch steps. The next time the satellites synchronize, they will only execute the new batch steps of these two batches. All the satellites will execute the new batch steps in the same order, ensuring that the satellites remain consistent after they all execute the new batch steps. If you have new satellites that have never synchronized, they will execute all the batch steps, including the new ones. Because all the satellites in a group that have the same application version run the same setup and cleanup batch steps in the same order, they will have a similar database definition and data after they execute all the batch steps.

The new level of the update batch, however, can be modified in any way that you require. Unlike the situation that applies to setup and cleanup batches, a satellite executes all the batch steps of the update batch each time that it synchronizes. By definition, the update batch is idempotent. That is, a satellite can execute the update batch repeatedly without changing its current state or database definition.

The way that an application interfaces with its data does not change within a version of an application. Once you have set up the underlying database definition to support the end-user application, it is unlikely that you will want to substantially modify it. Rather, you will want extend the database definition in small ways to address intermittent problems such as performance. For this reason, the setup and cleanup batches can only be appended to. Although the database definition sets the structure of the data, with a fixed number of tables and columns, the data content is subject to continual updates. One of the primary uses of the update batch is to replicate data between the satellite and one or more replication sources. You can change the replication operation in any way to ensure that satellites have the data that they require. For this reason, the update batch in a new level is fully modifiable.

In the situation described above, in which the performance of the application is impacted because of an increase in the amount of data managed by each satellite, you would add a new batch step to the setup batch to create indexes. All satellites in the group that have the same application version will execute the new batch step to create the index when they next synchronize. Differences exist, however, in the number of setup batch steps that the satellite will execute when it next synchronizes, depending on when the satellite was rolled out:

- If a satellite was part of an earlier stage of the deployment, and has already executed all of the original group batches, this satellite will first execute the new batch step in the setup batch to create the indexes, then continue by executing the update batch. The new index will result in improved performance for the end-user application.

- If a satellite is new and has not yet synchronized, the first time it synchronizes, the satellite will execute all the batch steps in the setup batch, including the new batch step that creates the indexes. When this satellite completes the synchronization process (that is, it executes all batch steps to the end of the cleanup batch), it will be consistent with older members of the group. This satellite will not report the performance impact on the end-user application.

The levels of an application version allow you to maintain and manage the changes to the database definition and data within an application version. Because the Satellite Administration Center maintains a history of the different levels of the application version, you can track the changes that have occurred over the life time of the application version.

States of an Application-Version Level

Levels simplify the administration of groups because they allow you to test database definition changes on test satellites, without affecting your production satellites. Levels also allow you to maintain the hundreds (or thousands) of production satellites that belong to a group. Each level in a application version has one of the following states: test, production, or obsolete. When you create the first level for an application version, that level, and the batches associated with it, are in the test state. These test batches set up and maintain the database definition and data for the test satellites. You can modify these test batches until you are satisfied with the results that they produce on the test satellites. Then, you promote the test level to production.

When you promote the level to production, the batches associated with this level can be executed by the production satellites. You will probably not want to enable all the satellites of the group to begin executing the production batches at the same time. Instead, consider rolling out subsets of the group satellites to execute the production batches. You can deploy subsets of the satellites according to a characteristic that the subset shares, such as a subgroup. When you enable these satellites, they will download and execute the production batches the first time that they synchronize. Over time, you will enable all the group satellites to execute the production batches.

When the production satellites are executing the batches of the production level, you may find it necessary to modify the database definition or data generated by the production batches to address a problem. For example, assume that your users are calling in, saying that performance of the end-user application is becoming worse over time. You determine that adding indexes to a table will improve the performance of the application. To fix the problem, you create a new test level from the existing production level. You then append a batch step to the setup batch to create the indexes. Because the new level is in the test state, only the test satellites will be able to execute the new

batch step of the test setup batch. When you are satisfied with the results that the test batch generates on the test satellites (that is, the performance of the application is again satisfactory), you promote the test level to the production level. The next time that they synchronize, the production satellites will download the new batch step and create the indexes.

The levels, and the associated states, represent the life cycle of the application version and its associated batches. Additional details about states are as follows:

Test You use a test level to try out database definition changes on the test satellites of your group. The batches of a test level can only be executed by test satellites. Whether you are deploying a new group, or you are testing changes to the database definition, you will want to test the batch steps in a test level to ensure that they produce the results that you want. You can only have one level in the test state. With only one test level, you always know which batches have been modified, and which changes are being tested.

By default, the first level of an application version is created in the test state. This level is level 0. When level 0 is created, all the batches and batch steps that you add to it are in the test state. When level 0 is in the test state, you can modify all of its batches and batch steps. This includes the reordering or deletion of batch steps, and the deletion of the batches.

If a new test level was created from a production level, the setup and cleanup batches can contain a mixture of production and appended test batch steps:

- The production steps of the setup and cleanup batches cannot be modified or reordered. Test batch steps can only be appended after the production batch steps to ensure that the changes that occur on the test satellites result from the test batch steps, and not from changes in the order of the production batch steps.
- The steps of the update batch in the test level are at the test state. They can be modified in any way that you require. For more information, see "Update Batch in a Test Level" on page 28.

When you are satisfied with the results of a test level, you can promote it to production, so that the batches within it can be executed by production satellites. For more information, see "Promoting a Test Level to a Production Level" on page 29.

Production
You use the production level to set up and maintain the database definition and data for the application version that your production satellites run. Because the batches associated with the production level

are fully tested before being promoted, they provide predictable results when the satellites execute them. Using fully tested batches in a production level allows you to scale your satellite environment to any size that you require.

To ensure consistency among the satellites that execute them, production batches cannot be modified or deleted, nor can the batch steps within them. When a test level is promoted to production, all the batches associated with it are set to the production state.

You cannot directly create a production level of an application version. A production level is always a test level that was promoted to production. In this way, you can always use your test satellites to test and tune changes to batches. This helps to isolate testing from the production environment. You must always explicitly promote a test level to production before the production satellites can execute the new or changed batches.

You can only have one production level of an application version. If you have an existing production level and promote a test level to production, the existing production level becomes obsolete. The existing production level is obsoleted because the database definition or the data that it sets up and maintains is no longer adequate to support the end-user application.

If you find that one or more batches of a production level no longer meet all of your requirements, you can create a new test level from it. For more information, see "Creating a Test Level from a Production Level" on page 30.

Obsolete

An obsolete level is no longer adequate to support the end-user application. For this reason, the batches of an obsolete level cannot be executed by any satellite. There are two ways in which a level can become obsolete:

- A new production level supersedes it. This occurs when a test level is promoted to production, and replaces an existing production level. For more information, see "Promoting a Test Level to a Production Level" on page 29.
- You explicitly obsolete it when it is no longer required. For more information, see "Obsoleting a Production Level" on page 30.

There can be many obsolete levels. If you want to be able to track the changes to an application version, you should keep them.

The levels of an application version support a test/production/obsolete development model, which can be used in conjunction with the procedure

that you use to implement your end-user application. For more information, see "Application Version Life Cycle" on page 26.

The following table is an overview of an application version, the available batch types, and the states at which a level can be:

Application Version			
Batches	**Levels**		
At most, one of the following types of batches can be associated with any level of an application version: • Setup • Update • Cleanup.	Test State	Production State	Obsolete State
	The batches of a test level can only be executed by test satellites. When a level is created, the level and the batches that it contains are always in the test state. The batch steps, however, may be a mixture of test and production batch steps. This enables you to validate the changes to the batches before making them available to the production satellites. An application version can contain only one test level.	The batches of the production level can only be executed by production satellites. The batches and batch steps of a production level cannot be modified in any way. This guarantees consistency among your production satellites. An application version can contain only one production level.	The batches of an obsolete level cannot be executed by any satellite. This occurs because obsolete batches are no longer adequate to support the end-user application. You can have multiple obsolete levels of an application version. If you keep the obsolete levels, you can use them to track the changes that have occurred to the database configuration and data that support the end-user application.

States of an Application Version

Like levels, the application version also has a state. You can use the application version details view to determine the state of an application version. This view is available from the Satellite Administration Center. The state is displayed in the **State** column. The state of the application version is set as follows:

- If the application version does not have a production level, the state is not set. That is, no entry exists for the application version in the **State** column.
- If a production level exists for the application version, and none of the production satellites that execute the group batches of this application version are in the failed state, the state of the application version is `Production Normal`.

Note: The application version can also be in the `Production Normal` state if no production satellites are executing the batches associated with this application version.

- If a production level exists for the application version, and one or more of the production satellites that execute the group batches of this application version are in the failed state, the state of the application version is `Satellite Failed`.

Application Version Life Cycle

The discussion that follows describes the test/production/obsolete development model, and how this model applies to the levels of an application version through the life cycle of the application version.

When you create the first level of a new application version, that new level is identified as level 0 and is created in the test state. In the example that follows, there are two batches: `configure`, which sets up the database definition, including data replication; and `replicate`, which maintains the data used by the end-user application:

Application Version				
Level	State	Setup Batch	Update Batch	Cleanup Batch
0	Test	configure	replicate	

You use your test satellites to fully test the batches associated with this level of the application version. When you are satisfied with the results on your test satellites, you promote this level to production. (If you are not satisfied with the results, you can continue modifying the batches until they produce the results that you want. You can also delete level 0, create level 0 again, and create new batches for level 0). When level 0 is promoted, your production satellites can execute the batches of this level. Level 0, which was at the test state, then moves to the production state, as follows:

Application Version				
Level	State	Setup Batch	Update Batch	Cleanup Batch
0	Production	configure	replicate	

Assume that you are in full production, and identify a performance problem. To address this problem, you decide to add an index to one of the replicated tables. You want to test this database definition change on your test satellites before making it available to your production satellites. To do this, start by creating a test level of your existing production level. This new level is in the test state and is a copy of your existing production level. Then, edit level 1 and append a new batch step to the `configure` batch to create the index:

Application Version				
Level	State	Setup Batch	Update Batch	Cleanup Batch
0	Production	configure	replicate	
1	Test	configure (with batch step added to create index)	replicate	

Again, you use your test satellites to fully test the changes associated with the new test level, level 1. Because all members of your test satellites already executed all the batch steps associated with level 0 when it was in the test state, the test satellites will only execute the new batch step in the `configure` batch to create the index when they next synchronize. (They will also execute the entire `replicate` batch.) When you are satisfied with the results of the index creation, promote level 1 to production. Then your production satellites can execute the batches of this level. Level 0 is no longer adequate to support the end-user application. Because only database definition can be in use in the production environment, level 0 is obsoleted:

Application Version				
Level	State	Setup Batch	Update Batch	Cleanup Batch
0	Obsolete	configure	replicate	
1	Production	configure (with batch step added to create index)	replicate	

When they next synchronize, the production satellites that have previously synchronized execute only the create index batch step of the `configure` batch. Satellites that are synchronizing for the first time execute all the batch steps of the `configure` batch, including the create index batch step. To maintain the end-user application data, all satellites run the `replicate` batch after they execute the `configure` batch.

Again, time passes. Although your tables are indexed, you begin to receive reports that the performance of the end-user application is slowly becoming worse. You decide that you need to perform data reorganization on the tables on the satellite to reduce data fragmentation. Again, you need to create a test level from your production level. You decide to add a cleanup batch to contain the batch step that reorganizes the data. In this way, the data is reorganized after it is updated by the `replicate` batch:

Application Version				
Level	State	Setup Batch	Update Batch	Cleanup Batch
0	Obsolete	configure	replicate	
1	Production	configure (with batch step added to create index)	replicate	
2	Test	configure (with batch step added to create index)	replicate	reorganize

Again, you test level 2 with your test satellites. When you are satisfied with results, you promote level 2 to production. Level 1 is automatically obsoleted:

Application Version				
Level	State	Setup Batch	Update Batch	Cleanup Batch
0	Obsolete	configure	replicate	
1	Obsolete	configure (with batch step added to create index)	replicate	
2	Production	configure (with batch step added to create index)	replicate	reorganize

Assuming no new satellites join the group, all satellites bypass executing the `configure` batch, because they have already executed all of its steps. The satellites start by executing the `replicate` batch to maintain the data for the end-user application. When the satellites complete executing the `replicate` batch, they run the `reorganize` batch to reorganize the table data. Each satellite only executes the batch step in the `reorganize` batch once.

Update Batch in a Test Level

When you create a new test level, the update batch steps that were copied from the production level of the update batch are changed to the test state. In this way, you can modify the steps and their order in the batch, as required. Because the update batch is, by definition, idempotent, it should not modify the state or database definition of the satellite. For example:

- There may be times when a change in the setup batch (and in the resulting database definition on the satellites that execute it), require that you change the parameters that the **asnsat** command uses to call the Capture and Apply programs.
- If you use the update batch to back up the database, you may want to change the backup buffer size to obtain better performance.

- If an operation is timing out, you can change a parameter that controls the amount of time that it can take to complete.

In these examples, the changes are implemented by minor changes to existing batch steps in the update batch.

Continuing the example that started in "Application Version Life Cycle" on page 26, assume that you now want to back up one mission-critical table on your production satellites each time that these satellites synchronize. Assume that you want a snapshot of the data of this table before the data is refreshed. You would create a test level from level 2, the current production level. Because you want the backup image to contain a snapshot of the data before it is refreshed, you would add the **backup tablespace** batch step before the batch steps that update the data:

Application Version				
Level	State	Setup Batch	Update Batch	Cleanup Batch
0	Obsolete	configure	replicate	
1	Obsolete	configure (with batch step added to create index)	replicate	
2	Production	configure (with batch step added to create index)	replicate	reorganize
3	Test	configure (with batch step added to create index)	replicate (**backup tablespace** followed by data replication)	reorganize

When you are satisfied with the results, you would promote level 3 to production. Level 2 would be automatically rendered obsolete.

Promoting a Test Level to a Production Level

When you create a level in an application version, it is created in the test state. By definition, only test satellites can execute the batches of a test level. After you are satisfied that the batches in the test level produce the results that you want, you will want the production satellites to execute them. To do this, you must promote the level to production. To promote a test level, use the Edit Application Version notebook. For more information, refer to the help for the Satellite Administration Center.

If you already have a production level and you promote the test level, the existing production level will be moved to the obsolete state. This occurs because the previous production level is no longer adequate to support the

end-user application. Because the test level is a copy of the batches and batch steps of the previous production level, no batch steps are lost from the previous production level. This means that the first time any new production satellites synchronize, they will run the same production batch steps in the same order as satellites that were rolled out earlier. The resulting database definition and data on the new satellites will resemble that of other members of the group. In this way, consistency is maintained across the group.

Creating a Test Level from a Production Level

As described in "Application Version Life Cycle" on page 26, you may find it necessary to refine or extend a setup, update, or cleanup batch that is associated with a production level. To do this, you use the Edit Application Version notebook to add a test level from a production level. When you add a test level, all the batches and batch steps of the production level are copied to the test level. Existing batch steps in the setup and cleanup batches cannot be modified or reordered. Instead, you can only append test batch steps to these batches. You can modify the update batch in any way that you require.

In the case of the setup and cleanup batches, the test satellites will execute only the appended test batch steps. For the update batch, the test satellites will execute all the batch steps, regardless of whether the batch is changed. To modify the batches in a test level, use the Change Level notebook. For more information, refer to the online help for the Satellite Administration Center.

Obsoleting a Production Level

When you obsolete a production level, the batches associated with it can no longer be executed by any satellites. All the batch steps in the batches become obsolete. There are two ways in which you can obsolete a production level:

- By promoting a test level to a production level when there is already an existing production level. The existing production level is no longer adequate to support the end-user application, so it becomes an obsolete level.
- By using the Edit Application Version window. You can select a production level and click on the **Obsolete** push button. You obsolete a level to prevent the execution of its batches. You may, for example, be working on a new test level. Or, you have fully deployed a new version of the end-user application, which has a different application version. In this situation, you do not intend to create any new levels of the previous application version because it is no longer in production.

Relationships Between Batches and Batch Steps

Batches are ordered sets of batch steps. Batch steps, when in group batches, can be test, production, or obsolete batch steps, depending on the state of the associated level. Each batch step contains a script, as well as other information, which the satellites use to set up and maintain the database definition and data for the end-user application that runs on the satellites.

The type of the batch step controls whether that batch step is executed by test or production satellites, and whether the batch step is modifiable. The relationships between the state of a level and the state of batch steps are as follows:

- When you first create an application version, the first level that you add to it is level 0. This level is at the test state, and is empty. It has no setup, update, or cleanup batches. As you create batches for level 0, their steps are in the test state. This occurs because you will want to rigorously test the batches and batch steps with your test satellites to ensure that they set up and maintain the database definition and data correctly.

  Because all the batch steps are in the test state, you can modify or reorder them until you obtain the database definition and data that you require. If required, you can also delete a batch and replace it with a different batch. This characteristic of being completely modifiable only occurs with batches and batch steps associated with level 0 when it is in the test state.
- When you create a new test level from the existing production level, all the batches and batch steps in them are copied to the new test level. Within the new test level, the steps that were copied with the setup and cleanup batches are marked as production batch steps, and cannot be modified or reordered. You can, however, append additional test batch steps to these batches.

  The copied production batch steps in the test level of the setup and cleanup batches cannot be modified, meaning that the changes to the database definition and data on the test satellites can only result from the new batch steps. The original batch steps are locked so that, if unexpected problems occur during the test phase, it will be much easier to identify and repair the problems that can occur. Changes cannot result from changes to the original batch steps, or from an interaction between changes in the original batch steps and the new batch steps.

  You can modify the test batch steps until they provide the results that you want on your test satellites: you can even remove these steps. Because the test of the batch is isolated from the production satellites, normal production is not affected.

  The batch steps that were copied from the update batch of the production level are marked as test batch steps in the new level of the update batch. You can modify these batch steps in any way that you require. For more information, see “Update Batch in a Test Level” on page 28.

- A test level can be promoted to production. When the level is promoted to production, the batch steps of all batches are changed to the production state. Because the batch steps are in production, they are locked and cannot be modified. The batches and batch steps cannot be modified because you do not want untested changes introduced to the production satellites.

  Ordinarily, only production satellites can execute production batch steps, and only test satellites can execute test batch steps. An exception occurs if a new test satellite is introduced to the group. In this situation, the new test satellite has not yet executed any batches. When this satellite first synchronizes, it will execute all the batch steps in the batches of a test level, both production and test steps. In addition, you can configure a test satellite to execute all the batch steps of a test batch, including any production batch steps in it.
- Obsolete batch steps only exist in the batches of an obsolete level. That is, obsolete batch steps cannot exist in either a test or a production level. Obsolete batch steps cannot be executed by any satellite.

How Satellites Execute Batch Steps

As a summary, a production level consists of group batches, the steps of which are in the production state, and are only executed by the production satellites of the group. Given sufficient time, all the satellites of the group will contact the DB2 control server to synchronize, download the batch steps that they have to execute, and execute the associated scripts. Consequently, all the group satellites begin at a consistent state, and end at a consistent state. If you find that you require modifications to the database definition or data, you can create a new level of the application version. The new level will be in test state. You can then modify the setup, update and cleanup batches, as required, to make the modifications.

As previously described, when an update batch is copied to a new level, all the batch steps in the new level of the update batch are set to the test state. The new level of the update batch contains no production batch steps. As a result, you can modify the batch steps of an update batch in any way that you require. You can change batch steps, reorder them, add new ones, and delete existing ones. When a test satellite synchronizes, it runs all the batch steps that are in the test state. By definition, the batch steps of the update batch are idempotent, and satellites always execute all the batch steps of the update batch when they synchronize.

A test satellite executes the batch steps of a setup or cleanup batch differently than how it executes the batch steps of an update batch. When you create a test level of an application version from a production level, all the batch steps in the setup and cleanup batches are copied, but they remain at the production state. You cannot modify these production batch steps in any way,

nor can you reorder or delete them. You can only append new test batch steps. Because the appended batch steps are in the test state, you can modify, reorder, or delete them.

The next time the test satellites synchronize, the DB2 control server will recognize that the test satellites have not executed the new test batch steps. The test satellites will only download and execute those test batch steps that they have not previously executed. A test satellite will not download and execute any of the production batch steps, unless that test satellite has not already executed the full set of production batch steps. This situation can occur with a new test satellite.

Typically you will want to verify the changes that the test batch steps implement on a small number of test satellites before promoting the associated test level to production. If changes are required, you can modify, reorder, and add or delete the test batch steps, and test them again. Because you can test repeatedly, you can refine the test batch steps until you are satisfied with the results that they generate on the test satellites. This process occurs without any impact to the production satellites.

When you are satisfied with the results generated by the test steps, you promote the test level to production. The next time a production satellite synchronizes, it will download and execute the new batch steps in the setup and cleanup batches, and all of the batch steps in the update batch, whether they are changed or not. At this point, one cycle of the test-production cycle of batch step development has been completed.

The interrelationship between the state of a batch step (test or production), and the sequence of its execution within a group batch, is pivotal to the test-production cycle of batch step development. The following example shows a single cycle of the test-production cycle. This example illustrates the development of a setup batch step.

1. Assume that you have the following batch steps in production in the setup batch for level 0 of an application version. These batch steps set up DB2 data replication.

Setup Batch of Production Level 0				
Batch Step	**Script**	**Batch Step State**	**Success Code Set**	**Execution Target**
1	Replication control server setup	Production	Replication success code set	Replication control server
2	Replication source setup	Production	Replication success code set	Replication source
3	Replication target setup	Production	Replication success code set	Local satellite database

Over time, the replicated tables on the satellites have more and more rows added to them. You begin to receive complaints that the end-user application is not performing as well as it did when it was first implemented. You realize that if one of the replicated tables had another index, the application would perform better. But, if you add another index, you should also adjust the heap size on the satellites to account for the new index.

2. The first step is to use the Edit Application Version window to create a test level of the application version from the production level. Then update the test level of the setup batch and append the batch steps that both create the index and change the heap size:

Setup Batch of Test Level 1				
Batch Step	**Script**	**Batch Step State**	**Success Code Set**	**Execution Target**
1	Replication control server setup	Production	Replication success code set	Replication control server
2	Replication source setup	Production	Replication success code set	Replication source
3	Replication target setup	Production	Replication success code set	Local satellite database
4	Create index on replicated table	**Test**	Create index success code set	Local satellite database
5	Heap size alterations	**Test**	Database manager success code set	Local satellite database manager

Two steps (4, 5) are added in sequence, immediately following the three pre-existing production batch steps. Because the new batch level is in the test state, only test satellites can download and execute the new batch steps in it. Production satellites will continue to execute the production-level setup batch, if required. During the testing of these new batch steps, you can modify the test batch steps as necessary to prepare them for production.

Because the test satellites already executed steps 1 through 3 of the setup batch when it was in the test state, they will begin executing the new test-level setup batch at step 4 (satellites only execute the steps of a setup batch once, and in the sequence that they appear in the batch). If you find that the results of the new batch steps are not satisfactory:

- Correct the test batch step that produced the incorrect results. You can have the test satellite re-execute this batch step and any that follow by changing the step at which the test satellite starts executing the test batch. To specify the step from which a satellite begins executing a

batch, use the Batches page of the Edit Satellite notebook, which is available from the Satellite Administration Center.

- In some situations, you must undo the results of the test batch step or steps. Assume that neither step 4 nor step 5 of the setup batch produce the required results. In this situation, you can use a fix batch to undo the results of steps 4 and 5. You then correct the test batch steps, and set the execution start point for the test satellite so that it re-executes the test batch steps.
- You can also use a fix batch to undo not only the effects of the test batch steps, but also of all the batch steps that the test satellite has previously executed. In this situation, after you fix the test batch steps, you set the execution starting point to step 1, and the test satellite executes all the production and test batch steps in sequence.

For more information about fixing a satellite, see "Identifying and Fixing a Failed Satellite" on page 169.

3. When you are satisfied with the results of the test, you promote level 1 to production. The setup batch steps change to the production state. You do this by using the Edit Application Version window, which is available from the Satellite Administration Center. The next time that the production satellites synchronize, they will download and execute the new batch steps, 4 and 5.

Setup Batch of Production Level 1				
Batch Step	**Script**	**Batch Step State**	**Success Code Set**	**Execution Target**
1	Replication control server setup	Production	Replication success code set	Replication control server
2	Replication source setup	Production	Replication success code set	Replication source
3	Replication target setup	Production	Replication success code set	Local satellite database
4	Create index on replicated table	**Production**	Create index success code set	Local satellite database
5	Heap size alterations	**Production**	Database manager success code sets	Local satellite database manager

Scripts and Batch Steps

Batches are made of batch steps. The script is one of the components of a batch step. The other components of a batch step are the execution target, the authentication credential, and the success code set. For more information, see "Components of a Batch Step" on page 36.

You can create scripts either manually, or use the SmartGuides, notebooks, and windows in the Control Center to create the scripts and save them to the Script Center. If the scripts that you want are in the Script Center, you can import them into the group batches of an application version by using the Change Level notebook. If you are creating or editing a fix or an unassigned batch, use the Create Batch or Edit Batch windows. For more information, refer to the online help for the Satellite Administration Center.

Some scripts that are used in group batches must be parameterized. The parameterization is either to take into account differences between satellites that belong to the same group, or to identify values that may be independent of the satellite that is synchronizing. Parameterizing a script means adding an embedded marker into a script that will be replaced with an attribute when the satellite synchronizes. For more information, see "Parameterizing Scripts" on page 40.

Components of a Batch Step

A batch step is the combination of a script, an execution target, authentication credentials, and a success code set.

Script The script component of a batch step can be a DB2 command, an SQL statement, or an operating system command. The script can be a sequence of one or more commands or statements.

For scripts that execute against DB2 instances or DB2 databases, all the commands or statements in the script must execute against a single target. For example, if the script has a target of a DB2 instance called `test`, all of the commands in the script will be executed against `test`. Similarly, if the target is a DB2 database called `payroll`, all of the commands and SQL statements in the script will be executed against `payroll`.

When you specify a DB2 command and its target is either a DB2 instance or a DB2 database, you do not have to specify the `DB2` command prefix, nor to you have to explicitly attach to the instance or connect to the database within the script. The instance attachment or database connection is automatically performed based on the target that is specified for the script. For example, to list the tables of a target DB2 database, the script would contain the following DB2 command:

```
LIST TABLES;
```

Operating system scripts can contain commands that execute against the operating system, as well as DB2 commands that execute against instances or databases.

Notes:

1. An operating system script must be either a batch script, or a command script. That is, the script must have an extension or either .bat or .cmd. If you want to execute a script, such as a Perl script, that does not have an extension of .bat or .cmd, that script must be inside the batch or command script.
2. You can only use .bat files on Windows 95 and Windows 98.

If you want to include DB2 commands that execute against instances or databases in an operating system script, ensure that the script includes the command to explicitly attach to an instance or connect to a database before the DB2 command or SQL statement is executed. This will ensure that the DB2 command or SQL statement in the script is executed against the intended target instead of the default instance or database. Also, you must use the DB2 prefix with the command. For example, to list the tables of a DB2 database, the operating system script would contain the following DB2 commands:

```
DB2 CONNECT TO DATABASE test;
DB2 LIST TABLES;
DB2 CONNECT RESET;
```

Execution target

A script executes locally on a satellite, but, depending on the type of the script, the target can be local or remote. For example, a DB2 command or SQL statement will execute against a target DB2 instance or DB2 database, which can be either local or remote. An operating system command, however, can only be executed by the command processor of the local operating system. Operating system scripts can be executed against anything that is managed by the operating system (such as programs and file systems).

The execution target defines how the script is launched, so the script cannot be associated with more than one target. For example, an instance attach (using authentication credentials) will be initiated before a script with an execution target type of instance is executed.

Note: If the target of a script is a DB2 instance, an ATTACH statement will be automatically issued to the instance before the script is executed. A DETACH statement is automatically issued when execution is complete. Similarly, if the target is a DB2 database, a CONNECT statement will be automatically issued to the database before the script is executed. When the execution is complete, a CONNECT RESET statement is automatically issued. This means that all DB2 instances and DB2 databases that are targets of scripts must be cataloged at the satellite, regardless of whether they are local or remote to the satellite.

Authentication credentials

Authentication credentials are required for DB2 commands and SQL statements to execute against a DB2 instance or DB2 database. In addition, at the start of the synchronization process, the satellite must be able to authenticate against the DB2 control server to obtain the scripts that it is to execute. An authentication credential is the combination of user ID and password that is required to attach to an instance, or to connect to a database. Each script that has an instance or database execution target is associated with a specific authentication credential. Authentication credentials are not required by operating system scripts. For more information, see "Authentication Credentials" on page 45.

Success code sets

When a script is executed, its success or failure is defined by the associated success code set. For DB2 command or SQL statement scripts, each command or statement is executed individually, and its SQLCODE compared with the success code set associated with the batch step. If the statement is successful, the next statement is executed, and so on. Otherwise, execution of the batch is terminated, subsequent steps in the batch are not executed, and an error is reported to the DB2 control server. After the error is reported, synchronization stops, and no additional batches or batch steps are executed. The satellite is disabled from synchronizing and marked as `FAILED` at the DB2 control server. The satellite cannot synchronize again until you fix it.

In the case of operating system scripts, the entire script is executed to completion, then the exit code or return code is compared with the associated success code set. If successful, the next batch step is executed. Otherwise, execution of the batch is terminated, subsequent steps in the batch are not executed, and an error is reported to the DB2 control server. After the error is reported, synchronization stops, and no additional batches or batch steps are executed. The satellite is disabled from synchronizing and marked as `FAILED` at the DB2 control server. The satellite cannot synchronize again until you fix it.

Within an operating system script, you can execute both database manager and database commands. You must include the appropriate ATTACH or CONNECT statement to ensure that the command is executed against the correct DB2 instance or DB2 database. You must also include the DETACH or CONNECT RESET statement after the command. Even if the operating system script contains DB2 commands and SQL statements, the entire script is executed to completion before the exit code or return code is compared to the associated success code set. In this situation, you will probably want to use a scripting language (for example, Perl) to check the SQLCODE

for each DB2 command or SQL statement that you execute against a DB2 instance or a DB2 database. You can use the scripting language to set the exit code or return code for the operating system script.

Notes:

1. An operating system script must be either a batch script, or a command script. That is, the script must have an extension or either .bat or .cmd. If you want to execute a script, such as a Perl script, that does not have an extension of .bat or .cmd, that script must be inside the batch or command script.
2. You can only use .bat files on Windows 95 and Windows 98.

A success code set is one or more comparative operators and numeric values. The comparative operators can be =, >, or <. Numeric values can be any positive or negative integer, or zero (0). All members of the success code set are compared to the SQLCODE that is returned by a DB2 command or an SQL statement, or compared to the exit code or return code that is returned by an operating system command. If the SQLCODE, exit code, or return code is within the defined set, execution of the script is considered to be successful.

The following rules apply for success code sets:

- The set can only have one greater than (>) condition, where the associated code must be greater than or equal to (≥) any less than (<) condition that is specified.

  For example, if you specify (>, 5) and (<, 0), the error codes for that set are 0, 1, 2, 3, 4, 5. You cannot specify (>, 5) and (<, 6), as this will provide all numbers.
- The set can only have one less than (<) condition, where the associated code must be less than or equal to (≤) any greater than (>) condition that is specified.

  For example, if you specify (<, 0) and (>, 5), the error codes for that set are 0, 1, 2, 3, 4, 5. You cannot specify (<, 5) and (>,4), as this will provide all numbers.
- There can be zero or more unique equals (=) conditions, but no duplicate equals conditions.

The following example shows how to set up a success code set for a script that contains multiple SQL statements. Assume that each of these statements drops a table. Each DROP TABLE statement in the script will return an SQLCODE. DROP TABLE statements can return non-zero SQLCODEs that do not represent an error state. You must determine the set of SQLCODEs that do not indicate an error, and include them in the success code set for the script. For example, the

following return codes indicate the successful execution of the script (that is, if any of the following conditions are met, execution of the script continues):

```
SQLCODE = 0, SQLCODE > 0, SQLCODE = -204
```

In this success code set, the `SQLCODE = 0` indicates that the DROP TABLE statement completed successfully. The `SQLCODE > 0` indicates that processing can continue even if a message is issued. The `SQLCODE = -204` indicates that if the table does not exist when the DROP TABLE statement is issued, processing can continue.

You can create the execution target, authentication credentials, and success code set for the batch step by using the Satellite Administration Center. You can create these components of the batch step when you create the batch step, or you can create them in advance, including them when you create the batch step. For more information, refer to online help for the Satellite Administration Center.

Parameterizing Scripts

When you administer satellites at the group level, certain types of scripts that are used in group batches need to be parameterized so that they can be executed by the satellites. You can parameterize a script to identify a value that is independent of the satellite. This is known as a *table parameter*. You can also parameterize a script to identify a value that is unique to a satellite (such as a user ID). This is known as a *contextual parameter*.

Note: For more information about the tables described in this section, and about parameterizing scripts, refer to the following Web site:

```
www.software.ibm.com/data/db2/library/satellite
```

You can add a contextual parameter to a WHERE clause to customize it for the satellites. You can find information that differs from satellite to satellite in the SATELLITES table in the satellite control database. You can also obtain this information from the satellite details view in the Satellite Administration Center. Other tables in the satellite control database that contain information to uniquely identify satellites are SAT_APPVER_PARMS and SAT_HOR_DATASLICES.

When the satellite synchronizes, requesting the group batches from the DB2 control server, the DB2 control server checks whether any script is parameterized before allowing the satellite to download it. If any script is parameterized, the DB2 control server will substitute the appropriate table or contextual parameter for the parameterized markers before the satellite downloads the script.

You specify whether a script is parameterized by using the Change Batch Steps notebook and the Create and Edit Script windows, all of which are available from the Satellite Administration Center. There are two types of parameter markers that you can use:

Table Parameters The table parameter is the general mechanism for specifying scalar values. You can use a table parameter to specify a single column value for a single row of a table in the satellite control database.

Syntax

{{*tablename*:*colname*:*predicates*}}

The syntax translates to `SELECT` *`colname`* `FROM` *`tablename`* `WHERE` *`predicates`*. The parameters are as follows:

tablename
: Is the fully qualified two-part name of a table in the satellite control database. For example, `schema.tablename`.

colname
: Is the name of the column in *tablename* that contains the value to be substituted for the parameter marker.

  Note: *colname* cannot resolve to a column with a data type of CLOB, BLOB, GRAPHIC(1), GRAPHIC(n), VARGRAPHIC(n), or LONG VARGRAPHIC.

predicates
: Is the WHERE clause predicates that are used to identify the row that contains the required column value. The predicates should identify a single value. If not, one value is returned, but the value is not deterministic.

Contextual Parameters Contextual parameters refer to the values that apply specifically to the satellite that will execute a script. These values are all of the attributes of the satellite recorded in the SATELLITES table for that satellite. Other tables in the satellite control database that contain information to uniquely identify satellites are SAT_APPVER_PARMS and SAT_HOR_DATASLICES. The contextual parameter is more restrictive than a table parameter. The *predicates* parameter is implicitly defined to select the specific row that applies to the satellite.

Syntax

{{SATELLITES:*colname*}}

Where:

colname
Is one of the columns in the SATELLITES table

Note: *colname* cannot resolve to a column with a data type of CLOB, BLOB, GRAPHIC(1), GRAPHIC(n), VARGRAPHIC(n), or LONG VARGRAPHIC.

Examples The examples that follow show how to use parameters:

- To obtain the name of the group to which a satellite belongs:

```
{{SATELLITES:GROUP}}
```

- To obtain the last name of the user of a satellite:

```
{{SATELLITES:LAST_NAME}}
```

- To obtain the WHERE clause predicates that are applied to the FINANCE.EXPENSES table in the GROUPA subscription set for the satellite in GROUPA:

```
{{satadmin.sat_hor_dataslices:predicates:satellite='{{SATELLITES:ID}}'
AND version='{{SATELLITES:APP_VERSION}}'
AND set_name='GROUPA'
AND group='{{SATELLITES:GROUP}}'
AND source_schema='finance'
AND whos_on_first='F'
AND source_table='expenses'
AND target_schema='finance'
AND source_view_qual=0
AND target_table='expenses'}}
```

Script Storage on the Satellite During a Synchronization Session

When a satellite synchronizes, it obtains the scripts that it is to execute from the DB2 control server, then stores the scripts in a location that is specific to whether the script is for a setup, an update, or a cleanup batch. The execution results are also stored. When the synchronization session ends normally, the contents of these directories are deleted. If the synchronization session is interrupted (either the user stops the session or an abend occurs), the contents of these directories are not deleted.

Directory	Description
instance_path\satellite	The base directory for stored scripts **Note:** You should not modify the contents of this directory; otherwise, synchronization may not be able to occur.
instance_path\satellite\setup *instance_path*\satellite\setup\results	The directory where the scripts for a setup batch are stored. Results are stored in the \results directory.
instance_path\satellite\update *instance_path*\satellite\update\results	The directory where the scripts for an update batch are stored. Results are stored in the \results directory.
instance_path\satellite\cleanup *instance_path*\satellite\cleanup\results	The directory where the scripts for a cleanup batch are stored. Results are stored in the \results directory.

Fix Batches

You can use fix batches for a variety of situations:

- To fix problems on test satellites when a group batch in a test level does not produce the intended results.
- To fix problems that occur on production satellites.
- To fix problems that are caused by a fix batch. In this situation, you can either modify the original fix batch, or create a different fix batch that both undoes the results of the original fix batch and applies a different fix.

  Note: If a specific fix batch is assigned to more than one satellite, you should not modify the batch until you are sure that all the satellites that are assigned to execute it have done so. If the fix is unsatisfactory, it will be unsatisfactory across all the satellites that execute it, and you will have a consistent starting point from which to apply a different fix. For information about how to identify which satellites are using a particular fix batch, refer to the online help that is available from the Satellite Administration Center.
- To query the results of a previous fix batch.
- To make a change to a satellite that is not required by any other satellite.
- To query the current state of a satellite.

Because the batch steps are not locked, you can repeatedly modify the steps in a fix batch until the problem on the satellite is fixed to your satisfaction. When you are satisfied with the results of a fix batch on a satellite, you

promote that satellite back to executing its group batches. For information about promoting a satellite, refer to the online help provided with the Satellite Administration Center.

You work with fix batches by using the Create Batch and Edit Batch windows. For more information about these windows, refer to the online help for the Satellite Administration Center. For more information about fixing satellites, see "Identifying and Fixing a Failed Satellite" on page 169.

Chapter 3. Authentication in the Satellite Environment

In the satellite environment, the administration solution is based on the synchronization between satellites and the DB2 control server. For synchronization to occur, you must set up a variety of authentication credentials, both to enable the satellite to download the scripts, and to enable the satellite to execute the scripts.

Authentication Credentials

An authentication credential is the combination of a user ID and a password. Almost every activity in the satellite environment requires authentication, from the first connection to the satellite control database for a test synchronization, to the execution of a script. Authentication credentials reside both at the DB2 control server (in the satellite control database), and at every satellite in the environment. The master copy is at the DB2 control server. Every satellite maintains a shadow copy of the authentication credentials. Because all passwords in the satellite environment are encrypted, the authentication information on the satellites cannot be updated independently of the synchronization process with the DB2 control server. The encryption prevents unauthorized access to a password for a user ID, and allows you to maintain tight control over the security of the environment.

Authentication Credentials Stored at the DB2 Control Server

The DB2 control server maintains the master copy of all the authentication credentials that are required in the satellite environment. Because all authentication credentials are at the DB2 control server, you can manage all your authentication credentials from a central location.

© Copyright IBM Corp. 1999

When you create an authentication credential, you give it a name and provide a user ID and password. The password is encrypted when it is stored in the satellite control database. When you create a target, you provide the name of the authentication credential. The user ID and password of this authentication credential is used to authenticate the user with the target.

You must create an authentication credential for every DB2 instance or DB2 database that will be the target of script execution. Because all the satellites in a group execute the same scripts against the same targets, you only need to create one set of authentication credentials for the group. Because operating system scripts run locally on the satellite under the authority of the system administrator, you do not need to create an authentication credential for them.

Because you will know which targets the satellites in a group will need to authenticate with before you set up a model office or perform a deployment, you may find it more convenient to create the authentication credentials and targets before creating any batches. You can set up and maintain the authentication credentials by using the Create Authentication and Edit Authentication windows. You can set up and maintain targets by using the Create Target and Edit Target windows. These windows are available from the Satellite Administration Center.

Note: The online help for the Create Authentication and Edit Authentication windows states that the maximum length for a password is 256 characters. This length is not correct. The maximum supported length for a password is 31 characters.

For information about how authentication credentials are used to authenticate the execution of a script in a batch step, see "Components of a Batch Step" on page 36. For information on how to update the passwords at the DB2 control server, see "Managing Password Changes" on page 48.

Authentication Credentials on Satellites

All user IDs and passwords that are used in the satellite environment for synchronization are stored in the *instance_directory*\security\satadmin.aut file on each satellite. The entries in this file mirror the authentication credentials at the DB2 control server. The satellite synchronization process maintains the satadmin.aut file by downloading the encrypted passwords from the satellite control database, and storing the encrypted passwords in the file. The use of encryption prevents direct access to all passwords that are used in the satellite environment.

Only satellites that are both recorded in the satellite control database and enabled to execute group batches can use the synchronization process to access and download the authentication credentials.

During a synchronization session, two types of security checks occur. First, the satellite must authenticate with the DB2 control server to download the batch steps that it will execute. Second, when each script is executed, if its target is an instance or a database, authentication occurs before the script can be executed. If the target of the script is the local operating system, the script executes locally on the satellite under the authority of the system administrator. For additional information, see "Authentication with Target Servers for Script Execution".

Creation and Maintenance of Authentication Credentials on a Satellite

The satadmin.aut file is created either during installation, or at the first test synchronization. During the installation process, you can supply a user ID and password that will be used to connect to the satellite control database on the DB2 control server. If you supply this information during installation, the authentication file is created and the user ID and password stored in it. If you do not provide the user ID and password during installation, you will be prompted for them the first time you run the **db2sync -t** command to perform a synchronization test. At this time the file will be created and the authentication credentials stored in it. For more information about the **db2sync** command, see "Appendix C. db2sync - Start DB2 Synchronizer" on page 243.

The first time that the satellite synchronizes, it downloads the authentication credentials for all the targets against which it must authenticate, and stores this information in its authentication file. All the passwords in the file will be encrypted. On subsequent synchronizations, changes to the authentication credentials are also downloaded so that the authentication credentials on the satellite are always current. For additional information, see "Managing Password Changes" on page 48.

Authentication with Target Servers for Script Execution

When a satellite executes a script against a target that is a DB2 instance or a DB2 database, the user ID associated with the authentication credentials must be authenticated before the script can be executed. Authentication credentials, the combination of a user ID and password, are required to authenticate on each attach to a DB2 instance, or connect to a DB2 database. When the satellite executes the batch step, its authentication file is accessed to retrieve the user ID and password for the target, which is then used on the attach or connect to authenticate the satellite.

Authentication for DB2 Data Replication

In addition to scripts, if you are using DB2 data replication, the Apply program also requires authentication credentials for each database that it must connect to when it replicates data. On a satellite, the Apply program checks for authentication credentials as follows:

1. The Apply program first checks the satadmin.aut file on the satellite. The Apply program will check that the authentication credentials are available for the replication control server, all replication source servers that it needs to replicate with, and the local DB2 database replica.
2. If the Apply program cannot find the information in the satadmin.aut file, it checks the replication password file for the authentication credentials. For information about creating a replication password file, refer to the *Replication Guide and Reference*.

   Note: Because the replication password file does not encrypt passwords, it is recommended that you define all authentication credentials required for DB2 data replication by using the Satellite Administration Center.

Managing Password Changes

Because of standard security procedures, situations will occur in which the password must be changed for one or more of the target DB2 servers against which satellites authenticate. When this occurs, however, the satellites may experience authentication errors when they attempt to attach or connect to the target DB2 server. To avoid this problem, the satellite maintains both the current and the previous password for all of the DB2 databases or servers against which that satellite must authenticate. The fact that the satellite maintains both passwords enables you to make a transition from the existing password on a DB2 server to the new password.

Managing Password Changes for Access to the DB2 Control Server

To change the password associated with the user ID that one or more groups use to synchronize with the DB2 control server:

1. Identify the authentication credentials that are being used by the group. You can use the Edit Group window to see the name of the authentication credentials that are being used by the group. This window is available from the Satellite Administration Center.
2. Edit the named authentication credentials to change the password associated with the user ID. To perform this task, use the Edit Authentication window.

When you change the password used to access the DB2 control server, the **Password changed** column of the satellite details view in the Satellite Administration Center changes to `Yes` for all the satellites in the group. The `Yes` value indicates that the password required to access the DB2 control server is changed.

When a satellite next synchronizes, it will recognize that the password is changed, and download the changed password. As soon as the satellite obtains the new password, the **Password changed** column of the satellite details view in the Satellite Administration Center changes to `No` to indicate that the satellite has the new password.

The next time that the satellite synchronizes, it will use the new password to connect to the DB2 control server. Because the password is not yet changed at the DB2 control server, authentication will fail. When the authentication fails, the satellite will attempt to connect again, this time using the old password. This time, the authentication will succeed.

3. When all the satellites have updated their authentication file with the new password (that is, the **Password changed** column of the satellite details view in the Satellite Administration Center is `No` for all the satellites of the group), change the password that is required to access the DB2 control server. You can use the ATTACH command with the CHANGE PASSWORD parameter to change the password for the DB2 control server, or you can use the operating system security manager.

Managing Password Changes at Target DB2 Servers

To change the password at a target DB2 server:

1. Edit the target to determine which authentication credential it is using. To perform this task, use the Edit Target window in the Satellite Administration Center.
2. Edit the authentication credentials to change the password that the satellites require to be authenticated at the target DB2 server to the new password that you intend to use on the target server. To perform this task, use the Edit Authentication window in the Satellite Administration Center.
3. Change the password on the target DB2 servers using the facilities provided by the operating system.

When the satellite next synchronizes, any new passwords are downloaded and stored, in encrypted format, in the satadmin.aut file on the satellite. The satellite will use the new password when it attempts to connect to the target DB2 server. If the password is not yet changed at the target server, authentication will fail. The satellite automatically recovers from this error by attempting to connect or attach again, this time using the old password. Authentication should succeed.

An authentication error can occur if a synchronization session is stopped or terminates before the satellite has executed all the scripts for that session. When an interrupted synchronization session is restarted, the satellite will execute the remaining scripts for the session before connecting to the DB2 control server to report the results of the session. The error occurs if any scripts that remain to be executed access a target DB2 server whose password changed between the time that the synchronization session was stopped and restarted. Because the satellite does not download updated passwords until it connects to the DB2 control server, the satellite will not have the correct password for the DB2 target that has undergone a password change. You can recover from this situation by refreshing the authentication credentials on the satellite. For more information, see "Re-Creating or Updating the satadmin.aut File" on page 176.

Chapter 4. Cataloging Instances and Databases

To be able to use the satellite administration solution and replication, you must catalog the instances and databases on each system, including the satellites, in the configuration. The sections that follow describe what DB2 objects must be cataloged on each system, and suggest techniques that you can use to catalog the required entries.

Note: LDAP (Lightweight Directory Access Protocol) is not supported on DB2 Satellite Edition.

Cataloging Instances and Databases in the Control Center Instance

The Control Center and the Satellite Administration Center require access to the instance of the DB2 control server, and the satellite control database. You can run the Control Center on the same system as the DB2 control server and the satellite control database. If you do, the DB2 control server is automatically installed when you install it. The satellite control database is automatically cataloged when you create it.

Note: The Control Center instance, here, and in the sections that follow, refers to the instance where the DB2 JDBC server is running. The Control Center uses the DB2 JDBC server instance.

If the Control Center is running on a system that is not the system where the DB2 control server is running, you must catalog the DB2 control server and the satellite control database in the Control Center instance. You can use the

© Copyright IBM Corp. 1999

Control Center to perform this task. See "Cataloging Systems, Instances, and Databases on the Control Center" on page 53 for an overview of how to catalog.

When you have cataloged the DB2 control server and the satellite control database, you can use the Satellite Administration Center to create groups and satellites, and to define batches to be executed by the satellites.

A batch step executes against an execution target, which can be a DB2 instance, a DB2 database, or the operating system on the satellite. You must create a named target for every DB2 instance or DB2 database against which a batch step will execute. You do not need to create a named target for the batch steps that execute against an operating system.

Before you can create a named target for a DB2 instance or DB2 database using the Satellite Administration Center, you must first catalog the following targets in the node and database directories of the Control Center instance.

- The instances and databases of the model office, as these are the targets of the administrative scripts.

  An additional advantage to cataloging the model office instances and databases in the Control Center instance is that you can then use the Control Center both to examine the model office, and to generate scripts to manipulate the model office. To create the scripts, use the Show SQL and Show Command facilities of the Control Center windows and notebooks. You can also use the Control Center to manage the model office, for example, to back up the databases on it. You must install the Inbound Remote Administration component of DB2 Satellite Edition on the model office before you can use the Control Center to administer it. By default, the Inbound Remote Administration component is installed when you perform a typical install. The component is optional on a custom install. See "Inbound Remote Administration" on page 189 for more information.

  Not all of the Control Center tools can be used to administer a model office. See "Appendix D. Administering DB2 Satellite Edition from the Control Center" on page 245 for more information.
- If you intend to use DB2 data replication, you must catalog the following:
  - All replication source and target databases. When the source and target databases are cataloged, you can use the Control Center Replication Source and Replication Subscription windows to define replication sources and subscriptions. You should catalog both your test and production source and target databases so that you can use the Control Center to manage the transition from test to production. In addition, when the production databases are cataloged, you can use the Generalize Replication Subscription window, which is available from the Satellite

Administration Center. For more information about setting up replication for the satellite environment, see "Chapter 7. DB2 Data Replication" on page 95.

– All replication control servers, and the databases on them that contain the control tables. A replication control server is a DB2 server that houses the database that contains the replication control tables. The Apply program uses the replication control tables to manage the apply process during data replication. You should catalog the replication control servers and the databases that contain the replication control tables for both the test and the production replication environments.

You can use the Control Center to perform the cataloging.

If you want your help desk to use the Control Center and the Satellite Administration Center to perform problem determination and diagnosis on satellites, you should catalog the following in the Control Center instance that the help desk will use:

- The DB2 control server and the satellite control database
- The replication control servers.

Cataloging Systems, Instances, and Databases on the Control Center

Use the Control Center to catalog the required systems, instances and databases. First, you catalog the DB2 control server and the satellite control database. You then catalog the model office. The sections that follow describe how to use the Control Center to perform these tasks.

Note: When you catalog systems and instances, you *must* specify TCP/IP as the protocol to use for communications. When you export the connection information from the Control Center, then import this information to the model office, all of its system and instance node directory entries will specify TCP/IP, which is the only protocol supported by DB2 Satellite Edition.

Cataloging the DB2 Control Server and the Satellite Control Database

To catalog the DB2 control server and the satellite control database in the Control Center object tree perform the following steps.

Note: Examine the object tree of the Control Center under the system where the DB2 control server is running. If the DB2CTLSV instance and the SATCTLDB database already appear, you do not have to perform these steps. This situation occurs when the Control Center's JDBC server is using the DB2 control server instance.

1. Open the Control Center.
2. Add the system where the DB2 control server is running:
   a. Right click on the Systems folder and select **Add** from the pop-up menu.

      The Add System window opens.
   b. You can use one of two methods to add the system:
      - You can click **Refresh** to add the system that contains the DB2 control server. Ensure that you set the protocol to TCP/IP. For more information about DB2 discovery, refer to the *Administration Guide*.
      - You can enter the required information:
        1) Type the name of the system into the **System name** field.
        2) Type the name of the instance in the **Remote instance** field.

           On Windows NT, the name is `DB2DAS00`.
        3) Select the correct value for the **Operating system** field.
        4) For the **Protocol** field, select `TCP/IP`.
        5) Type the TCP/IP host name for the system on which the DB2 control server resides in the **Host name** field.
        6) Use the **Service name** default of `523`.
        7) Optionally, add a comment to the **Comment** field.
   c. Click **OK**.
3. Add the DB2 control server instance:
   a. Expand the tree under the newly added system by clicking on the **+** beside it.
   b. Right click on the Instances folder and select **Add** from the pop-up menu.
   c. Complete the Add Instance panel. You can use one of two methods to add the instance:
      - You can use DB2 discovery to retrieve the required information. This method is recommended.

        Click **Refresh** to retrieve a list of instances. Select the DB2 control server instance. On Windows NT, the name of the instance is `DB2CTLSV`. Ensure that the protocol is set to `TCP/IP`.
      - You can enter the required information:
        1) In the **Remote instance** field, type the name of the remote instance.
        2) In the **Instance name** field, type a unique instance name.
        3) For the **Protocol** field, select `TCP/IP`.
        4) Type the TCP/IP host name for the system on which the DB2 control server resides in the **Host name** field.

5) Type the port number of the DB2 control server instance in the **Service name** field.

   On Windows NT, if you did not create the DB2 instance when you installed the DB2 control server, the port number is 50000 (unless you changed the default value during the installation). If you created the DB2 instance, the value is likely 50002.

d. Click **OK**.

4. Add the satellite control database, SATCTLDB:
   a. Expand the tree under the DB2CTLSV instance by clicking on the **+** beside it.
   b. Right click on the Databases folder and select **Add** from the pop-up menu.
   c. Complete the Add Database panel. You can use one of two methods to add a database:
      - Click **Refresh** to retrieve a list of databases. Then select the SATCTLDB database.
      - Type `SATCTLDB` in the **Database name** field.
   d. Click **OK**.

The Control Center can also be used to catalog replication sources and targets, and the database that will act as the replication control server by following the steps outlined above. If these data replication objects reside on a system that does not support DB2 discovery, you cannot use the **Refresh** button to discover the object. In this situation, you must enter all the data manually.

Cataloging the Model Office

To catalog a model office and its instances and databases in the Control Center, perform the following steps:

1. Open the Control Center.
2. Add the system on which the model office runs:
   a. Right click on the Systems folder and select **Add** from the pop-up menu.
   b. Complete the Add System panel.

      Note: You cannot use the **Refresh** and **Retrieve** buttons to add a satellite to the Control Center. DB2 discovery is not supported on DB2 Satellite Edition.

      1) Type the name of the system on which the model office runs in the **System name** field.
      2) For the **Remote instance** field, type the name of the remote instance. In most situations, the name is `DB2`.

3) Select the value of `Windows NT` for the **Operating system** field, even if Windows 95 or Windows 98 is running on the system.
4) Select `TCP/IP` for the **Protocol** field.
5) For the **Host name** field, type the host name for the system.
6) Use the **Service name** default of `523`.
7) Optionally, add a comment to the **Comment** field.

c. Click **OK**.

3. Next add the DB2 instance.
   a. Expand the object tree under the newly added system by clicking on the + beside it. You will receive the following message:

      ```
      The DB2 Administration server is not active
      ```

      Ignore this message. No DB2 Administration Server exists on a DB2 Satellite Edition system.
   b. Right click on the Instances folder and select **Add** from the pop-up menu.
   c. Complete the Add Instance window.

      Note: You cannot use the **Refresh** button to retrieve a list of instances. DB2 discovery is not supported on DB2 Satellite Edition.

      1) In the **Remote instance** field, type the name of the remote instance. In most situations, the name is `DB2`.
      2) In the **Instance name** field, type a unique name.

         Note: The name you type must be cataloged on the model office as an alias for the local instance name. Most likely, the name of the local instance is `DB2`.
      3) Select `TCP/IP` for the **Protocol** field.
      4) Type the TCP/IP host name for the model office in the **Host name** field.
      5) Type the port number for the instance in the **Service name** field. Unless you changed the default value when you installed DB2 on the model office, the value is `50000`.

   d. Click **OK**.
4. Add the databases on the model office:
   a. Expand the object tree under the newly added instance by clicking on the + beside it.
   b. Right click on the Databases folder and select **Add** from the pop-up menu.

c. Complete the Add Database panel.

   Note: You cannot use the **Refresh** button to retrieve a list of databases. DB2 discovery is not supported on DB2 Satellite Edition.

   1) For the **Database name** field, type the name of the database.
   2) Optionally, type an alias for the database in the **Alias** field.

d. Click **OK**.

e. Repeat the procedure to add the other databases on the model office.

Cataloging Instances and Databases on the Model Office

The sections that follow describe how to catalog instances and databases, both remote and local, on the model office. Ensure that you catalog the instances and databases on each model office that you use.

Cataloging Remote Instances and Databases

When a satellite synchronizes, it connects to the satellite control database at the DB2 control server. To connect (and consequently to synchronize), the satellite must have catalog entries for the DB2 control server instance and the satellite control database in its node and database directories.

When the satellite downloads scripts from the DB2 control server to execute, the DB2 instance or database that is the execution target must already be cataloged on the satellite. For replication this would include:

- All replication sources and targets.
- All replication control servers. The database alias names used on the model office must match the database alias names that are defined in the instance of the Control Center.

You can use two methods to catalog the instances and databases that are remote to the model office:

- You can issue DB2 **catalog** commands through the CLP to catalog the remote instances (nodes) and databases.

  If you use the **catalog** command, you must ensure that the instance and database alias names that you specify for the command match those that are recorded in the instance of the Control Center. These names are the same names that you use when you create execution targets in the Satellite Administration Center. One simple way to ensure that the names recorded at the model office match those recorded at the Control Center is to write a script that issues the **catalog** commands. You then run the script on both the Control Center instance and the model office instance. As an alternative, you can also use the **Show Command** button on the different Control

Center windows that you use to catalog instances and databases to display the **catalog** command, then save the command to a file which you can later execute on the model office.

- You can export a client profile to a file from the instance of the Control Center. You then import this file on the model office. The file contains the information that is required to create the node and database directory entries on the model office.

  Importing the client profile from the Control Center provides one major advantage: you ensure that the instance and database alias names are identical to those in the instance of the Control Center. You use these names when you use the Satellite Administration Center to create execution targets. See "Using a Client Profile to Perform Cataloging on the Model Office" for more information.

You may have to catalog other nodes and databases on the model office, for example, if your user will be using the satellite as a client to access another DB2 system. You can create entries in the node and database directories on the model office by using the **catalog tcpip node** and the **catalog database** commands. For more information, refer to the *Command Reference*.

Using a Client Profile to Perform Cataloging on the Model Office

To use a client profile to perform cataloging on the model office, perform the following steps.

Note: The procedure that follows applies to the Windows NT environment.

1. On the system where you are running the Control Center:
   a. Start the Client Configuration Assistant. Click **Start** and select **Programs -> DB2 for Windows -> Client Configuration Assistant**. You use the Client Configuration Assistant to create a client profile file. This file contains information to catalog the required target instances and databases on the model office.
   b. Click **Export**.
   c. On the Select Export Option window, select **Customize** and click **OK**.
   d. On the Customize Export window, select the databases that you require as targets on the model office. Do not select the database that is on the model office. Deselect **Client settings** and **CLI/ODBC common parameters**.

      Note: If you are running the Control Center on the same system as the DB2 control server and the satellite control database, do not select the SATCTLDB database. If you do, the satellite control database will be set up on the model office as a local database.
   e. Click **OK**.

f. On the Export Client Profile window, type a name for the file and specify the directory to store the file in.

g. Click **OK**.

   The client profile file is saved under the name and location that you specified. You now make the file available to the model office by importing it to the model office.

2. On the model office:

   a. Open a command window.

   b. Issue the **db2cfimp** *file_name* command to import the client profile file.

For more information about the **db2cfimp** command, refer to the *Command Reference*.

Note: If you are running the Control Center on the same system as the DB2 control server and the satellite control database, and you followed the note associated with step 1d on page 58, you did not select the satellite control database, SATCTLDB. As a result, you will have to catalog the DB2 control server instance, DB2CTLSV, and the SATCTLDB database on the model office using the **catalog tcpip node** and **catalog database** commands. For more information about these commands, refer to the *Command Reference*.

Cataloging Local Instances

If any local instance on the model office will be used as an execution target, you must also catalog this instance on the model office using the same name that you used when you cataloged the instance on the Control Center. Specifically, you must map the instance name used on the Control Center to the actual local instance name using the **catalog local node** command.

For example, assume that the name that was given to the local instance when it was created is `DB2` (On Windows NT, DB2 is the default instance that is created by the installation program). Also assume that you called the `DB2` instance on the model office `SALESINS` when you cataloged it on the Control Center. Because `SALESINS` is the execution target of scripts that you want to run against the local instance called `DB2`, you must run the following command on the model office:

```
CATALOG LOCAL NODE SALESINS INSTANCE DB2
```

This command makes `SALESINS` an alias for `DB2` on the model office.

Completing the Setup of the Model Office

As you begin to use the model office during your development phase, you may set up ODBC/CLI values, and modify the database manager configuration values and DB2 registry variables to represent a production environment. You can transfer these configuration values, in addition to the cataloged entries for instances and databases, to your test and production satellite systems using the **db2cfexp** and **db2cfimp** commands. For more information, see the sections that follow and "Chapter 10. Performing a Mass Deployment" on page 145.

Cataloging Instances and Databases on Test Satellites

A test satellite is a copy of the model office that you use for testing scripts and replication. You can use test satellites during the development process, or, after you deploy the production satellites, to test scripts that will change the database definition.

To have test satellites execute batches, all the execution targets and the replication sources, targets and replication control server must be cataloged on the test satellite. These instances and databases must be cataloged on the test satellite under the same instance and database alias names that are used on the model office. The simplest way to achieve this is to export a client profile file from the model office, then import the file on each test satellite.

In addition, if you are using the test satellites to verify the application that you intend to deploy on the production satellites, consider setting up ODBC/CLI for each database, and configuring the database manager configuration and DB2 registry information so that you can test the application in an environment that is very similar to the production environment. If you set up this information on the model office, you can use the **db2cfexp** and **db2cfimp** commands to copy this information from the model office to the test satellite.

Using a Client Profile to Set up a Test Satellite

You can automate the configuration of the test satellite during the development process by exporting a client profile from the model office and importing it on the test satellite. Use the **db2cfexp** command with the TEMPLATE option to generate a client profile from the model office. When DB2 Satellite Edition is installed on the test satellite, import the client profile by using the **db2cfimp** command.

For more information about the **db2cfexp** and **db2cfimp** commands, refer to the *Command Reference*.

When you begin to deploy your production satellites, you can use the same method that you use to deploy production satellites both to create the required catalog entries on the test satellites, and to configure the test satellites to have the same configuration as the production satellites. You can then use the test satellites to test proposed changes to the production satellites. For more information, see "Chapter 10. Performing a Mass Deployment" on page 145.

Cataloging Instances and Databases on Production Satellites

Assuming that you maintain the model office throughout the development cycle so that it truly represents your production satellites, you can use the node and database directory entries, ODBC settings, database manager configuration values and DB2 registry values of the model office to set up your production satellites.

To automate the set up of the production satellites, export a client profile from the model office and import it to the production satellite. Use the **db2cfexp** command with the TEMPLATE option to generate a client profile from the model office. When DB2 Satellite Edition is installed on the satellite, import the client profile to the production satellite by using the **db2cfimp** command.

To automate the process, you can export the client profile from the model office when you run the response file generator utility, **db2rspgn**, to generate the response file that you will use to install your production satellites. The install program will import the client profile to set up the production satellites. For more information, see "Chapter 10. Performing a Mass Deployment" on page 145.

Chapter 5. Setting up and Testing Your Satellite Environment

To set up and test the satellite environment, you perform a variety of different tasks. Some of these tasks you do at the DB2 control server, while others occur at the satellite. In the overview described here, the procedure is to:

1. Prepare for a test synchronization:
   a. Set up the DB2 control server.
   b. Set up a test satellite.
   c. Run the test synchronization.
2. Setup and test the group batches:
   a. Create an application version.
   b. Create a level.
   c. Create the batches.
   d. Have the test satellites synchronize to test the group batches.
   e. Promote the group batches to production.

© Copyright IBM Corp. 1999

Preparing for a Synchronization Test

You run a test synchronization to ensure that a satellite can connect to, and is recognized by, the DB2 control server. Before a test synchronization can be done, a number of different pieces of information must be created on both the DB2 control server, and on the satellite from which the test synchronization will be executed. The following sections describe how to:

1. Prepare the DB2 control server, including:
   a. Installing the DB2 control server.
   b. Creating the user ID and password, and the authentication credentials that the satellites require to synchronize.
   c. Granting access to the satellite control database for this user ID.
   d. Creating a group.
   e. Creating satellites for the group.
2. Prepare the test satellite, including:
   a. Installing DB2 on the satellite.
   b. Setting the satellite ID.
   c. Setting the application version.
   d. Setting the user ID and password required for synchronization.
3. Run the synchronization test.

Setting up the DB2 Control Server

You use a variety of windows and notebooks in the Satellite Administration Center to prepare the DB2 control server for satellite synchronization. The description that follows provides an overview of the recommended steps and the order in which they should be performed. For detailed information about any of the windows or notebooks referenced in this section, refer to the online help provided with the Satellite Administration Center.

Before you can use the Satellite Administration Center, ensure that a satellite control database is cataloged on the Control Center. If the DB2 control server is on the same machine as the Control Center, the DB2 control server instance, and the satellite control database, SATCTLDB, are automatically cataloged. If the DB2 control server is running on a different machine than the Control Center, you must catalog the system, the DB2 control server and the satellite control database on the Control Center. For information about performing this task, see “Chapter 4. Cataloging Instances and Databases” on page 51.

The Satellite Administration Center is enabled when the Control Center detects that any of the cataloged instances contain a satellite control database. You can open the Satellite Administration Center from the pop-up menu associated with the instance that contains the satellite control database, or

from the pop-up menu associated with the satellite control database. You can also open it from the Control Center tool bar. For information about using the Satellite Administration Center, refer to the online help provided with it.

Note: You should only administer the satellite environment from the Satellite Administration Center.

Installing the DB2 Control Server

See "Chapter 13. DB2 Control Server Planning and Installation" on page 181 for information about installing and setting up the DB2 control server.

Creating the User ID for Satellite Synchronization

In the satellite environment, you administer groups of satellites, and not individual satellites. This basic principle also applies to the authentication that is required to access the DB2 control server and the satellite control database. All the satellites of the group use a group-level user ID and password when they connect to the satellite control database for synchronization.

To set up the DB2 control server to enable satellite synchronization:

1. Decide which user ID and password that you want the group of satellites to use when they connect to the satellite control database for synchronization.
2. Define this user ID and password to the operating system security manager using the tools provided by the operating system on which the DB2 control server resides. The user ID and password must be set up at the operating system level because authentication is performed by the operating system.
3. Create an authentication credential using the Create Authentication window of the Satellite Administration Center. Ensure that the authentication credential has the same user ID and password that you just specified at the operating system. When you create the authentication credential, give it a name that is meaningful. In one of the steps that follows, you will create a group of satellites. When you create the group, you will specify that this authentication name be used by the group.

Notes:

1. You can use a group ID instead of a user ID.
2. User IDs and passwords are case sensitive.

You specify this same user ID and password either when you install DB2 on a satellite that will belong to the group, or when you run a synchronization test for the first time from the satellite. The satellite requires the user ID and password so that it can authenticate with the DB2 control server. Each satellite

that belongs to the group will use the same user ID and password to authenticate with the DB2 control server for the purpose of synchronization. For more information, see "Setting up the Satellite Authentication File" on page 71.

Granting Access to the Satellite Control Database

The next task is to grant privileges to the tables in the satellite control database. You will need to grant table privileges both to the user ID that the group satellites use to connect to the DB2 control server, and to administrative and help-desk staff.

You use standard DB2 authentication and authorization mechanisms to control access to the satellite control database. For more information about how to grant privileges to tables, refer to the description of the GRANT statement in the *SQL Reference*. You can also use the Privileges notebook in the Control Center to grant privileges on the tables. For information about using this notebook, refer to the online help that is available from it. For general information about DB2 security, refer to the description of how to control database access in the *Administration Guide*.

The sections that follow describe why you need to grant table privileges, and provide guidance on the types of privileges that you should consider granting to the different types of users. In addition, information is provided about the type of privileges that you should grant on the stored procedures that are part of the DB2 control server.

Granting Access to the Group Satellites: Each group of satellites uses one authentication credential to both connect to the DB2 control server, and to access the tables of the satellite control database. You specify the name of this authentication credential when you create the group. See "Creating a Group" on page 70 for more information.

The authentication credential, as described in "Chapter 3. Authentication in the Satellite Environment" on page 45, is the combination of a user ID and a password. You can use the Edit Group window to see the name of the authentication credential that the group satellites use to access the DB2 control server. You can use the Edit Authentication window to obtain the user ID that is associated with the authentication credential.

You must grant appropriate table privileges to the user ID of the group-level authentication credential so that the group satellites can access the tables in the satellite control database when they synchronize. "Tables and Authorizations" on page 68 lists the recommended authorities that you should grant to this user ID for the satellite control tables. To simplify the administration of the privileges, you can create a group at the operating

system level to contain the user ID associated with the authentication credential for each group. If you use an operating system group, you can grant and manage database and table privileges for the operating system group, instead of having to manage the privileges for each user ID for each group of satellites. To set up and manage the privileges:

1. Create a group in the operating system security manager for the user IDs.
2. Grant the required authorities to the group.
3. Add additional user IDs to this group as you add new groups to the satellite environment, as required.

If you change the user ID that the group satellites use to access the DB2 control server and the satellite control tables, add it to the operating system group to ensure that this user ID has the appropriate table privileges.

If you change the password that is associated with the user ID that the group satellites use to synchronize, the user ID's table privileges are not affected.

Granting Access to the Administrative and Help Desk Staff: In your organization, you may have staff who require administrative authority to the satellite control database, SATCTLDB. If you have a help desk, likely the help-desk staff also require access to the satellite control database. You should consider using two operating system groups to control access to the satellite control database, one for administrative staff, and one for help-desk staff. If you use groups, you can grant and manage database and table privileges for two groups, instead of having to manage the privileges of each user separately. To set up and manage the privileges for the administrative staff:

1. Create a group in the operating system security manager for the satellite administrators.
2. Grant the required authorities to the group (DBADM authority, for example, on the SATCTLDB database).
3. Add additional administrative staff to this group, as required.

You can create a group for your help-desk staff following a similar procedure. "Tables and Authorizations" on page 68 lists the minimum recommended authorities that you should grant to the group to which the help-desk staff belong. Depending on your requirements, the recommendations in "Tables and Authorizations" on page 68 may be too restrictive, as it only enables the help-desk staff to view information when they use the Satellite Administration Center. For example, if you want help-desk staff to be able to use the Satellite Administration Center to fix satellites, the group for the help-desk staff requires the following table privileges:

- To apply an existing fix batch to a satellite, grant the UPDATE privilege on the SATELLITES table.

- To enable or disable a satellite, grant the UPDATE privilege on the SATELLITES table.
- To modify the execution starting point for a satellite, grant the UPDATE privilege on the SATELLITES table.

  Note: For information about these tables, refer to the following Web site:

  `www.software.ibm.com/data/db2/library/satellite`

You may want the help-desk staff to use the Satellite Administration Center to perform other tasks as well. For example:

- To reset passwords, grant the UPDATE privilege on the TARGET_AUTH table, and the EXEC privilege on the satencrypt user-defined function.
- To delete log records, grant the DELETE privilege on the LOG table.
- To customize replication parameters for a satellite, grant the INSERT, UPDATE, and DELETE privileges on the following tables:
  - GROUP_SUBSCR_SETS
  - GRP_HOR_DATASLICES
  - SAT_HOR_DATASLICES
  - GROUP_APPVER_PARMS
  - SAT_APPVER_PARMS.
- To change success code sets, grant the INSERT, UPDATE, and DELETE privileges on the following tables:
  - SUCCESS_CODES
  - SUCCESS_RELATIONS.

Tables and Authorizations: In the following table, recommendations for privileges granted to the tables in the satellite control database are provided. Before granting these privileges, you must grant the CONNECT privilege on the SATCTLDB database to:

- The group that contains the user ID associated with the authentication credential for each group
- The group to which your administrative staff belong
- The group to which your help-desk staff belong.

Note: A blank cell in the following table indicates that no privilege is required for the table in the satellite control database.

Table Name	Privilege for Group Authentication Credential	Privilege for Help-Desk Staff Group
TARGET_AUTH	SELECT	SELECT
TARGETS	SELECT	SELECT
GROUPS		SELECT
SCRIPTS	SELECT	SELECT
SUCCESS_CODES	SELECT	SELECT
SUCCESS_RELATIONS	SELECT	SELECT
BATCHES	SELECT	SELECT
BATCH_STEPS	SELECT	SELECT
APP_VERSIONS		SELECT
GROUP_BATCHES	SELECT	SELECT
SATELLITES	SELECT, UPDATE	SELECT
GROUP_APPVER_PARMS	SELECT	SELECT
SAT_APPVER_PARMS	SELECT	SELECT
GROUP_SUBSCR_SETS		SELECT
GRP_HOR_DATASLICES	SELECT	SELECT
SAT_HOR_DATASLICES	SELECT	SELECT
LOG	INSERT	SELECT

Note: When you create the authentication credential that the satellites use to synchronize in test mode, you can set up the user ID to have minimal access to the tables in the satellite control database. For example, you can grant this user ID the SELECT privilege on only the SATELLITES, TARGET_AUTH, and TARGETS tables. Consider doing this if, for example, you want to use a different authentication credential for the test synchronization than for a synchronization session in which the satellites execute group batches.

If the satellites are enabled to execute the group batches when they synchronize in test mode, they will download the group-level user ID and password that they require to authenticate with their DB2 control server for synchronization.

For information about the table privileges required for data replication, refer to the description of how to set up the replication environment in the *Replication Guide and Reference*.

Granting Privileges on Stored Procedures and Bind Files: The following table lists the bind files and stored procedures, and the type of privileges that you should grant on them to your groups.

Bind File or Stored Procedure	Privilege for Administrative Group	Privilege for Group Authentication Credential	Privilege for Help-Desk Staff Group
DB2SATCS	REBIND/EXECUTE	EXECUTE	
DB2PROM	EXECUTE		

Creating a Group

The next step is to create a new group. The group will consist of related satellites that share characteristics such as the database definition and the application that runs on the satellites. When you create the group, you must specify the authentication credential that you created in "Creating the User ID for Satellite Synchronization" on page 65. The satellites in the group will use this authentication credential when they synchronize.

To perform this task, use the Create Group window. You specify the name of the authentication credential required for synchronization in the **Authentication name** field.

Note: You cannot create a group without specifying an authentication credential for it.

Creating Test Satellites

The next step is to create the test satellites that will belong to the group. You will use these test satellites to perform the synchronization test. Later, you will use them to test the group batches.

Use the Create Satellite notebook to create the satellites. When you create a satellite, you must specify an identifier for it. This identifier must match the value that exists on the satellite. For more information, see "Setting the DB2SATELLITEID Registry Variable" on page 72.

You can use the subgroup attribute of satellites to further categorize the satellites. The subgroup can identify different regions, or any other grouping that you require. You specify the subgroup attribute in the **Subgroup** field of the Create Satellite notebook. You can also use the subgroup attribute to control the staging of a deployment. For example, you could enable one subgroup to synchronize before enabling the other subgroups.

When satellites are created, they are production satellites by default, and they are not enabled to execute batches.

The next task is to make the satellites test satellites. To perform this task:

1. In the Satellite Administration Center, open the satellite details view.
2. In the contents pane, select the satellites that you want to use as test satellites, and click the right mouse button.
3. Select **Set as test satellite** from the pop-up menu.

   The satellites are now test satellites, but still cannot execute the group batches. You will enable these satellites to execute the group batches when you test the group batches.

Installing and Preparing a Satellite for Synchronization

Before a satellite can connect to the DB2 control server, three different sets of information must be set up on the satellite: the authorization file, the application version, and the satellite ID. You can set up this information either when you install the satellite, or later.

Setting up the Satellite Authentication File

The satellite authentication file, satadmin.aut, is created either when you install DB2 Satellite Edition with the control server synchronization component, or when the first test synchronization is run from the satellite. During the installation process, you can supply a user ID and password that will be used to connect to the satellite control database on the DB2 control server when the satellite synchronizes. If you supply this information during installation, the satadmin.aut file is created and the user ID and password stored in it. If you do not provide the user ID and password during installation, you will be prompted for them the first time you run the **db2sync -t** command to perform a synchronization test. At this time the file will be created and the authentication credentials stored in the file.

For information about the **db2sync** command, see "Appendix C. db2sync - Start DB2 Synchronizer" on page 243.

Setting the Application Version

When a satellite synchronizes, it passes its application version to the DB2 control server. The DB2 control server uses this information, in conjunction with the group that the satellite belongs to, to determine which of the group's application-version batches the satellite will execute. The application version allows you to support multiple versions of the end-user application within a group, which enables you to perform a staged deployment of a new version of the application. For more information, see "Installing a New Version of an Application" on page 156 and "Application Versions" on page 19.

You can specify the application version of the satellite when you install DB2 Satellite Edition with the control server synchronization component. You can also specify the application version when you install the end-user application on the satellite. If you specify the application version when you install the end-user application, the install program must call either the db2SetSyncSession API or the **db2sync -s** command to record the application version. For more information about the db2SetSyncSession API, refer to the *Administrative API Reference*. Also see "Setting the Application Version on the Satellite" on page 115. For more information about the **db2sync** command, see "Appendix C. db2sync - Start DB2 Synchronizer" on page 243.

Note: The application version is case sensitive.

You may find it more convenient to supply the application version when you install your initial test satellites. In a large production environment, you should specify the application version when you install the end-user application.

Setting the DB2SATELLITEID Registry Variable

When a satellite synchronizes, it passes the its ID to the DB2 control server. The satellite ID is determined as follows:

1. If a value is specified for the DB2SATELLITEID registry variable, the satellite ID is this value.
2. Otherwise, the logon ID is used as the satellite ID.

The DB2 control server verifies that the same value is recorded in the satellite control database. You create the satellite ID entry in the satellite control database when you create the satellite using the Create Satellite notebook in the Satellite Administration Center. For more information, see "Creating Test Satellites" on page 70. The satellite ID must be recorded on both the satellite and in the satellite control database before synchronization can occur. Having the satellite ID on both the satellite and in the satellite control database ensures that only known satellites can synchronize.

If you do not want to use the logon ID as the satellite ID, you can set the value of the DB2SATELLITEID registry variable when you install DB2 Satellite Edition with the control server synchronization component. If you do not specify this registry variable during installation, you must specify it before synchronization can occur.

To set this registry variable, use the **db2set** command on the satellite as follows:

```
db2set DB2SATELLITEID='identifier' -i instance_name
```

Where:

identifier
Is the character string that uniquely identifies the satellite. The string can be a maximum of 20 characters (including blanks).

Notes:

1. The value that you specify on the satellite for *identifier* must match the satellite ID that you specified when you created the satellite with the Create Satellite notebook. To find the ID of a specific satellite, you can use the satellite details view in the Satellite Administration Center. For more information, refer to the online help provided with the Satellite Administration Center.
2. The satellite ID is case sensitive.

instance_name
Is the name of the instance on the satellite. Unless you have created another instance on the satellite, this value is `db2`.

The **db2set** command can be issued from the operating system command line, or from an application program, for example, an installation program for the end-user application. For more information about the **db2set** command, refer to the *Command Reference*.

Installing the Satellite

For information about installing DB2 Satellite Edition, see "Chapter 14. Planning for DB2 Satellite Edition Installation" on page 187 and "Chapter 15. Installing DB2 Satellite Edition" on page 195.

Verifying the Setup

When you are finished setting up the DB2 control server and your test satellites, you should ensure that the configuration information in the satellite control database and on the satellite is both correct, and consistent.

Configuration Information	Satellite Control Database	Satellite
Satellite ID	Create the ID using the Create Satellite notebook. You can view the ID using the satellite details view.	The satellite ID can be either the user's logon ID, or the value of the DB2SATELLITEID registry variable on the satellite.

Configuration Information	Satellite Control Database	Satellite
Authentication credential required for synchronization	Create the authentication credential using the Create Authentication window, and associate the credential with the group when you create the group.	The file satadmin.aut must exist on the satellite. Create the file using the **db2sync -t** command. Alternatively, you can specify the user ID and password when you install DB2 Satellite Edition with the control server synchronization component.
Application version	Create using Create Application Version window.	Set the application version on the satellite using either the **db2sync -s** command, or the db2SetSyncSession API. You can also specify the application version when you install DB2 Satellite Edition with the control server synchronization component on the satellite.

In addition, the versions of DB2 on both the DB2 control server and the satellite must be compatible. See "Incompatible Versions of DB2 on the Satellite and the DB2 Control Server" on page 165 for details.

Running a Test Synchronization

You have now set up and configured the DB2 control server, and one or more test satellites. You are at the point at which you can run a test synchronization. To test the synchronization capability of the satellite:

1. Open a command prompt window and enter the following command:

   ```
   db2sync -t
   ```

   The DB2 Synchronizer opens in test mode.
2. Click **Test**.

   The Connect to Control Database window opens if you did not specify the user ID and password that the satellite requires to synchronize when you installed DB2 Satellite Edition.
3. If the Connect to Control database window opens, type the user ID and password, and click **OK**.

   The Catalog Control Database window opens if the satellite control database is not yet cataloged at the satellite.

4. If the Catalog Control Database window opens, specify the required information about the instance where the satellite control database resides. This information includes the name of the instance, the host name of the machine where the instance resides, and the service name or port number of the instance. You can click the **Refresh** button to have DB2 discovery search for the systems that have DB2 installed on them.
5. When DB2 discovery completes, select the system that has the DB2 control server from the drop-down list.
6. Click **Retrieve** to retrieve the list of DB2 instances on the system.
7. When the retrieve operation completes, select the DB2 control server instance from the drop-down list.

   When you select the DB2 control server instance, the **Host name** and **Service name** fields are automatically updated with the values for that instance.
8. Click **OK**.

If the list of systems returned by DB2 discovery does not include the system on which the DB2 control server resides, you must enter the host name for the system in the **Host name** field, and the port number in the **Service name** field. Then click **OK**.

If all the information specified at the DB2 control server and at the satellite matches, and the satellite control database is correctly cataloged, the test synchronization will succeed. The next major task is to create the test batches that set up the database definition and data for the first version of the end-user application, then to test these batches.

Creating and Testing Group Batches

When the test synchronization is successful, meaning that the test satellite can connect to the DB2 control server and validate its configuration, you are ready to set up the test batches that create and maintain both the database definition and the data for the first version of the end-user application. When these batches are set up, the test satellites will be able to execute them when they synchronize. The examples that follow are limited to scripts that execute locally on the satellite, and not against remote targets. For information about enabling script execution against targets that are remote to a satellite, see "Chapter 4. Cataloging Instances and Databases" on page 51. The steps that you perform are as follows:

1. Create authentication credentials.
2. Create execution targets.

3. Create an application version:
   a. Create your first test level, level 0, for the application version.
   b. Edit level 0 to add batches.
4. Catalog the remote targets on the satellite. For more information, see "Cataloging Instances and Databases on Test Satellites" on page 60.
5. Synchronize the test satellites.
6. Promote level 0 to production.

Creating Authentication Credentials

You create authentication credentials for the DB2 instances and DB2 databases that a satellite will either attach or connect to when it synchronizes. You can create a separate authentication credential for every potential target, or you can use one authentication credential for all targets, whichever is appropriate. For each batch step that the satellite executes when it synchronizes, the satellite will authenticate with the execution target using the user ID and password of the associated authentication credential.

For example, assume that you have a DB2 instance that is called `DB2`, and a DB2 database that is called `SALES`. Assume that there is a user ID with SYSADM authority that is called `SALESADM` with a password of `SALESPW`. This is the user ID that you will associate with scripts that require SYSADM authority to execute. You could create the authentication credential associated with this user ID and password under the name `SALESCRED`. This is the authentication credential that you would specify when you create a target.

DB2 uses the operating system security manager to authenticate clients. Because of this, the user ID and password must exist in the security manager of the operating system of the target. In the example here, the `SALESADM` user ID must exist in the security manager with a password of `SALESPW`. To enable this user ID to be able to execute the DB2 commands and SQL statements contained in the scripts, you must give this user ID SYSADM authority. Add this user to a group, then update the value of the database manager configuration parameter *sysadm_group* to be the group name. For more information about this configuration parameter, refer to the *Administration Guide*.

To create authentication credentials, use the Create Authentication window. Enter the following values (the values follow the example above):

1. Specify `SALESCRED` for the **Name** field.
2. Specify `SALESADM` for the **User ID** field.
3. Specify `SALESPW` for the **Password** and **Confirm password** fields.
4. Click **OK**.

Creating Execution Targets

All scripts execute against a target, which can be a DB2 instance, a DB2 database, or the operating system on the satellite. You must create a named target for every DB2 instance or DB2 database against which a script will execute. You do not need to create a named target for scripts that execute against an operating system.

Before you can create a named target for a DB2 instance or DB2 database, you must first catalog them in Control Center. For more information, see "Cataloging Systems, Instances, and Databases on the Control Center" on page 53.

Note: You must catalog the instance and the database using the same name and alias under which the instance and database is cataloged on the satellites.

When you have cataloged all of the instances and databases against which scripts will execute, you use the Create Target window in the Satellite Administration Center to create the named targets for these instances and databases.

For example, you would have to create targets for the `DB2` instance and the `SALES` database. To create the target for the `DB2` instance:

1. In the **Alias** field, select the target. You must select the same alias under which the instance is cataloged on the satellites. This is the same name under which the instance is displayed in the Control Center.

   The **Type** field is automatically filled based on the type of the alias. In this situation, the type is `Instance`.
2. In the **Authentication name** field, select `SALESCRED`. This is the name of the authentication credential that was created in the previous example. This authentication name is associated with a user ID that has SYSADM authority.
3. Click **Test** and a test attachment is attempted to verify the user ID and password against the target for validation. Type the password for the user ID when you are prompted for it.

   Note: The user ID and password must already exist at the target; otherwise, the test will fail. For more information, see "Creating Authentication Credentials" on page 76.

Assuming that you want to use the same authentication credentials, follow the same procedure for the `SALES` database. Again, remember to select `SALES` in the **Alias** field for the target. The database alias must match the alias under which the database is cataloged at the satellites.

Creating an Application Version

The next step is to create an application version for the group. An application version is associated with a set of setup, update, and cleanup batches that set up and maintain a specific database definition and data. The database definition and data applies to a specific version of the end-user application that runs on the satellites of a group. For more information about application versions, see "Application Versions" on page 19.

To perform this task, use the Create Application Version window:

1. In the **Version** field, specify the application version.

   When the Create Application Version window opens, this field is filled with the default application version (that is, if you have not already created an application version). If the default application version does not match the value that you set on the satellites, change it to match the value on the satellites.
2. In the **Description** field, specify a description for the application version.
3. Click **OK**.

Creating Level 0 of the Application Version

The next step is to add the first level of batches, level 0, of the application version. You initially use level 0 to contain the set of group batches that set up and maintain the database definition and data for the end-user application. This level is created in the test state.

To perform this task, use the Edit Application Version notebook. The first time that you open this window, it is empty, as the application version does not yet have any levels associated with it:

1. Click **Add** to add level 0 to the application version.
2. When you see the message `Are you sure that you want to add a new level`, click **YES**.

When level 0 is first created, it is empty. No batches are associated with it. The next step is to create the batches for the application version.

Editing Level 0 to Create or Modify Group Batches

The next step is to add group batches to level 0. When level 0 is in the test state, all the batches and batch steps you add are fully modifiable. For information about levels of application versions, see "Levels of an Application Version" on page 20.

To perform this task:

1. Select level 0 in the Edit Application Version window, and click **Change**.
2. On each of the Setup, Update, and Cleanup pages of the Change Level notebook:
   a. Specify the name of the batch.
   b. Add or import the scripts that you want to be executed to the **Batch steps** field:
      - If you already have created scripts in the Satellite Administration Center, click **Add** to add them to the batch.
      - If you have saved scripts in the Script Center, click **Import** to import them from the Script Center to the Batch.

You do not have to use all three types of batches. For example, in the initial phase of developing a database definition, you may not consider it necessary to have a cleanup batch. If a particular type of batch is not present when a satellite synchronizes, that batch type will be bypassed.

Changing Batch Steps

The next step is to combine the scripts for the batches with the authentication credentials, execution targets, and success code sets that are required for the scripts to be executed. The combination of a script with the information that it requires for the satellite to be able to execute it is known as a batch step. For more information, see "Components of a Batch Step" on page 36.

To perform this task, use the Change Batch Step notebook.

Repeat the following steps for every script of every batch:

1. On the Edit Application Version window, select the new level, level 0, and click **Change**.

   The Change Level notebook opens.
2. On the Change Level notebook, select the batch page for which you want change batch steps.
3. Select the batch step with which you want to associate an execution target, a success code set, and an authentication credential, and click **Change.**

   The Change Batch Step notebook opens.
4. On the Script page:
   a. Optional. Change the name of the script in the **Script** field.
   b. Specify a description of the script in the **Description** field.
   c. Optional. Change the type of target against which the script will execute from the **Type** radio buttons.

d. Optional. If you want to import additional scripts from the Script Center, click **Import**.
e. Optional. If you want, you can edit the contents of the script in the **Script contents** box.
f. Optional. If the script is parameterized, select the **Parameterized** check box. For more information about parameterization, see "Parameterizing Scripts" on page 40.
g. Optional. To use a character other than the semicolon (;) as the statement termination character for a script that executes against a DB2 instance or a DB2 database, specify the character in the **Statement termination character** field.

5. For a script that executes against a DB2 instance or a DB2 database, on the Execution Target page:
   a. Select the alias of the DB2 instance or DB2 database for the **Target alias** field. Use the **...** button to display the list of targets.

      Note: The alias that you select must be the alias which is cataloged at the satellite.

   b. Select the named set of authentication credentials for **Authentication** field. The credentials are required by the satellite for its authentication with the target. Use the **...** button to display the list of authentications.
6. On the Success Codes page:
   a. Specify the name of the success code set in the **Success codes set name** field.

      Note: If you have already created a success code set, you can click **...** to list it.

   b. Specify a description for the success code set in the **Description** field.
   c. Specify the success code set for the script in the **Specify codes** box.
   d. Click **Add** to add a success code relation pair to the **Specify codes** box.
7. Click **OK** to exit from the Edit Batch Step notebook.
8. Click **OK** to exit from the Change Level notebook.

Testing Group Batches

When you have the group batches set up for the first level of the application version, it is probable that you will want to test them on a small number of satellites before promoting the batches to production. To perform this task, you first assign a subset of the satellites in the group to be test satellites. When a satellite is created, it is, by default, a production satellite. You must specify which satellites you want to execute the test batches. You performed this task in "Creating Test Satellites" on page 70.

Note: It is recommended that you maintain the same set of test satellites throughout the life cycle of the end-user application.

Enabling the Test Satellites to Execute the Test-Level Batches

The next task is to enable the test satellites to execute the test-level group batches. In "Creating Test Satellites" on page 70, you created the test satellites. When satellites are created, they are, by default, unable to execute batches. Before the satellites can synchronize, you must enable them. To perform this task:

1. In the Satellite Administration Center, open the satellite details view for the group.
2. In the contents pane, select the test satellites and click the right mouse button.
3. Select **Enable** from the pop-up menu. Click **OK** when you are prompted for confirmation.

   The test satellites are now enabled to execute test batches.

When the test satellites synchronize, they will download and execute the batch steps that set up the database definition and data for this version of the end-user application.

Synchronizing Test Satellites to Execute the Test-Level Batches

At this point, you are ready to have your test satellites synchronize so that they can download and execute the test batches. The test satellites can synchronize using one of two methods:

- If you performed the install interactively, the DB2 menu is available from the **Start** menu. Click **Start** and select **Programs -> DB2 for Windows NT -> DB2 Synchronizer**.
- Open a command prompt window and enter the following command:

  ```
  db2sync
  ```

When the DB2 Synchronizer application opens, do the following:

1. Click **Start**.

   The status indicator bar shows the progress of the synchronization.

   If a problem occurs, you can click **Log** to open the Log Information window. This window shows the detailed log messages that are written during the synchronization session.
2. When the synchronization session is complete, click **Close**.

Note: The first time that a satellite synchronizes to execute its group batches, its application version is recorded in the satellite control database. That is, the satellite details view in the Satellite Administration Center does

not display the satellite's application version until it synchronizes. You cannot use the Satellite Administration Center to specify the application version of the satellite.

Checking the Results of the Synchronization Session

To determine the results of the synchronization session, perform the following steps:

1. Open the satellite details view, and locate the test satellites that have executed the batches. In particular, check the values under the **State** column. If the entry in this column is `Results stored`, the satellite has executed the batches and reported back the results of the execution.

   If the entry in this column is `Satellite failure`, an error occurred when the satellite was synchronizing. The satellite does not execute any batch steps after encountering the error.
2. Regardless of whether or not an error occurs, you should check the logs of the test satellites that synchronized. To check the logs:
   a. From the satellite details view, select a test satellite and click the right mouse button.
   b. Select **Show logs** from the pop-up menu.

      The Log Details window opens for the satellite. If the satellite reported a failure, the last log entry for the satellite indicates the nature of the failure, and supplies additional information about the failure for diagnosis.

If a failure occurs, it is easy to identify which batch step you will have to modify, if necessary. If, however, no satellites report a failure, you will have to examine the database definition and the data on the test satellites to determine whether the batches produced the results that you intended. If a failure occurred or the results of the synchronization were not satisfactory, you can use one or more of the following actions to correct the situation using the Satellite Administration Center.

- Correct the test batch step that produced the incorrect results. The next time the satellite synchronizes, it will begin executing the associated batch from this batch step. This method is appropriate if the batch step that failed did not leave the satellite in an inconsistent state, or did not produce an inappropriate change to the database definition. As an example, assume that you have an SQL statement that fails because of incorrect syntax. This error will not result in changes to the database definition on the satellite that must be undone. If, however, the SQL statement did run and resulted in a table being populated with incorrect data, this method would not be appropriate. Instead, you would use one of the following methods.

  Note: If the test satellite failed, you must enable it.

- In some situations, you must undo the results of the test batch step or steps. In this situation, you can use a fix batch to undo the results of specific batch steps:
  1. Create a fix batch to undo the results of one or more batch steps.
  2. Select the satellite from the satellite details view and click the right mouse button.
  3. Select **Fix** from the pop-up menu.
  4. On the Fix Satellite window, specify the fix batch that you want the satellite to execute.
  5. Correct the batch step or steps that failed.
  6. When the satellite has successfully executed the fix batch (and the batch steps that caused the problem have been corrected), promote the satellite back to executing its group batches. For information on promoting a satellite, refer to the online help for the Satellite Administration Center. When you promote the satellite, set the batch step at which the test satellite will resume executing the batch to be the first step of the batch that failed. In this way, the test satellite will re-execute the scripts when it synchronizes
- You can also use a fix batch to undo not only the effects of the test batch steps in error, but also of all the batch steps that the test satellite has already executed. In this situation, after you fix the test batch steps, you set the execution starting point to step 1, and the test satellites will execute all the test batch steps in sequence the next time they synchronize. For information about setting the execution starting point for a satellite, see "Setting the Execution Starting Point for a Satellite" on page 84.

After fixing all the problems that occurred, you are ready to have the test satellites execute the test batches again. The next time that the satellites synchronize, they will begin executing each batch at the batch step that you specified when you set the execution starting point, or the batch step they stopped at when they last synchronized.

After the test satellites synchronize, check the satellite details view and the logs of the test satellites again. If an error occurred, or individual batch steps did not produce the required results, perform the necessary corrective action, and repeat the synchronization. Repeat this process until you are satisfied with the results that you obtain. When the database definition and the data that are produced by the batches meet your requirements, you are ready to promote level 0 and its batches to production.

Promoting a Group Batch to Production

When you are satisfied with the results of the test-level group batches, the next step is to promote level 0 and its batches from test to production. Promoting level 0 makes the group batches of level 0 available to the production satellites, that is, when they are created and enabled to execute batches. When the production satellites synchronize, they can download and execute these batches to set up the production environment. (The test satellites will no longer be able to execute the batches of level 0.)

To perform this task, use the Edit Application Version notebook:

1. Select level 0, which is currently in the test state.
2. Click **Promote**. Click **OK** when you are prompted for confirmation.

   The state of level 0 changes from `Test` to `Production`.
3. Click **OK**.

Performing a Deployment

The final step is to perform the deployment. To deploy, enable the production satellites to begin executing the production-level group batches for their application version. If you want, you can stage the deployment by enabling production satellites according to a shared characteristic. For example, you may decide to enable satellites according to the subgroup to which they belong. For more information, see "Chapter 10. Performing a Mass Deployment" on page 145.

Setting the Execution Starting Point for a Satellite

You set the execution starting point for a satellite for one of two reasons:

- You are using test satellites to execute the test-level group batches. In this situation, you may need to reset the execution starting point. To perform this task:
  1. Select the test satellite from the satellite details view, and click the right mouse button.
  2. Select **Edit** from the pop-up menu.

     The Edit Satellite notebook opens.
  3. Select the Batches page.
  4. Set the satellite to begin executing each batch from the appropriate batch step.
  5. Click **OK**.
- You also set the execution starting point if you are migrating a system that was already running DB2 to DB2 Satellite Edition.

If you have satellites that were already configured when they joined the group, you will probably want them to begin executing the group-level batches of the application version at different batch steps than those satellites that have not yet executed batches, or are not yet properly configured. For more information, see “Planning the Migration to DB2 Satellite Edition” on page 207.

To perform this task, use the Set Execution Starting Point window, which is available from the Satellite Administration Center.

Chapter 6. Working with the Model Office

A model office is a special member of the test satellites of a group, as it is representative of the satellites in its group. You typically have one model office for each version of the application that is used in the group. Because the model office is representative of your satellites, you can use it perform administrative tasks for the group. For example, you can use the model office to re-create a problem that occurs on a production satellite. When you re-create the problem on the model office, you can then use it to create and test a fix for the problem.

The administrative tasks that you can perform using the model office differ depending on whether you are in the development and acceptance-testing phase, the production-deployment phase, or the post-deployment phase of an application version for a group. The sections that follow describe the different ways in which you can use the model office in the different phases.

Development and Acceptance-Testing Phase

When you build the model office, it becomes the model of the test and production satellites that you later deploy for a specific group. Although the model office does not have to be installed on the same hardware that you intend to use with the other satellites of the group, the model office should match its group satellites on several characteristics:

- The same installed components.

  When you install DB2 Satellite Edition on the model office, you should install the same components that you intend to install on the other satellites of the group. This shared characteristic is important when you use the

© Copyright IBM Corp. 1999

model office as the model for deploying production satellites. See "The Production-Deployment Phase" on page 92 for more information.

- The same DB2 database definition.

  The model office should have a DB2 instance with the same database manager configuration values, and a DB2 database with the same name and database configuration values that you intend to use on your production satellites. The model office should also have the same connection information (node, database and DCS directory information) as the other satellites. The connection information should include:

  – The definitions that are required to connect to the satellite control database on the DB2 control server
  – The DB2 systems with which the satellite will exchange data during replication, as well as the replication control database (which can reside on the DB2 control server, or on a different DB2 system).

- You should configure the model office to replicate a portion of the corporate data that is representative of the data to be maintained by the group satellites. For example, if you are deploying a sales application that supports the maintenance of customer data, then the data about the customers of a test sales person should be replicated to the model office.

  The replication setup on the model office should include the replication subscription information for data that is replicated with corporate databases. Typically, the subscriptions define the tables on the satellite as replicas, meaning that the replication process is the update-anywhere process.

  You should test the replication definitions. That is, you should test both the initial data that is loaded onto the model office from the corporate databases, and the actual replication process. You can perform the test by updating the corporate and satellite databases (if you are using replica tables). If the test is successful, the two databases will be synchronized when the Capture and Apply programs execute. To test the model office, you may have to set up a subscription for a test sales person who is assigned fictitious customer accounts in the corporate databases. You can use the update batch to initiate the replication process on the model office.

- The model office will have a specific version of the end-user application installed on it. You should install the same version of the application on the model office that you intend to install on the group satellites. You set the application version on the model office to ensure that, when it synchronizes, it executes the scripts for the correct version of the end-user application. Because the model office is a model of the deployed satellites, you should test the model office to ensure that it behaves similar to the other satellites. For example, you can use the update batch to initiate the replication process.

Creating the Model Office

You build the model office to represent the initial version of an end-user application during the development and acceptance-testing phase. As described in "Development and Acceptance-Testing Phase" on page 87, the model office should match the group satellites on several characteristics:

- It should have the same DB2 components installed on it.
- It should have the same DB2 database definition.
- The replication setup for the model office should result in the model office maintaining a portion of the corporate data that is representative of the data being maintained by the group satellites.
- It should have the specific version of the end-user application install on it that you are deploying.

The model office should be at this state before you begin the production-deployment phase. The sections that follow describe how to set up the model office so that you can use it as the model for the production-deployment phase.

Installing and Setting up the Model Office

Perform the following steps to install and set up the model office.

- Install the operating system on the model office.
- Install DB2 Satellite Edition on the model office. You should install the same components that you intend to install on the production satellites. See "Chapter 14. Planning for DB2 Satellite Edition Installation" on page 187 for information about the DB2 components that are unique to DB2 Satellite Edition. Use this information to determine which components you require in production.

  When you install the model office, you can provide values for:

  – The satellite ID
  – The user ID and password that are required to connect to the DB2 control server and the satellite control database during synchronization
  – The user ID and password required for the Remote Command Service on Windows NT.

  You can also set the application version when you install DB2 Satellite Edition. It may, however, be better to set the application version when installing the end-user application.

  When you install DB2 Satellite Edition, you can use the options that are available to create a database, enable it as a replication source, and make it recoverable. You can use these options if the installation application that you use to install the end-user application does not create a database.

You can perform the installation either interactively, or by using a response file. See "Chapter 15. Installing DB2 Satellite Edition" on page 195 and "Chapter 16. DB2 Satellite Edition Distributed Installation" on page 199 for more information about the different methods of installing DB2 Satellite Edition.

- You should catalog the remote databases and instances, including the DB2 control server and the satellite control database, on the model office. See "Cataloging Instances and Databases on the Model Office" on page 57 for information that you can use to simplify setting up the connection information.
- If you did not create a database on the satellite when you installed DB2 Satellite Edition, and the installation of the end-user application also does not create a database, create the database on the satellite.
- Install the end-user application. The installation application should set the application-version information that is required for synchronization. If the installation application does not set the application version, set it by using the **db2sync -s** command. For more information about this command, see "Appendix C. db2sync - Start DB2 Synchronizer" on page 243. If the installation application does not create the required tables, indexes and other database objects that are required by the application, create them.

  Note: If you are using update-anywhere replication, the target tables on the satellite are created by the SQL script that creates replication subscriptions.
- Perform any specific DB2 customization that is required to run the application. For example:
  - You can tune the environment for the application by setting database manager configuration parameters, database configuration parameters, and DB2 registry variables. Refer to the *Administration Guide, Performance* for more information.
  - Set any CLI/ODBC values that are required for the application. Consult the application documentation or the application developer for specific requirements. You can also refer to the *CLI Guide and Reference*.

Setting Up the DB2 Control Server to Handle the Model Office

If you have not installed and set up the DB2 control server to support synchronization, follow the steps in "Setting up the DB2 Control Server" on page 64.

The last steps in "Setting up the DB2 Control Server" on page 64 instruct you to create a group and test satellites. The group that you create should be the group you intend to use for your production deployment. When you use the Create Satellite window to add the model office to the group, provide a name for the model office that conforms to the conventions that you intend to use

for your model offices. You can use the **Description** field, and possibly the **Subgroup** field, to provide additional identification about the model office. When you complete your acceptance testing, you will use this model office to support the production-deployment and post-deployment phases for the application version in the group. You should set up the model office as a test satellite.

When you have set up the DB2 control server, you are ready to perform a synchronization test from the model office.

Running a Synchronization Test from the Model Office

See "Running a Test Synchronization" on page 74 for information about running a synchronization test. If problems occur, see "Synchronization Test Problems" on page 163 for information about how to recover from them.

Enabling the Satellite to Execute Group Batches and Replicate Data

You must create the application version and its associated group batches before the model office can execute these batches. See "Creating and Testing Group Batches" on page 75 for more information. The batches that you create will be at the test level so that you can modify them to obtain the results that you want.

If you intend to use data replication with this group, enable the model office to replicate data against a test subset of the data that will be replicated from the corporate data source. To perform this task, and you have not yet set up replication for the application version and generalized its subscriptions, follow the instructions in "Chapter 7. DB2 Data Replication" on page 95 to complete these tasks. Generalizing the replication subscriptions add batch steps to the setup and update batches to enable replication on the satellites in the group. When you have completed testing the replication setup by synchronizing one or more test satellites, your should perform a final test of the replication setup by using the model office. See "Testing Group Batches on the Model Office" for more information.

Testing Group Batches on the Model Office

When the group batches are created, you can synchronize the model office. You can use the **db2sync** command, or the application that you use for synchronization. When the model office synchronizes, it downloads and executes its group batches for its application version, then uploads the results of the execution to the DB2 control server.

Check the results. You should answer the following questions:

- Did the model office execute the batches successfully?

- Is the model office in the failed state?
- Do the output logs contain unexpected results?

If the update batch contains batch steps for data replication, and this is the first time that the model office has replicated, data is extracted from the source tables and loaded into the target tables. On subsequent replications, only changes to data on either the model office or the corporate data source is exchanged.

You should verify that the correct data is loaded on the model office. You can perform this task by running queries against the data tables, and by running the end-user application. If the data that is downloaded from the corporate data source is not correct, verify that the replication source and subscription definitions are appropriate. Modify the replication definitions as required. To reload the data, cold start the Capture program on the model office. The cold start forces the data from the corporate data source to be reloaded the next time that the model office synchronizes.

You must completely test the database definition and data that are created on the model office by the group batches. See "Checking the Results of the Synchronization Session" on page 82 for more information. You to may have to modify the group batches more than once before they produce the correct results.

Proceeding to the Production-Deployment Phase

When you complete the development and acceptance-testing phase, the setup of the model office should match what you intend to deploy on your production satellites. The next task is to use the setup on the model office as the model for performing a mass deployment during the production-deployment stage, as described in "The Production-Deployment Phase".

The Production-Deployment Phase

You can use the model office to generate the files that are required for automating the deployment of the group satellites. The installation process uses the generated files to install and set up the satellites. To generate the files, use two utilities that are provided by DB2 Satellite Edition:

- Use the response file generator to *reverse engineer* the installed model office, which produces an installation response file. You use the response file to install exact duplicates of the model office. When you perform the installation, DB2 Satellite Edition is installed with the same components and configuration values that are on the model office.

Note: The term *reverse engineer* refers to the process of extracting information from an existing system and producing a script from that system. The script can then be used to re-create a similar or exact copy of the original system.

- Use the client profile export utility to generate a configuration file. This file contains configuration and connectivity information about the model office. When you import the configuration file to the satellite, the configuration and connection data for the model office is re-created on the satellite.

When using the DB2 Universal Database distributed installation process, you can specify a client profile to import, which facilitates complete automation of the installation and configuration process. In this way, you can deploy all of your production satellites with a minimum of difficulty.

When the group satellites are deployed, you can simplify and automate the setup for replication on each satellite by reverse engineering the group's replication setup, so that it is general enough to be applied to each satellite of the group. You reverse engineer by generalizing the replication subscription.

Using the Model Office During the Production-Deployment Phase

See "Chapter 10. Performing a Mass Deployment" on page 145 for detailed instructions about using the model office during the deployment of an application version to a group of satellites. For information on generalizing the replication subscription to set up replication for the satellites in the group, see "Chapter 7. DB2 Data Replication" on page 95.

Post-Deployment Phase

After the group's satellites are deployed, the model office reflects the current state of a typical group satellite, though some differences may exist between the model office and a satellite. For example, the hardware configuration or the operating system may be different on the model office.

Because the model office is configured to run one version of the end-user application that is being used by the group satellites, you can use the model office to re-create the problems that occur on a production satellite. This means that you can use the model office to define and test fix batches to correct these problems.

The model office is vital, as it provides existing tools, like the Control Center, a working and representative database against which to operate. You can generate scripts using the SmartGuides, notebooks, and windows of the Control Center against the model office. You then submit the scripts for execution against the model office and examine the changes caused by them,

without affecting any production satellites. When you are satisfied with the results that the scripts produce, you can add the scripts to the group batches (or to a fix batch), as required. The scripts will be executed by the group satellites when they synchronize.

You should take backup images of the model office. If a change you make while testing a fix batch for a group satellite produces unexpected results, you can use the backup image to restore the model office. After the model office is restored, it will be at a known state, and you can try a different fix. See "Chapter 9. Recovering the Satellite Environment" on page 129 for more information.

Using the Model Office System During the Post-Deployment Phase

When your users begin to use the production satellites to run the application, unanticipated problems can occur. For example, a performance problem could occur. Because the model office represents the deployed satellites, you can use it to investigate the problem, and to develop a test script to correct it.

- Using tools such as the Control Center and the Command Line Processor, you can examine the model office and determine which changes are required to address a production problem.
- You can use the Control Center to make changes to the model office to correct problems. Instead of using the Control Center tools to complete the changes, use the Show SQL and Show Command options to generate and save scripts. You can then create a new test level of the batches for the application version, and add the scripts to the setup, update and cleanup batches, as required. The next time the model office synchronizes, it will execute the new batch steps. You can determine the effect of the changes by reviewing the execution logs, and by using the Control Center tools to examine the model office. If the changes are correct, the next step would be to promote the test level to production for the production satellites to execute.

The model office has already executed the batches that you promoted to production. When the rest of the group satellites next synchronize and execute the new batch steps, the model office will represent the group satellites that are at the same application version. If another problem occurs, you can again use the model office to investigate the problem, and to develop and test scripts to correct the problem.

Chapter 7. DB2 Data Replication

In the satellite environment, you can have group satellites synchronize on more than just the database definition. You can also have the satellites synchronize data with corporate databases.

For information that is supplementary to the information here, refer to the following Web site:

`www.software.ibm.com/data/db2/library/satellite`

To accomplish data synchronization, you first have to set up and test a replication environment, independent of the satellite environment. Then, when you are satisfied with the results you obtain from the replication environment, you apply the replication definitions to the satellite environment. To apply the replication definitions to the satellite environment, you generalize these definitions to the satellites of a specific group by using the Generalize Replication Subscriptions window of the Satellite Administration Center. When you use the Generalize Replication Subscription window, the DB2 control server will either create or modify the group's setup batch, adding batch steps that create the tables on the satellite and the replication control server that are required for data replication. These tables are based on the source-target replication definitions that you generalized. The DB2 control server will also create or modify the update batch, adding a batch step to call the **asnsat** command, which, in turn, calls the Capture and Apply programs. Figure 4 on page 96 provides an overview of the steps that you perform both to set up the replication environment, and to activate replication in the satellite environment.

Note: Both the replication and the satellite environments use the term *promote*. This term, however, has a different meaning in each environment. In the replication environment, promote means to copy the replication definitions from a test to a production environment, and no satellites

© Copyright IBM Corp. 1999

are involved. In the satellite environment, promote means to change the level of an application version from a test level (which is executed by test satellites) to a production level (which is executed by production satellites).

Setting up Data Replication		
Steps	Replication Sources	Replication Targets
Step 1. Set up the replication test environment. The replication test environment is outside of the satellite environment.	Define the replication sources. To perform this task, you can use either the DJRA tools, or the Control Center. The replication source is expected to be either a test database or a test subsystem. The DB2 system that you use must be the same type of DB2 system that you intend to use as the production source system when replicating to satellites.	Define the replication subscriptions. To perform this task, you can use either the DJRA tools, or the Control Center. The replication target is expected to be either a test database or a test subsystem. You must use a DB2 Universal Database system as the replication target.
Step 2. Promote the replication test environment to a replication production environment.	If you used a test database or test subsystem when you defined the replication source, use the DJRA tools to promote the source table definition to a production database.	If you promoted the replication source, then you should promote a subscription to ensure that the replication control tables that record the source and target databases have been updated.
Setting up the Satellite Environment		
Step 3. Generalize the subscription	Use the Generalize Replication Subscription window of the Satellite Administration Center to perform this task. Test batch steps are written to the setup and update batches of the test level of the application version. These batch steps both set up and initialize data replication for the satellites. You can test the replication by having test satellites execute these batches.	
Step 4. Promote the test level of the application version.	When you are satisified with the results that you obtain on the test satellites from the test level of the application version, promote the test level. When the production satellites synchronize, replication will be set up for them, and the satellites will replicate data.	

Figure 4. Overview of Setting up and Activating Replication for the Satellite Environment

Satellites can replicate data to and from all sources supported by the DB2 product family.

The sections that follow describe how to set up, configure, and start data replication in the satellite environment.

Setting up Data Replication

You set up replication for the satellite environment much like you set up replication for any other environment. Perform the following tasks to set up replication for the satellite environment:

1. Create the replication environment.

   You set up a separate replication environment, which is independent of the satellite environment. You set up the replication control tables, replication source or sources, and the replication subscriptions in a test environment. After testing the replication definitions, you may want to promote these definitions to a production environment. For more information, see "Creating the Replication Environment".

2. Set up the satellite environment.

   Setting up the satellite environment entails setting up the DB2 control server, creating a group, satellites for the group, an application version, group batches, scripts, targets, success code sets, and authentications. See "Chapter 5. Setting up and Testing Your Satellite Environment" on page 63 for the details on how to set up the satellite environment. To enable the group satellites to replicate data, you must perform one additional task, which is to generalize the replication subscriptions from the replication environment to the satellite environment. When you generalize the replication subscription, the generalize function creates batch steps which, when executed by the satellite, will define the replication subscription for that satellite for its application version. "Enabling the Satellite Environment for Replication" on page 99 describes how to set up the satellite environment for replication.

Creating the Replication Environment

When you create the replication environment, you set up the replication control server, and define the replication sources and subscription sets in a replication test environment. When the replication test environment is fully tested and performs to your satisfaction, you can promote this test environment to a production environment.

1. Create your replication test environment.

   Create the replication control tables, define the replication sources, subscription sets and replication subscriptions using the Control Center or the DB2 DataJoiner Replication Administration (DJRA) tool.

   You can install the DJRA tool on the same machine as the Control Center by going to the \sqllib\djra directory on the Control Center instance. Run the **djra** executable in this directory to start the DJRA installation program. Refer to the online help provided with the DJRA installation program for more information.

For information about creating the replication definitions, and about starting the DJRA tool, refer to the *Replication Guide and Reference*. For information about how to use the DJRA tool, refer to the online help that is provided with it. Consider the following when you decide on where to locate the replication databases:

- The replication control tables can be generated either automatically or manually at the replication source, target, and control databases using the Control Center or the DJRA tool. These tables are always required at the replication source and control database, and at the target for update-anywhere replication.
- Define the replication source or sources in the same type of DB2 database that you intend to use when you deploy the production satellite environment. For example, if you want the production satellites to replicate data from MVS, set up the source or sources in DB2 for OS/390. In this situation, do not place the sources of the test environment in DB2 Universal Database. You should select a test environment for the location of the replication sources.
- Define the replication subscriptions in DB2 Universal Database. Because you will later generalize the subscriptions for the satellite environment, the target database for the subscriptions must be DB2 Universal Database. Only consider using a test database on the same system as the DB2 control server if the DB2 control server serves a small number of satellites. Otherwise, use a different system.
- Specify the replication control server when defining the replication subscriptions in DB2 Universal Database. You initially set up the replication control server in a non-satellite DB2 Universal Database database. Only consider using a test database on the same system as the DB2 control server if the DB2 control server serves a small number of satellites. Otherwise, use a different system.

2. Promote the replication sources and subscriptions from the test environment to the production environment.

   You can use the DJRA promote functions to reverse engineer the setup of the test replication environment to move it to production databases. You must perform this step to move the replication source or sources from a test database to the production database from which the production satellites will access data.

   Note: You can only use the DJRA tools to promote between the same type of database. That is, if you use DB2 Universal Database databases in the test replication environment, you must use DB2 Universal Database databases in the production replication environment.

The steps that follow provide an overview of how to promote the test replication environment to production. For detailed information, refer to the online help that is available from the DJRA tool.

a. Use the DJRA promote table function to move the replication source table definitions from the test database to the production database. Edit the generated script to connect to the production database. The satellites will replicate data from this database.
b. Move the table source registration. Use the DJRA promote registrations function to perform this task, then edit the generated script to connect to the production database.
c. Use DJRA to promote the subscription. Specify the name of the production system that you will use as the replication source. You can also specify a new replication control server and target server.

   The target database must be a DB2 Universal Database database. Only consider using a production database on the same system as the DB2 control server if the DB2 control server serves a small number of satellites. Otherwise, use a different system. You will generalize the replication subscriptions for the satellite environment from this database.

   The replication control server can be any non-satellite database. You would typically use the target production server.

Notes:

1. The online help for the Satellite Administration Center contains information about the Promote Source and the Promote Replication Subscription windows. These windows are not available with the Control Center for Version 6.1. For the interim, to promote replication sources and subscriptions, use the promote functions that are supplied with the DB2 DataJoiner Replication Administration (DJRA) tool.
2. The replication test and production environments are not associated with the satellite environment. For replication to work in the satellite environment, the subscription sets of either the replication test environment or the replication production environment must be generalized for the satellite environment. For more information, see "Enabling the Satellite Environment for Replication".

Enabling the Satellite Environment for Replication

After you create the replication environment, you must set up the satellite environment, and generalize the subscription sets for it. As a basic requirement, you must have already created a group, and an application version for it. For information on creating a group, see "Creating a Group" on page 70. For information on creating an application version, see "Creating an Application Version" on page 78.

In addition, you must ensure that you have cataloged the production replication control server, and all production replication sources and targets at the DB2 control server instance. If the Control Center's JDBC server and the DB2 control server are on the same instance, see "Cataloging Instances and Databases in the Control Center Instance" on page 51 for the procedure. If this condition is not true, you must still catalog the same objects at the Control Center, then use the Client Configuration Assistant to export the production replication control server database, and the production replication source and target databases, and import the resulting file at the DB2 control server instance using the **db2cfimp** command. The procedure to follow is similar to that described in "Using a Client Profile to Perform Cataloging on the Model Office" on page 58, except that you only select the replication-related databases on the Customize Export window.

Note: Typically, you will not create the replication control tables on a satellite. Instead, you will use a DB2 server that is not a satellite as the replication control server. One situation exists, however, in which you may want to consider using the satellite as the replication control server. This situation occurs when you have a very small group of satellites functioning as target servers, and one source server. Placing the replication control tables on the satellite can result in better performance by the Apply program. If you decide to use the satellite as the replication control server, however, you do not have the centralized reporting of replication information that occurs if you use a central replication control server. In addition, if a replication-related error occurs on any of the satellites of this group, you will have to perform maintenance on that satellite manually.

You can now create a generalized replication subscription to be used in the satellite environment.

Generalizing Replication Subscriptions

You generalize replication subscriptions to create the replication setup for the satellite environment. When you generalize, you reverse engineer the subscription definitions for the single production replication target that you defined, which creates parameterized scripts that will be used to define all of the satellites as replication targets.

The DB2 control server stores these scripts in the satellite control database, and adds test batch steps to the setup and update group batches for the application version that you specify. The scripts of these test batch steps set up the tables that are required for replication on the satellites, and specify, by use of a WHERE clause, the rows that each satellite will replicate. In addition, the DB2 control server also adds a batch step to the update batch to invoke the Capture and Apply programs through the **asnsat** command.

To generalize replication subscriptions, use the Generalize Replication Subscriptions window. For more information about this window, refer to the online help that is available from the Satellite Administration Center. In this window, you must specify the replication control server that contains the production subscription information for the subscription that you want to generalize, as well as the application version that corresponds to the end-user application that will be using the replicated data.

When you generalize the subscription, the replication definitions, by default, apply to all the satellites of the group. If you want, you can modify the definitions for the entire group, or you can modify the definitions for a subset of the group satellites. For more information, see "Changing Replication Parameters" on page 102.

If you generalize the replication subscription before creating level 0 for an application version, the DB2 control server automatically creates level 0. The DB2 control server also creates a setup batch called `setup` and an update batch called `update` for level 0. The batches are created with the test batch steps that are required for data replication.

Editing the Setup Batch

In some situations, you may need to edit the setup batch. For example:

- If you generalize the replication subscription before creating the setup batch for the application version, you should add the batch steps to create the tables on the satellite that you want to be replicated. The batch steps that create the tables should precede the batch steps that set up data replication.

  Notes:

  1. If you want a table to be replicated, the script that you use to create it must have the DATA CAPTURE CHANGES option specified for the CREATE TABLE statement. This parameter causes extra information about changes to the table's data to be written to the logs. For more information about the CREATE TABLE statement, refer to the *SQL Reference*.
  2. If the table contains LONG VARCHAR columns, and these columns are to be replicated, you must alter the table using the ALTER TABLE statement and the DATA CAPTURE CHANGES INCLUDE LONGVAR COLUMNS option. For more information about the ALTER TABLE statement, refer to the *SQL Reference*.
- If one of your target table types is a replica table, the control tables for the Capture program must exist on the satellite. See "Creating Control Tables for the Capture Program" on page 102 for more information.

Creating Control Tables for the Capture Program

If the subscription that you generalize is for a replica table, the control tables that are required for the Capture program must exist on the satellite *before* replication can occur. You can create these tables by executing the **dpcntl.sat** SQL script file on the satellite. The script contains a series of CREATE TABLE statements that must be executed on the satellite.

During installation, this command file is automatically executed if you create a database *and* specify that the database will be used as a replication source. If this is not the situation, you must execute the command file to create the control tables.

You can find the **dpcntl.sat** script in the sqllib\samples\repl directory on the satellite.

You can also invoke the **dpcntl.sat** script from a batch step in the setup batch. Ensure that you specify that the execution target is the operating system on the satellite. The batch step that executes the **dpcntl.sat** script must immediately precede the batch steps that configure replication. The **dpcntl.sat** script also sets the default values in the tuning parameters table. To invoke the script from a batch step, use the following script for the batch step:

```
db2 connect to server-name USER authorization-name USING password
-tvf x:\sqllib\samples\repl\dpcntl.sat
db2 connect reset
```

Where *x* is the drive on which you installed DB2 Satellite Edition.

Changing Replication Parameters

In some situations, you may need to customize the generalized replication subscription. For example, you may want to modify the WHERE clause for one or more group satellites to have them replicate different data than other satellites in the same group. You may also want to change the replication control server for one or more of the group satellites to more evenly distribute the work load on the replication control servers.

To work with the generalized replication subscription, use the Change Replication Parameters notebook. From this notebook, you can specify the satellites for which you want to change the replication parameters. You can also use this notebook to select the WHERE clause for a specific source-target table pair, then modify the clause using the Change Subscription Definition window. You can also use the Change Replication Parameters notebook to change the replication control server for one or more group satellites. For more information about working with replication parameters, refer to the online help that is available from the Satellite Administration Center.

Testing Replication on a Satellite

To test the replication setup, you require a test satellite. The test satellite must be known to the DB2 control server. For more information, see "Creating Test Satellites" on page 70. For information about creating and setting up a satellite so that you can use it to test batches, see "Installing and Preparing a Satellite for Synchronization" on page 71. Remember to run a synchronization test from the satellite to verify that the correct configuration exists both at the DB2 control server, and at the satellite. Use the **db2sync -t** command to perform the synchronization test. The sections that follow describe additional considerations and steps that you need to perform.

Setting the logretain Database Configuration Parameter on the Satellite

If you did not create a database during installation and specify that the database would be used as a replication source, or that the database would be recoverable, you must now set the value for the *logretain* database configuration parameter:

- If you want to capture data and enable the database for forward recovery, set the *logretain* database configuration parameter to `Recovery`.
- If you want to capture data and not enable the database for forward recovery, set *logretain* to `Capture`.

  When *logretain* is set to `Capture`, the Capture program automatically calls the **prune logfile** command when it completes its *cycle*. For more information, see "Logging and DB2 Data Replication" on page 141.

Notes:

1. Both the Capture program and the Apply program have a cycle. The cycle of the Capture program is the process of the Capture program identifying replication sources, reading the DB2 logs or journal to identify the point up to which the previous cycle captured changes, capturing the changes that occurred subsequent to the previous cycle, and updating the control tables with reference information that can be used by the next cycle of the Capture program.

   During the Apply program cycle, the Apply program checks whether any replication subscriptions are due for replication. If changed data exists, the Apply program applies the data to the appropriate targets. During the process, different control tables are updated so that the Capture and Apply programs can identify what data was replicated, when this data was replicated, and which tables were updated.
2. When you change the value of the *logretain* database configuration parameter to `Recovery`, the database moves to the *backup pending* state. You must back up the database before it can be accessed.

Synchronizing the Test Satellite

The first time that the test satellite synchronizes, it downloads the batch steps that contain the appropriate SQL statements and commands to set up replication on the satellite.

The **asnsat** command is invoked by one of the batch steps of the update batch during synchronization. The **asnsat** command automatically starts the Capture and Apply programs with the optimal parameters for the satellite environment. The **asnsat** command bypasses the Capture program or the Apply program, as necessary. The Capture and Apply programs self-terminate after each has completed a cycle.

Note: If the **asnsat** command must access a DB2 database that is remote to the satellite for replication, ensure that the authentication credentials that you create for the replication batch step has adequate privileges on the tables at the remote database. If the **asnsat** command only executes against the database on the satellite, the **asnsat** command executes with SYSADM authority when it is started as part of a synchronization session.

After the test satellite synchronizes, examine the results of the data replication. See "Checking the Results of the Synchronization Session" on page 82 for additional information. In addition, you should verify that the data that was replicated to the test satellite is the correct subset of the corporate data that this satellite should receive. You can perform this task by issuing SQL queries against the data. You can also use the end-user application to verify the data.

If errors occurred, correct them, and synchronize again. When you are satisfied with the results on the test satellite, you should have the model office synchronize to replicate data. Because the model office represents and is configured as you intend to configure your production satellites, this final test will help validate the behavior of the replication setup in the production environment.

Additional Considerations

The sections that follow provide additional information, including how to override how the **asnsat** command invokes the Capture and Apply programs, and replication restrictions that apply in the satellite environment.

Modifying Capture and Apply Program Parameters

The generalize replication subscription function inserts a batch step in the update batch of the application version to call the Capture and Apply programs from the **asnsat**. The command is inserted in the batch step with the optimized parameters for the Capture and Apply programs in the satellite environment. In some situations, these parameters may not be appropriate for your environment.

The default parameters for the Capture program are: WARM, PRUNE, NOTRCTBL, LOGREUSE, LOGSTDOUT, NOTRCTBL, TRCFILE, and AUTOSTOP. The default parameters for the Apply program are: NOINAM, NOTRC, NONOTIFY, LOGREUSE, LOGSTDOUT, TRLREUSE, TRCFILE, COPYONCE, and LOADX. You can override these parameters by editing the batch step that contains the **asnsat** command.

For more information about the **asnsat** command, and its invocation parameters for the Capture and Apply programs, refer to the *Replication Guide and Reference*.

Restrictions

Restrictions for the satellite environment are as follows:

- The enhanced level of conflict detection is not supported. If you specify the enhanced level, the Apply program assumes a standard level of conflict detection.
- The compiled ASNLOAD exit routine works only if your target definitions match your source definitions. Parameters such as number of columns, name of columns, type, and length must be exactly the same for both your target and source. If your target definitions do not match your source definitions, you need to customize and recompile the ASNLOAD sample source code or use the NOLOADX option. If your ASNLOAD exit routine requires a password, you must provide a replication password file.

  Note: The replication password file is not the same file as the satadmin.aut file on the satellite. For more information about the replication password file, refer to the *Replication Guide and Reference*.

Chapter 8. Building Applications

To build a synchronizing application for the satellite environment, you can use either the DB2 Personal Developer's Edition, or the DB2 Universal Developer's Edition. Either product includes the tools that you need to build applications that run on the Windows 32-bit operating systems.

Note: Neither of these products is available from the DB2 Satellite Edition CD. You can obtain them from a different DB2 Universal Database product CD. For information on how to install either product, refer to the *Quick Beginnings* for the DB2 Universal Database product that you have.

Figure 5 on page 108 shows one method in which the synchronizing application that you build can interface with the database on the satellite and the satellite control database on the DB2 control server in a development environment. You can contrast this possible development configuration with how the satellite synchronizes in a production environment, which is shown in Figure 2 on page 10. The development environment shown in Figure 5 on page 108 includes the following:

- A compiler or interpreter
- A satellite, including the operating system, DB2 Satellite Edition (the database manager, DB2 SDK, a client connection to the DB2 control server using TCP/IP, and a local satellite database)

  Note: The DB2 SDK is not available from the DB2 Satellite Edition CD. You can, however, install the DB2 SDK from a different DB2 Universal Database product CD onto the satellite.
- A DB2 control server

© Copyright IBM Corp. 1999

- The satellite control database (SATCTLDB) on the DB2 control server.

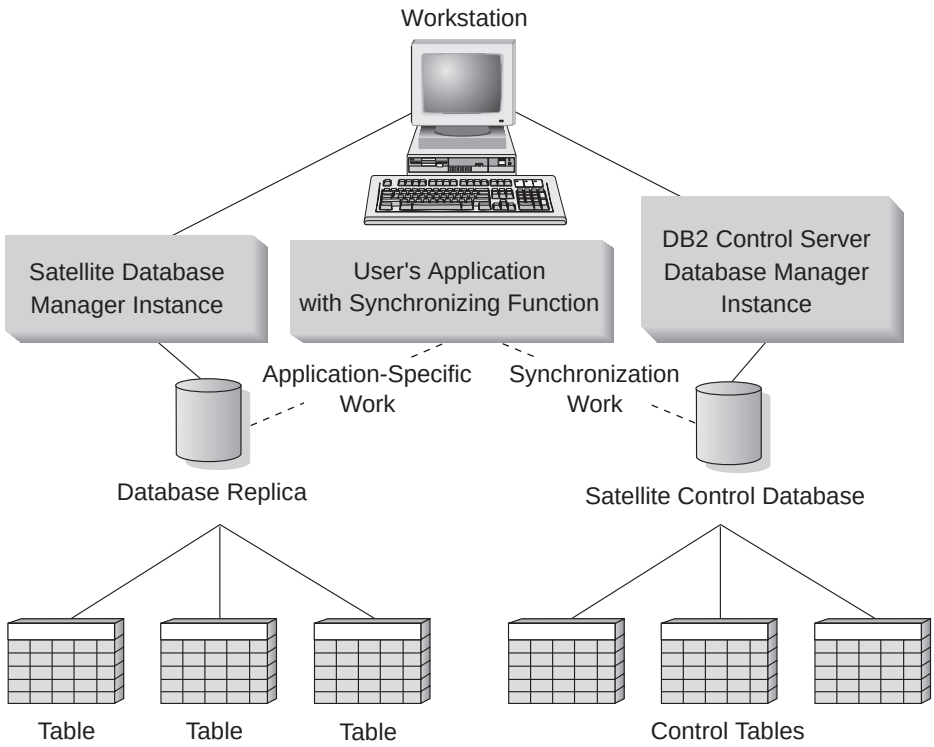


Figure 5. Synchronizing Application Program

When building applications, you can find relevant information in the following books that are provided with DB2 Universal Database:

- The *Application Development Guide* describes how to build non-DB2 CLI applications that access DB2 databases.
- The *CLI Guide and Reference* describes how to build DB2 CLI applications.
- The *Application Building Guide* describes how to set up the application development environment, and how to compile, link, and run DB2 applications. This book also contains information about problem determination.
- The *SQL Reference* describes the syntax of SQL statements and functions.
- The *Administrative API Reference* describes the administrative functions used to manage DB2 databases. The full set of DB2 application programming interfaces (APIs) are available for the satellite environment.

  Note: The APIs for synchronization are available in C only.

- The *Command Reference* describes the DB2 commands, and how to use the DB2 Command Line Processor.
- The *Troubleshooting Guide* describes how to determine the source of, and recover from, problems. This book also describes how to use diagnostic tools.
- The *Message Reference* lists the full set of DB2 messages, and is useful for debugging applications.

For problem determination information that is specific to the satellite environment, see "Chapter 11. Problem Determination" on page 161.

When building your applications, you can use a variety of tools to write and test the SQL statements in your applications, and to monitor their performance. You can use the Satellite Administration Center to both set up and administer the satellite environment. You use the DB2 Synchronizer application to both test and validate the set up of the satellite environment.

The sections that follow provide information that is specific to setting up the development environment, and building an application for the satellite environment.

Supported Interfaces

The application that you build must be suitable for the Windows 32-bit operating systems. This includes the Windows NT, Windows 95, and Windows 98 platforms.

You can build your application to use any of the supported programming interfaces. These interfaces include:

- DB2 APIs. Included in this interface are the C APIs that are specific to synchronization.
- DB2 Call Level Interface (DB2 CLI).
- Embedded SQL.
- Embedded SQL for Java (SQLJ).
- Java Database Connectivity (JDBC).

The DB2 Software Developer's Kit (DB2 SDK) for Windows 32-bit operating systems provides the tools and environment that you need to build applications that access local and remote databases on the Windows 32-bit operating systems. For more information about the DB2 SDK, refer to the *Application Building Guide*.

You should use the DB2 precompiler that is supplied for the Windows 32-bit operating systems, and not the precompiler that is available with the compiler that you are using. The DB2 SDK supports Windows NT Version 4.0 or later (both workstation and server versions), Windows 95, and Windows 98. For the satellite environment, you should build your applications using Microsoft Visual C++ Version 5.0 and Version 6.0, or IBM VisualAge C++ for Windows, Version 3.5 and Version 4.0.

Setting up the Development Environment

To set up the development environment, you require the following:

- A compiler or interpreter

  You should ensure that your compiler or interpreter environment is correctly set up by first building a non-DB2 application. If a problem occurs, refer to the product documentation for your compiler or interpreter.
- A satellite, including the operating system, DB2 Satellite Edition (the database manager, DB2 SDK, a client connection to the DB2 control server using TCP/IP, and a local satellite database)

  Note: The DB2 SDK is not available from the DB2 Satellite Edition CD. You can, however, install the DB2 SDK from a different DB2 Universal Database product CD onto the satellite.
- Verification that the satellite and the DB2 control server are correctly set up. You can accomplish this task by performing a synchronization in test mode using the DB2 Synchronizer application. See "Chapter 5. Setting up and Testing Your Satellite Environment" on page 63 for more information.

  If you want to try a synchronization, you require an application version and group batches. See "Creating and Testing Group Batches" on page 75 for details. The application version must be created using the Satellite Administration Center, and the application version must be recorded on the satellite.
- A DB2 control server
- The satellite control database (SATCTLDB) on the DB2 control server.

Setting up the Windows 32-Bit Operating System Environment

When you install the DB2 SDK on Windows NT, the install program updates the Windows NT configuration registry with the environment variables, INCLUDE, LIB, PATH, DB2PATH, and DB2INSTANCE. The default instance is DB2.

See the *Application Building Guide* for the Java environment variables that are updated by DB2.

When you install the DB2 SDK on Windows 95 or Windows 98, the install program updates the autoexec.bat file.

You can override these environment variables to set the values for the machine or the currently logged-on user. To override these values, use any of the following:

- The Windows NT control panel
- The Windows 95 or Windows 98 command window
- The Windows 95 or Windows 98 autoexec.bat file.

Note: Exercise caution when changing these environment variables. Do not change the DB2PATH environment variable.

These environment variables can be updated for running most programs on Windows 32-bit operating systems. In addition, you must take the following specific steps for running DB2 applications:

- When building C or C++ programs, you must ensure that the INCLUDE environment variable contains %DB2PATH%\INCLUDE as the first directory.
- Ensure that the LIB environment variable points to %DB2PATH%\lib by using:

  ```
  set LIB=%DB2PATH%\lib;%LIB%
  ```

- Ensure that the DB2COMM environment variable is set at the DB2 control server of the satellite control database to support communications over TCP/IP.
- Ensure that the security service has started at the server for SERVER authentication, and at the client, depending on the level of CLIENT authentication. To start the security service, use the **net start db2ntsecserver** command.

  Note: If you install DB2 Satellite Edition as the local machine user, you cannot specify CLIENT authentication on the satellite. You can only specify SERVER because the security service is not available on the satellite.

Notes:

1. All DB2 environment variables can be defined in the user's environment or set up as registry variables. Refer to the *Administration Guide* for information about registry variables. You use the **db2set** command to set registry variables. Refer to the *Command Reference* for information about this command.
2. DB2INSTANCE should only be defined at the user environment level. It is not required if you make use of the DB2INSTDEF registry variable, which defines the default instance name to use if DB2INSTANCE is not set.

3. The database manager on a Windows NT environment is implemented as an NT service, and hence does not return errors or warnings if the service is started successfully, though other problems may have occurred. This means that when you run the **db2start** or the **net start** command, no warnings will be returned if any communication subsystem failed to start. Therefore, you should always examine the NT Event Log or the DB2DIAG.LOG file for any errors that may have occurred during the running of these commands. For information about the DB2DIAG.LOG file, refer to the *Troubleshooting Guide*.

Enabling Communications

Communications between the DB2 control server and the satellite must be conducted using TCP/IP. To set up communications, perform the following steps on the DB2 control server:

Note: Under normal conditions, when the DB2 control server instance is created by the install on Windows NT or AIX, the instance is set up for inbound connections using TCP/IP. You may not need to perform these steps.

1. Set the DB2COMM environment variable as follows:
   ```
   db2set DB2COMM=tcpip
   ```
2. Set up the Services file to configure TCP/IP support. Refer to the *Quick Beginnings* for the appropriate platform for details.
3. Set the value of the *svcename* database manager configuration parameter to the same value as the name in the services file.

Notes:

1. If the satellite is remote to the DB2 control server, or is running a different version of DB2, or runs a different operating system, you need to bind the database utilities on the satellite, including the DB2 CLI, to the SATCTLDB database on the DB2 control server. See "Binding Utilities" on page 113 for details.
2. If the satellite is running the Windows NT operating system, you must ensure that the DB2 Remote Command Service is started. To determine whether the service is running, click **Start** and select **Settings -> Control Panel**. When the Control Panel window opens, double-click on the Services folder. Look for the DB2 Remote Command Service in the window. If this service is running, it has a status of `Started`.

Cataloging the Satellite Control Database

When you catalog the satellite control database, the database directory on the satellite is updated with the alias of the database that the application needs to access using a CONNECT statement. When the database manager processes requests from the application on the satellite, it uses the database alias to both find and connect to the database.

To access the satellite control database on the DB2 control server, both the DB2 control server and the satellite control database must be cataloged on the satellite. In addition, the satellite must have the correct authentication credentials to access the satellite control database. To set up the catalog and authentication information, you can use the DB2 Synchronizer application in test mode. For more information about running the DB2 Synchronizer application in test mode, see "Running a Test Synchronization" on page 74 and "Using the DB2 Synchronizer Application" on page 127.

If you do not use the DB2 Synchronizer application, you can manually catalog the satellite control database and the node on which it resides by issuing the **catalog node** and **catalog database** commands on the satellite:

```
db2 catalog tcpip node nodename remote hostname server service-name
db2 catalog database satctldb at node nodename
```

For more information about the **catalog** commands, refer to the *Command Reference*.

You do not need to catalog the satellite control database on the DB2 control server. The database is automatically cataloged on the DB2 control server when the database is created.

Binding Utilities

If the satellite is remote to the DB2 control server, or is running a different version of DB2, or runs a different operating system, you need to bind the database utilities on the satellite, including the DB2 CLI, to the SATCTLDB database on the DB2 control server.

Binding creates the package that the database manager needs to access the database when an application is executed. Binding can be done explicitly by specifying the **bind** command against the bind file that is created during precompilation.

The *Command Reference* provides general information about binding the database utilities. This following example provides specific instructions to bind the database utilities to the SATCTLDB database.

On the satellite:

1. Click **Start** and select, for example, **Programs -> DB2 for Windows NT -> Command Window**

   The command window opens.
2. Connect to the SATCTLDB database. At the prompt, enter:

   ```
   db2 connect to satctldb
   ```
3. Bind the utilities to the database by entering the following **bind** command (for formatting purposes, the command is shown here across two lines):

   ```
   db2 bind %DB2PATH%\bnd\@db2ubind.lst blocking all
   sqlerror continue messages bind.msg
   ```

   Where %DB2PATH% is the path where DB2 is installed.
4. Exit the command window, and verify that the bind was successful by checking the bind message file, bind.msg. This file is in the path specified by %DB2PATH%.

The development environment is now set up.

Using DB2 APIs

Refer to the *Application Building Guide* for general information about building DB2 applications. Specifically, refer to the information that applies to building applications for Windows 32-bit operating systems. In addition, you should familiarize yourself with the information that applies to using the DB2 C/C++ APIs. The APIs that are used for synchronizing are provided in C only.

Each DB2 SDK includes sample programs that call DB2 APIs. The sample programs for Windows 32-bit operating systems are in the %DB2PATH%\samples\cli directory. You can build these sample programs for the Microsoft Visual C++ and IBM VisualAge C++ for Windows compilers using the DB2 API build files that are supplied with the DB2 SDK. You can also use the makefiles that are supplied. Both the makefiles and the build files show you the compiler options that you can use. These options are defined for each platform's supported compilers. You may need to modify the options for your environment. Refer to the *Application Building Guide* for information.

After you build the sample programs, you can use them as templates for creating your own applications.

Satellite APIs

The following sections describe the satellite APIs that you can use to set up and test the satellite environment, and to synchronize a satellite. For the full list of available APIs, refer to the *Administrative API Reference*.

Setting the Application Version on the Satellite

Before a satellite can synchronize with its DB2 control server, two attributes of the satellite's environment must be configured: the unique satellite ID, and the application version. You do not use an API to set the satellite ID. Instead, you use the **db2set** command to set the value of the DB2SATELLITEID registry variable, or you use the logon ID of the satellite user as the satellite ID. See "Setting the DB2SATELLITEID Registry Variable" on page 72 for information on setting the satellite ID.

You can use the db2SetSyncSession API to set the value of the application version on the satellite:

```
SQL_API_RC SQL_API_FN                  /* Set sync session for a satellite    */
  db2SetSyncSession (
   db2Uint32 versionNumber,            /* Database version number             */
   void * pParmStruct,                 /* Input parameters                    */
   struct sqlca * pSqlca);             /* SQLCA                               */
```

Note: The application version is case sensitive.

The maximum length of an application version is:

```
#define SQL_SYNCSESSIONID_SZ 18
```

When you set the application version for a satellite, use the following structure for the pParmStruct to define the string:

```
typedef struct db2SetSyncSessionStruct
{
   char            *piSyncSessionID;   /* ID for sync session                 */
} db2SetSyncSessionStruct;
```

If you supply an empty string for piSyncSessionID, you will reset the satellite's application version to NULL.

When setting the application version, the most common return codes are:

- SQL3942I Synchronization session identifier was set successfully for the satellite.
- SQL3943N Synchronization session identifier exceeds the maximum length of 18 characters
- SQL3944I Synchronization session identifier was reset successfully for the satellite.
- SQL3946N Synchronization session identifier operation failed.

Retrieving the Application Version

You can use the db2GetSyncSession API to query the value of the application version on the satellite:

```
SQL_API_RC SQL_API_FN                  /* Get sync session ID for a          */
                                       /* satellite                          */

  db2GetSyncSession (
   db2Uint32 versionNumber,            /* Database version number            */
   void * pParmStruct,                 /* Output parameters                  */
   struct sqlca * pSqlca);             /* SQLCA                              */
```

When you retrieve the current application version for a satellite, supply the following structure for the pParmStruct to retrieve the value:

```
typedef struct db2GetSyncSessionStruct
{
   char            *poSyncSessionID;   /* ID for sync session                */
} db2GetSyncSessionStruct;
```

You must supply a buffer at least as large as SQL_SYNCSESSIONID_SZ (+1 for the NULL terminator) to receive the application version.

When retrieving the application version, the most common return codes are:

- SQL3945I Synchronization session identifier for the satellite was retrieved successfully.
- SQL3946N Synchronization session identifier operation failed.

Testing the Satellite Environment

You can test the ability of a satellite to synchronize with its DB2 control server by executing the db2SyncSatelliteTest API on the satellite:

```
SQL_API_RC SQL_API_FN                  /* Test Satellite's ability to        */
                                       /* synchronize                        */

  db2SyncSatelliteTest (
   db2Uint32 versionNumber,            /* Database version number            */
   void * pParmStruct,                 /* Input/Output parameters             */
   struct sqlca * pSqlca);             /* SQLCA                              */
```

Because the satellite ID and application version are retrieved from the satellite environment, no parameter structure (pParmStruct) is required.

This API validates the following required configuration elements, which are necessary for synchronization to occur:

- The DB2 control server is cataloged at satellite (that is, the SATCTLDB database is cataloged).
- The release level of DB2 on both the satellite and its DB2 control server are compatible.
- The satellite ID is set on the satellite.

- The application version is set on the satellite.
- The satellite can communicate with the DB2 control server.
- The authentication credentials that the satellite uses for authentication with the DB2 control server are correct.
- The satellite is defined at the DB2 control server.
- The satellite is enabled for batch execution at the DB2 control server.
- The satellite is not in the failed state at the DB2 control server.
- The satellite's application version exists for the satellite's group at the DB2 control server

When testing the satellite environment, the most common return codes are:

- SQL3911I Test synchronization session completed successfully.
- SQL3931W The test synchronization session completed successfully. The satellite ID, however, could not be found in the satellite control database.
- SQL3932W The test synchronization session completed successfully. The satellite application version, however, is not set locally, or does not exist for this satellite's group at the satellite control server.
- SQL3933W The test synchronization session completed successfully. The release level of the satellite, however, is not supported by the release level of the satellite control server.
- SQL3934W The test synchronization session completed successfully. This satellite, however, is disabled at the satellite control server.
- SQL3935W The test synchronization session completed successfully. This satellite, however, is in the failed state at the satellite control server.
- SQL3955N The satellite control database name or its alias could not be found.

Synchronizing a Satellite

Figure 6 on page 118 shows the events that can occur when a satellite is synchronized, including the APIs that can be called from the synchronizing application. Specifically:

1. The db2SyncSatellite API is invoked, which starts the synchronization session. This API does not return until the session completes.
2. The db2QuerySatelliteProgress API is invoked, which returns information about the progress of the synchronization session. This API can be invoked more than once to return progress information. Invoke this API on another thread or process.
3. The db2SyncSatellite API returns when the synchronization session ends. The API can return normally, or the db2SyncSatelliteStop API can be invoked (on another thread or process) during a synchronization session to interrupt the session. When invoked, the db2SyncSatelliteStop API issues a

STOP request to the db2SyncSatellite API. The db2SyncSatellite API, however, does not return until it completes executing the current step of the synchronization process.

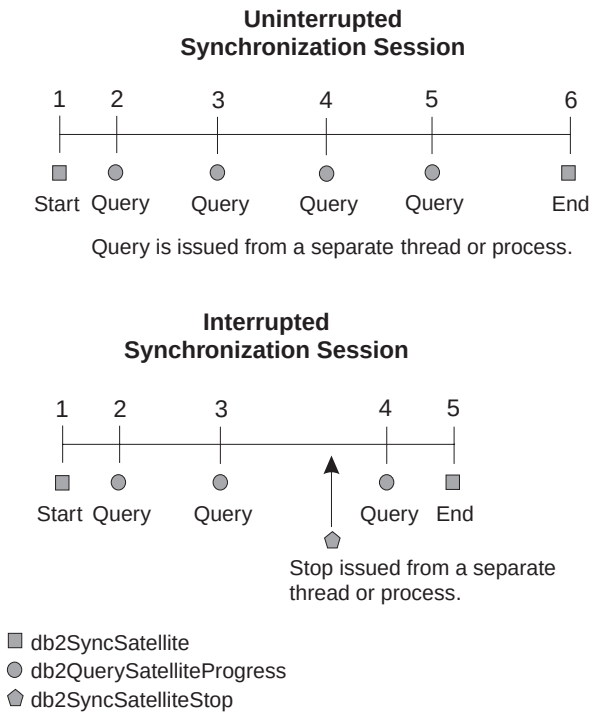


Figure 6. Synchronization Time Line

The sections that follow described the APIs that you can use to cause these events. For additional information about these APIs, and other satellite-related APIs that are not described in this book, refer to the *Administrative API Reference*.

Initiating a Synchronization Session: To initiate the synchronization session, use the db2SyncSatellite API:

```
SQL_API_RC SQL_API_FN                  /* Synchronize Satellite              */
  db2SyncSatellite (
   db2Uint32 versionNumber,            /* Database version number            */
   void * pParmStruct,                 /* Input/Output parameters             */
   struct sqlca * pSqlca);             /* SQLCA                              */
```

You must invoke the db2SyncSatellite API from the context of the satellite's instance (the same as a synchronization test). This API does not return until the synchronization session ends, successfully or otherwise.

Only one synchronization session can be active on the satellite at a time. If this API is invoked and a synchronization session is currently active on the satellite, the following SQLCODE is issued:

```
SQL3950N A synchronization session is active.
At most one synchronization session can be active.
```

When this API ends successfully, the following SQLCODE is issued:

```
SQL3910I Synchronization session completed successfully.
```

Querying the Progress of a Synchronization Session: Use the db2QuerySatelliteProgress API to return information about the progress of an active synchronization session:

```
SQL_API_RC SQL_API_FN                  /* Query Satellite Progress           */
  db2QuerySatelliteProgress (
   db2Uint32 versionNumber,            /* Database version number            */
   void * pParmStruct,                 /* Output parameters                   */
   struct sqlca * pSqlca);             /* SQLCA                              */
```

If the db2QuerySatelliteProgress API successfully returns the progress information, the following SQLCODE is issued:

```
SQL3918I Synchronization progress information was obtained successfully.
```

If a synchronization session is not active when the API is invoked, the following SQLCODE is issued:

```
SQL3936W No progress information is available.
```

You must supply the following output parameters for the pParmStruct parameter structure:

```
typedef struct db2QuerySatelliteProgressStruct
{
   db2int32        oStep;              /* current step of synchronization    */
   db2int32        oSubstep;           /* substep of the current step        */
   db2int32        oNumSubsteps;       /* total number of substeps           */
   db2int32        oScriptStep;        /* step of current script substep     */
   db2int32        oNumScriptSteps;    /* total number of script steps       */
   char            *poDescription;     /* description of step                */
   char            *poError;           /* error text, if available           */
   char            *poProgressLog;     /* contents of progress log           */
} db2QuerySatelliteProgressStruct;
```

Where:

oStep Output. The current phase of the synchronization session (defined in db2ApiDf.h). For example, `sync initiated` or `downloading batch steps`.

oSubstep
Output. If the synchronization step (oStep) can be broken down into substeps, this will be the current substep. Typically, the substep is the execution of a batch step.

oNumSubsteps
Output. If there exists a substep (oSubstep) for the current step of the synchronization session, this will be the total number of substeps (or batch steps) that comprise the synchronization session. Typically, oNumSubsteps is the total number of batch steps.

oScriptStep
Output. If the current substep is the execution of a batch step, this parameter reports on the progress of the script execution, if available. For example, if DB2 data replication is invoked as a batch step, the replication will report its own process.

oNumScriptSteps
Output. If a script step is reported, this parameter contains the total number of steps that comprise the script's execution.

poDescription
Output. A description of the state of the current phase of the satellite's synchronization session.

poError
Output. If the synchronization session is in error, a description of the error is passed by this parameter.

poProgressLog
Output. The entire log of the satellite's synchronization session is returned by this parameter.

Stopping a Synchronization Session: You can use the db2SyncSatelliteStop API to interrupt an active synchronization session:

```
SQL_API_RC SQL_API_FN                  /* Stop Satellite's Synchronization    */
                                       /* Session                             */

  db2SyncSatelliteStop (
   db2Uint32 versionNumber,            /* Database version number             */
   void * pParmStruct,                 /* Input/Output parameters              */
   struct sqlca * pSqlca);             /* SQLCA                               */
```

The db2SyncSatelliteStop API issues an asynchronous STOP request to the synchronization session. That is, the API returns immediately after it issues the STOP request, and not after the synchronization session stops. If a synchronization session is active and the STOP request is issued successfully, the following SQLCODE is issued:

```
SQL3912I STOP completed successfully.
```

If no synchronization session is active when the API is invoked, the following SQLCODE is issued:

```
SQL3913I STOP issued, but no synchronization session is currently active.
```

The db2SyncSatellite API only returns as a result of the interrupt when it has reached an appropriately *safe* point in the session. That is, a STOP request is not necessarily executed as soon as it is received. Instead, the db2SyncSatellite API finishes executing the current step in the synchronization process before exiting. For example, if the db2SyncSatelliteStop API is issued while the db2SyncSatellite API is executing a script, the db2SyncSatellite API completes executing the script before stopping. As another example, if the db2SyncSatellite API is interrupted before the satellite can upload the results of the synchronization session to the DB2 control server, the API issues the following SQLCODE:

```
SQL3917I A STOP request was received before the results were uploaded to the
satellite control server.
The results will be uploaded during the next synchronization session.
```

The next time db2SyncSatellite is invoked, the synchronization session resumes where it stopped.

Because the db2SyncSatellite API does not return until the synchronization session completes, and the db2SyncSatelliteStop is asynchronous, you must execute these two APIs on different threads or processes.

Building Applications for Satellites

The following sections provide detailed information about building applications for satellites that run Windows 32-bit operating systems.

Microsoft Visual C++

The Visual C++ compiler is used for both C and C++ sample programs supplied in the %DB2PATH%\samples\c and %DB2PATH%\samples\cpp directories. The batch files in both these directories contain commands to accept either a C or C++ source file, depending on the file extension. By default, the C++ commands are commented out in the batch files in %DB2PATH%\samples\c, and the C commands are commented out in the batch files in %DB2PATH%\samples\cpp.

DB2 API Applications

The batch file bldmsapi.bat, in %DB2PATH%\samples\c, and %DB2PATH%\samples\cpp, contains the commands to build a DB2 API program.

The parameter, %1, specifies the name of your source file.

```
@echo off
rem bldmsapi.bat file
rem Builds a DB2 API program in C or C++
rem using the Microsoft Visual C++ compiler.
rem Usage: bldmsapi prog_name
if "%1" == "" goto error
rem  Compile the C program.
cl -Z7 -Od -c -W2 -D_X86_=1 -DWIN32 %1.c util.c
rem For C++, comment out the above line, and uncomment the following:
rem cl -Z7 -Od -c -W2 -D_X86_=1 -DWIN32 %1.cxx util.cxx
rem Link the program.
link -debug:full -debugtype:cv -out:%1.exe %1.obj util.obj db2api.lib
goto exit
:error
echo Usage: bldmsapi prog_name
:exit
@echo on
```

Compile and Link Options for bldmsapi

The batch file contains the following compile options:

cl	The Microsoft Visual C++ compiler.
-Z7	C7 style CodeView information generated.
-Od	Disable optimizations. It is easier to use a debugger with optimization off.
-c	Perform compile only; no link. This book assumes that compile and link are separate steps.
-W2	Set warning level.
-D_X86_=1	Compiler option necessary for Windows 32-bit operating systems to run on Intel-based computers.
-DWIN32	Compiler option necessary for Windows 32-bit operating systems.

Compile and Link Options for bldmsapi
The batch file contains the following link options: **link** Use the 32-bit linker to link edit. **-debug:full** Include debugging information. **-debugtype:cv** Indicate the debugger type. **-out:%1.exe** Specify the executable. **%1.obj** Include the object file **db2api.lib** Link with the DB2 library. Refer to your compiler documentation for additional compiler options.

To build the sample program, client, from either the source file client.c, in %DB2PATH%\samples\c, or from the source file client.cxx, in %DB2PATH%\samples\cpp, enter:

```
bldmsapi client
```

The result is an executable file, client.exe. You can run the executable file by entering the executable name (without the extension) on the command line:

```
client
```

IBM VisualAge C++ Version 3.5

The VisualAge C++ compiler is used for both C and C++ sample programs supplied in the %DB2PATH%\samples\c and %DB2PATH%\samples\cpp directories. The batch files in both these directories contain commands to accept either a C or C++ source file, depending on the file extension. By default, the C++ commands are commented out in the batch files in %DB2PATH%\samples\c, and the C commands are commented out in the batch files in %DB2PATH%\samples\cpp. This section demonstrates building programs using the C batch files.

DB2 API Applications

The batch file bldvaapi.bat, in %DB2PATH%\samples\c, and in %DB2PATH%\samples\cpp, contains the commands to build a DB2 API program.

Ensure that the LIB environment variable points to %DB2PATH%\lib as follows:

```
set LIB=%DB2PATH%\lib;%LIB%
```

The parameter, %1, specifies the name of your source file.

```
@echo off
rem bldvaapi.bat file
rem Builds a DB2 API program using the IBM VisualAge C++ compiler.
rem USAGE: bldvaapi <prog_name>
rem  Compile the program.
icc -c -Ti -W1 %1.c util.c
rem For C++, comment out the above line and uncomment the following:
rem icc -c -Ti -W1 %1.c util.cxx
rem  Link the program.
ilink /MAP /DEBUG /ST:32000 /PM:VIO %1.obj util.obj db2api.lib
goto exit
:error
echo Usage: bldvaapi <prog_name>
:exit
@echo on
```

Compile and Link Options for bldvaapi

The batch file contains the following compile options:

icc The IBM VisualAge C++ compiler.

-c Perform compile only; no link. This book assumes that compile and link are separate steps.

-Ti Generate debugger information.

-W1 Output warning, error, and severe and unrecoverable error messages.

The batch file contains the following link options:

ilink Use the resource linker to link edit.

/MAP Generate a map file.

/DEBUG Include debugging information.

/ST:32000
Specify a stack size of at least 32 000.

/PM:VIO
Enable the program to run in a window or in a full screen.

%1.obj Include the object file.

util.obj
Include the error-checking utility object file.

db2api.lib
Link with the DB2 library.

Refer to your compiler documentation for additional compiler options.

To build the sample program client from the C source file client.c, or the C++ file client.cxx, enter:

```
bldvaapi client
```

The result is an executable file client.exe. You can run the executable file against the SATCTLDB database by entering the executable name (without the extension):

```
client
```

IBM VisualAge C++ Version 4.0

The VisualAge C++ compiler differs from other compilers on Windows 32-bit operating systems. To compile a program with VisualAge C++ Version 4.0, you must first make a configuration file. For more information, refer to the documentation that is provided with the compiler.

DB2 provides configuration files for the different types of DB2 programs you can build with this VisualAge C++ compiler. To use a DB2 configuration file, first set an environment variable to the program name that you want to compile. Then compile the program with a command supplied by VisualAge C++ Version 4.0. The api.icc file is the DB2 API configuration file. See "DB2 API Applications" for information on how to use this file to compile your programs. For information about the other DB2 configuration files that are available, refer to the *Application Building Guide*.

DB2 API Applications

The configuration file, api.icc, in %DB2PATH%\samples\c, allows you to build DB2 API programs in C. There is also a C++ DB2 API configuration file in %DB2PATH%\samples\cpp. These files can be used on AIX, OS/2 and Windows 32-bit operating systems.

```
// api.icc configuration file for DB2 API programs
// for VisualAge C++ Version 4.0
// To use on AIX, enter: 'export API=prog_name'
// To use on OS/2 and Windows, enter: 'set API=prog_name'
// Then compile the program by entering: 'vacbld api.icc'

if defined( $API )
{
  prog_name = $API
}
else
{
  error "Environment Variable API is not defined."
}

infile = prog_name".c"
util   = "util.c"
```

```
if defined( $__TOS_AIX__ )
{
  // Set db2path to where DB2 will be accessed.
  // The default is the standard instance path.
  db2path     = $HOME"/sqllib"
  outfile     = prog_name
  group lib   = "libdb2.a"
  option opts = link( libsearchpath, db2path"/lib" ),
                incl( searchPath, db2path"/include" )
}
else // if defined( $__TOS_OS2__ ) | defined( $__TOS_WIN__ )
{
  db2path     = $DB2PATH
  outfile     = prog_name".exe"
  group lib   = "db2api.lib"
  option opts = link( libsearchpath, db2path"\\lib" ),
                incl( searchPath, db2path"\\include" )
}

option opts
{
  target type(exe) outfile
  {
    source infile
    source util
    source lib
  }
}
```

VisualAge C++ Version 4.0 defines one of the following environment variables, depending on the operating system on which it is installed: __TOS_AIX__, __TOS_OS2__, __TOS_WIN__.

To use the configuration file to build the DB2 API sample program called client from the source file client.c, do the following:

1. Set the API environment variable to the program name by entering:

   ```
   set API=client
   ```

2. If you have an api.ics file in your working directory, produced by building a different program with the api.icc file, delete the api.ics file with this command:

   ```
   del api.ics
   ```

   An existing api.ics file produced for the same program you are going to build again does not have to be deleted.

3. Compile the sample program by entering:

   ```
   vacbld api.icc
   ```

   Note: The **vacbld** command is provided by VisualAge C++ Version 4.0.

The result is an executable file, client. You can run the program by entering the executable name:

```
client
```

The DB2SATELLITEID Registry Variable

The satellite ID is recorded on the satellite by the DB2SATELLITEID registry variable. The satellite ID must be set on the satellite before the satellite can synchronize. You can use the **db2set** command to set the value of the DB2SATELLITEID registry variable. You can also use the logon ID of the user of the satellite as the satellite ID. See "Setting the DB2SATELLITEID Registry Variable" on page 72 for information on setting the satellite ID. All the synchronization APIs that execute on the satellite obtain the value for the DB2SATELLITEID registry variable when they are executed.

Using the DB2 Synchronizer Application

The DB2 Synchronizer is a synchronizing application that is available with all DB2 Universal Database products that can be used as satellites. The DB2 Synchronizer is available in both test and synchronize modes.

You use the DB2 Synchronizer in test mode to verify the ability of a satellite to synchronize with its DB2 control server.

1. To open the DB2 Synchronizer window in test mode, issue the following command on the satellite:

   ```
   db2sync -t
   ```

2. To start the synchronization test, click on the **Test** push button.

   Notes:

   a. When a synchronization test is executed, the satellite does not download and execute batches.

   b. If the satadmin.aut file has been destroyed on a satellite, you can use the DB2 Synchronizer window to re-create it. For more information, see "Re-Creating or Updating the satadmin.aut File" on page 176.

For information about recovering from synchronization test errors, see "Synchronization Test Problems" on page 163.

If your end-user application does not call the synchronizing APIs or you have not written your own synchronizing application, your end users can also use the DB2 Synchronizer application to start a synchronization session.

1. To open the DB2 Synchronizer window in synchronize mode, issue the following command on the satellite:

   ```
   db2sync
   ```

2. To start the synchronization session, click on the **Start** push button.

If you performed the install interactively, the DB2 menu is available from the **Start** menu. Click **Start** and select **Programs -> DB2 for Windows NT -> DB2 Synchronizer**.

For information about recovering from synchronization errors, see "Synchronization Problems" on page 166.

Chapter 9. Recovering the Satellite Environment

Figure 7 on page 130 shows the different parts of the satellite solution. The sections that follow describe some of the considerations, as well as possible actions that you can take to ensure that you can, if required, restore any of these parts. When you have decided on a recovery strategy for the satellite environment, you should test it to determine whether it meets all of your requirements.

© Copyright IBM Corp. 1999

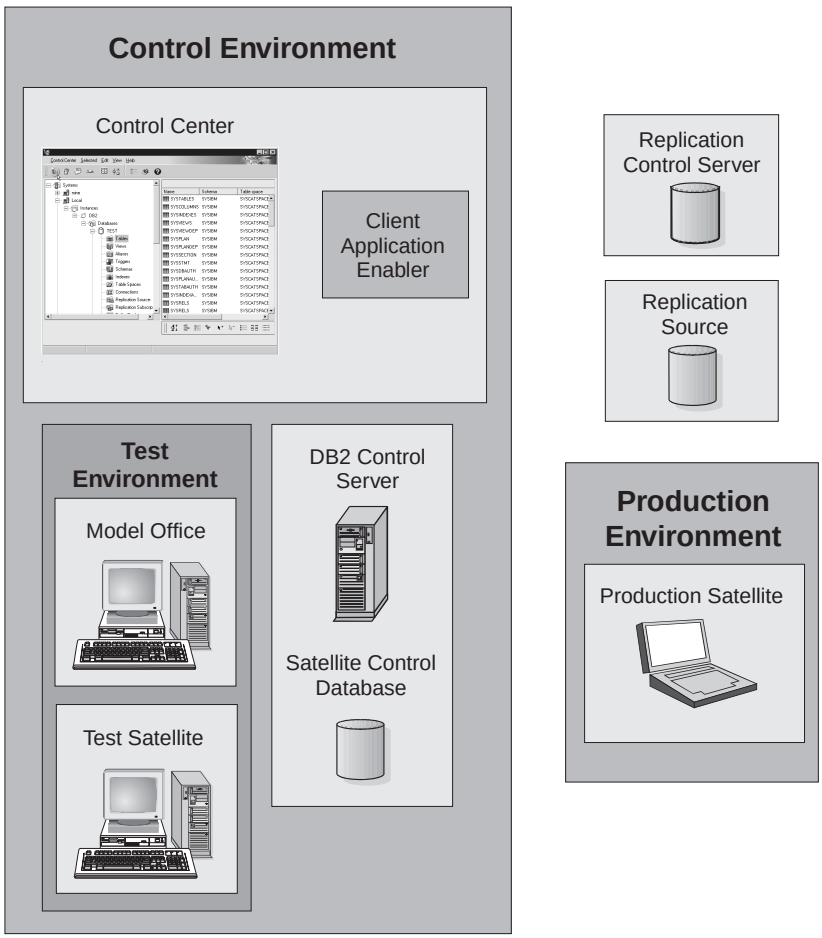


Figure 7. Recovery in the Satellite Environment

In the satellite environment, you require recovery strategies for the control environment, and for the production environment. The following sections describe the recovery considerations that apply in the satellite environment. For general information about recovery, refer to the *Administration Guide*.

Recovering Control Information

The control segment of the satellite environment includes the information that you use to both set up and maintain the entire satellite environment. To be able to fully recover the control segment, you require backup images that you can use to restore the DB2 instances and the DB2 databases:

Instance-level considerations
In the control segment, the Control Center server (or the JDBC server), the DB2 control server, and the model office each has its own DB2 instance. To facilitate the restoration of any of these instances, you need to back up the database manager configuration file, and the node, database, and DCS directory of each of these three instances. You can perform this task by using the **db2cfexp** command with the **backup** parameter to export the file and the directories to a file. If you need to restore the database manager configuration and the communications information, you can easily do this by first re-creating the instance, then using the **db2cfimp** command to import the file. For information about these commands, refer to the *Command Reference*.

Database-level considerations
The control segment contains two databases: the satellite control database (SATCTLDB), which is on the DB2 control server, and the database that you create on the model office. To be able to fully recover the satellite control database, you should enable it for forward recovery. For detail, see "Recovering the DB2 Control Server and Satellite Control Database" on page 132. In addition, you should also take a copy of the database configuration file `SQL`*`nnnnn`*`\SQLDBCON`.

Whether the database on the model office is recoverable depends on the recovery strategy that you want to implement on your satellites. For additional information, see "Recovering the Model Office" on page 133.

The sections that follow describe how to implement a recovery strategy for the Control Center, the DB2 control server and the satellite control database, and the model office.

Recovering the Control Center Directories

When you specify the DB2 servers and DB2 databases that you want to use as execution targets by adding them to the Control Center, information about these DB2 servers and databases (such as the name, alias, and the host on which they reside) is stored in the node, database, and DCS directories of the Control Center instance. For information about setting up execution targets, see "Chapter 4. Cataloging Instances and Databases" on page 51.

You can export the Control Center directories to a file with the **db2cfexp** command, using the **backup** parameter as follows:

```
db2cfexp filename history
```

You should export these directories to a file every time you add a new DB2 server or database to the Control Center, and save the resulting file to a location other than the Control Center instance (for example, to a diskette). For additional information about the **db2cfexp** command, refer to the *Command Reference*.

If you need to re-install the Control Center, you can use the file to import its directories, and obtain the set up on the Control Center that you had before the re-installation. To import the directories from the file, you can use the Import Client Profile window, which is available from the Client Configuration Assistant. For information about performing this task, refer to the online help that is available from the Client Configuration Assistant. You can also use the **db2cfimp** command to import the file, as follows:

```
db2cfimp filename
```

For additional information about the **db2cfimp** command, refer to the *Command Reference*.

Recovering the DB2 Control Server and Satellite Control Database

The availability of the DB2 control server and the satellite control database is critical both to administering the satellite environment, and to the ability of satellites to synchronize. If the DB2 control server or the satellite control database is not available, you cannot administer the satellite environment, nor can satellites synchronize.

To ensure the availability of the DB2 control server, you should consider setting up a high-availability solution for it. For information about performing this task, refer to the *Administration Guide*.

You should enable the satellite control database for forward recovery. To do this, you can either set the *logretain* database configuration parameter to `Recovery` or `On`, use a user exit to archive log files, or both. With forward recovery, you will be able to reconstruct the satellite control database to either a specific point in time, or to the last update to the database that is recorded in the logs. For a discussion of the issues and how to implement a solution, refer to the *Administration Guide*.

In addition, you should also keep a copy of the database configuration file `SQL`*nnnnn*`\SQLDBCON`. You should take a copy of this file every time that you change the value of one or more of the database configuration parameters. Changes to database configuration parameters are not recorded in the logs. If

you maintain a copy of the database configuration file, you can restore the file after rolling the database forward. In this way, you do not have to use the **update database configuration** command to update the database configuration after the roll-forward operation is complete. You will, however, have to stop and restart the database so that the configuration values specified in the configuration file are activated for the database.

To protect the satellite control database against media failure, you can use the **backup database** command to write the backup image of the database to a different physical device than the device on which the database resides. You can also use the *newlogpath* database configuration parameter to specify that the logs required for forward recovery are stored on a different physical device than the database. For information about this configuration parameter, refer to the *Administration Guide*.

In addition to storing backup images and logs on a different device than the database, you can further protect against media failure by using disk mirroring or RAID.

Recovering the Test Environment

The test environment typically contains one model office for each application version supported in the group, and the test satellites. The discussion that follows assumes that you are implementing a model office as part of your satellite environment, and that DB2 Satellite Edition is running on the model office.

Recovering the Model Office

When you use a test satellite as your model office, it is the model of all the group satellites, both test and production. You need to be able to recover the model office for two reasons:

- The model office is representative of the group satellites. If you want to know the average state of the group, the model office must be available.
- Because the model office is representative of the group satellites, you can use it to derive changes in the database definition that is use in the group. For you to be able to use the model office to derive changes in the database definition, the model office must be available.

An important decision to make when deciding on the configuration of the model office is the type of recovery that you want to use to restore the database on the model office. The recovery method that you decide to implement will apply to all the test and production satellites that are at the same application version as the model office.

Three types of database recovery are available for the model office. You can use refresh recovery, version recovery, or forward recovery to restore the database on the model office:

- With *refresh recovery*, you reinstall the model office, using the install image that you used for the mass deployment. Then, if you are using DB2 data replication, you cold start the Capture program to refresh the data in the database. Refresh recovery is the easiest method to use to recover a database.
- With *version recovery*, you can restore the database to the state it was in when the backup image was taken. Version recovery is more expensive than refresh recovery, as you must maintain backup images of the database. In addition, you must be able to identify which backup image you want to use to restore the database.
- With *forward recovery*, you restore the database using a backup image, then use the logs to apply changes that are subsequent to the backup image. Forward recovery is the most expensive method of recovering a database, as you must maintain backup images of the database and the database logs. In addition, forward recovery can be the most error-prone method of recovering a database.

For more information about version recovery and forward recovery, refer to the *Administration Guide*. Refresh recovery is unique to the satellite environment. The following description provides more information about the different types of database recovery that you can use:

Refresh recovery

With refresh recovery, you do not use DB2 utilities to either back up or restore the database on the model office. Instead, you use the satellite install image to re-install the entire model office, then have the model office re-execute all its group batches, from the first batch step of each batch. In effect, with refresh recovery, you rebuild the model office. The method that you use to have the model office re-execute its group batches depends on whether the model office is configured as a test satellite or as a production satellite:

- If the model office is a test satellite, use the Edit Satellite notebook to set the model office to re-execute its group batches from the first batch step.
- If the model office is a production satellite, use the Set Execution Starting Point window to set the model office to re-execute its group batches from the first batch step.

You can identify whether the model office is configured as a test or a production satellite from the satellite details view, which is available from the Satellite Administration Center. For information on using the

Edit Satellite notebook or the Set Execution Starting Point window, refer to the online help that is available from the Satellite Administration Center.

Refresh recovery may be sufficient if none of your production satellites will contain unique data. That is, all of the data on the satellite is a replica of corporate data. If you need to re-create the data on the model office, you can perform a cold start of the Capture program. For information on this task, refer to the description of how to start the Capture program in the Windows and OS/2 environments in the *Replication Guide and Reference*.

To use refresh recovery, set the *logretain* database configuration parameter to `Capture` (if you are using DB2 data replication) or to `No`. To set a database configuration parameter, you can use either the Configure Database notebook, which is available from the Control Center, or the **update database configuration** command. For information about this command, refer to the *Command Reference*.

Development phase considerations:

Before you complete the development of the model office, you will likely not have an install image that you can use for refresh recovery. Instead, you may need to rebuild the model office by performing such tasks as reinstalling the operating system and DB2, and re-creating the database definition. The information that follows provides some guidelines that may help simplify the process of rebuilding the model office.

If a problem occurs during the development phase, you can simplify setting up communication information on the model office by either:

- Using the **db2cfexp** command with the BACKUP parameter to export the node, database, and DCS directories of the model office. Then, after you re-install DB2, you can use the **db2cfimp** command to import the directories into the instance.
- Following the process described in "Cataloging Instances and Databases on the Model Office" on page 57 to recreate the node, database and DCS directories on the model office.

Consider keeping a copy of the database configuration file `SQL`*`nnnnn`*`\SQLDBCON` for the database on the model office. You can take a copy of this file every time that you change the value of one or more of the database configuration parameters. If you need to rebuild the model office, you can use the file to restore the database configuration without having to use the **update database**

configuration command. You have to stop and restart the database so that the configuration values specified in the configuration file are activated for the database.

If you can easily re-create the database and its data (for example, you have the SQL statements and DB2 commands in a file), you will likely not require a backup image of the database. If, however, this assumption is not true, consider backing up the database, and storing the backup image in a location other than the model office.

Version recovery

With version recovery, you use DB2 utilities to take an offline backup image of the database on the model office, which you can use to restore the database, as required. You have the option of taking a backup of the database every time you implement a change that results in the database being in a consistent state, or taking only one backup image of the database, then having the model office re-execute its group batches.

You take an offline backup of the database using the **backup database** command.

Note: For an offline backup to succeed, the backup database operation must be the only application that is connected to the database. To ensure that the backup operation is the only application that is connected to the database, you can use the **force application all** command or the **db2stop force** command before issuing the **backup database** command. For information about using these commands, refer to the *Command Reference*.

- If you want to have multiple backup images of the database, back it up when one of the following events occur. In either situation, the database is in a known and a consistent state:
  - You promote a batch to production to change the database definition or the data that supports the end-user application.
  - You successfully apply a fix batch to the model office.

  If you need to restore the database, you can restore it using the latest backup image that you have. Taking an offline backup each time that the database is in a consistent state as the result of a change to a batch or the successful application of a fix batch, however, can result in many backup images of the database.
- If you want to maintain only one backup copy of the database, take the backup image before the model office has executed any of its group batches (that is, the model office has not yet synchronized). If you need to restore the database, you can use this backup image, then have the model office re-execute all of its group batches, from

the first batch step of each batch. The method that you use to have the model office re-execute its group batches depends on whether the model office is configured as a test satellite or as a production satellite:

- If the model office is a test satellite, use the Edit Satellite notebook to set the model office to re-execute its group batches from the first batch step.
- If the model office is a production satellite, use the Set Execution Starting Point window to set the model office to re-execute its group batches from the first batch step.

You can identify whether the model office is configured as a test or a production satellite from the satellite details view, which is available from the Satellite Administration Center. For information on using the Edit Satellite notebook or the Set Execution Starting Point window, refer to the online help that is available from the Satellite Administration Center.

To use version recovery, set the *logretain* database configuration parameter to `Capture` (if you are using DB2 data replication) or to `No`. To set a database configuration parameter, you can use either the Configure Database notebook, which is available from the Control Center, or the **update database configuration** command. For information about this command, refer to the *Command Reference*.

If you are using version recovery, you should export the database manager configuration file and the node, database, and DCS directories on the model office to a file. This way, if you need to restore the configuration on the model office, you can restore it by importing the file, then restoring the database backup. You can export the directories and database manager configuration to a file using the **db2cfexp** command with the **backup** parameter. This command is described in the *Command Reference*.

With version recovery, you do not need to maintain a separate copy of the database configuration file `SQL`*nnnnn*`\SQLDBCON`. When you back up the database, the configuration file is automatically saved with the backup image. If you always back up the database after making a change to the database definition, you should always be able to restore the database and its configuration. If you only maintain the initial backup image of the database, the database configuration will be updated when the model office re-executes its group batches.

If you take a backup image every time the database is at a consistent state and you store the backup images on the model office, disk space

may be an issue. You should decide how many backups that you want to maintain, and consider using a fix batch to delete older backup images that you do not want to retain. Consider deleting the older backups only after the current backup completes; otherwise, if a backup operation fails, your recovery strategy may be compromised. You can use the **list backup** command to return the list of backups that are available. When you delete a backup image, remember to use the **prune history** command to remove the record for the backup from the history file.

If you want to protect the database against media failure on the model office, you should consider storing the backup image on a different physical disk than the disk on which the database resides.

If anything happens to the database on the model office, you can restore it to its last consistent state. For example, if the results of a group batch or a fix batch are not satisfactory, instead of applying a fix batch, you can restore the database from the previous backup image, and try the change again. When you obtain the results that you want, then take a backup of the database. To restore the database, you can use the Restore Database notebook or the Restore Database SmartGuide, both of which are available from the Control Center. You can also use the **restore database** command. For information about this command, refer to the *Command Reference*.

Note: After restoring the database, you may have to cold start the Capture program to refresh the replicated data on the model office. You may have to perform this task because the replication control server contains log information indicating that the model office has already successfully executed the Apply program, so the replication control server assumes that the data on the model office is up-to-date. For information on cold starting the Capture program, refer to the description of how to start the Capture program in the Windows and OS/2 environments in the *Replication Guide and Reference*.

Forward recovery

If you are using forward recovery, you can restore the database and apply the changes in the logs to roll the database forward either to a specific point in time, or to the end of the logs. In this way, you can re-create any configuration state of the model office database.

You should only consider using forward recovery if your production satellites will contain unique data in the database that supports the end-user application. That is, not all of the data in this database will be a replica of corporate data. Because you will not be able to re-create the data on a production satellite from corporate databases,

you should retain the database logs if it is important that this unique data can be recovered. The most likely situation in which you would use forward recovery is to recover the database from an error that cannot be undone by a fix batch, for example, a data replication error. In this situation, you would restore the database and roll it forward to the end of the logs to recover the unique data on the satellite.

To use forward recovery, set the *logretain* database configuration parameter to `Recovery`, regardless of whether you are using DB2 data replication. The value of `Recovery` retains the logs that are required for both forward recovery, and for DB2 data replication. To set a database configuration parameter, you can use either the Configure Database notebook, which is available from the Control Center, or the **update database configuration** command. For information about this command, refer to the *Command Reference*.

If you are using forward recovery, you should export the database manager configuration file and the node, database, and DCS directories on the model office to a file. This way, if you need to restore the configuration on the model office, you can restore it by importing the file, then restoring the database backup and rolling forward. You can export the directories and database manager configuration to a file using the **db2cfexp** command with the **backup** parameter. This command is described in the *Command Reference*.

With forward recovery, you may not need to maintain a copy of the database configuration file `SQL`*`nnnnn`*`\SQLDBCON`. Whenever you back up the database, the configuration file is saved with the backup image. If, however, the database configuration is changed and the database is not backed up when the change occurs, you should take a copy of the configuration file. After you roll the database forward, you can use the file to restore the database configuration without having to use the **update database configuration** command. You have to stop and restart the database so that the configuration values specified in the configuration file are activated for the database.

If you are storing the backup images and logs on the model office, disk space may be an issue. You should decide how many backups that you want to maintain, and consider using a batch step to delete the older backup images, as well as their associated logs. You can use the **list backup** command to return the list of backups that are available. This command also displays the name of the first log file that is associated with the backup. Consider deleting the older backups and logs only after the current backup completes; otherwise, if a backup operation fails, your recovery strategy may be compromised. When you delete a backup image, remember to use the **prune logfile** to delete the logs associated with the backup image, and

the **prune history** command to remove the record for the backup from the history file. For more information about the **prune** commands, refer to the *Command Reference*

If you want to protect the database against media failure on the model office, you should consider storing the backup image on a different physical disk than the disk on which the database resides. You should also store the logs on a different physical disk than that of the database. To perform this task, use the *newlogpath* database configuration parameter. For information about this parameter, refer to the *Administration Guide*.

Recovering Test Satellites

Because the test satellite can be installed from the image that you create from the model office, the recovery strategy that applies to the test satellite will be the same as that of the model office. That is, you will use refresh recovery, version recovery, or forward recovery to restore the database on the test satellite. For information about these types of recovery, see "Recovering the Model Office" on page 133.

Recovering Satellites in the Production Environment

Because the model office is the template for the production satellites, the recovery strategy that you decide on for the model office also applies to the production satellites. "Recovering the Model Office" on page 133 describes the considerations for using refresh, version, or forward recovery on the model office. The considerations for using these recovery strategies with production satellites are as follows:

Refresh recovery
With refresh recovery, you do not use database backups to recover the database on the production satellite. Instead, you use the install image that was generated from the model office to re-install the entire satellite, then use the Set Execution Starting Point window to specify that the satellite re-execute all of its group batches from the first batch step of each batch.

Version recovery
If you are using version recovery, you can back up the database on a periodic basis. In this situation, you will want to schedule how frequently the database is backed up. For example, you may want to take a backup image of the database on a weekly, or even a monthly basis. Or you may want the database to be backed up more frequently. One important factor to consider is the volume of transactions that are applied to the database. If the database is

frequently updated, you may want to back up the database more frequently. With version recovery, the backup schedule will determine how much you can protect against the loss of data.

Forward recovery

If you are using forward recovery, you will want to schedule how frequently the database is backed up. The more frequent the backup, the fewer the logs that will have to be applied during the roll-forward phase when the database is recovered. For example, if updates to the database are infrequent, you may want to take a backup image of the database on a weekly, or even a monthly basis. Because few log records will be written, it will not require much time to fully recover the database. If, however, the database is updated frequently, you may want to back up the database more frequently to reduce the number of logs, and the amount of time that will be required to fully restore the database.

Logging and DB2 Data Replication

DB2 always uses logs to record changes that are made to the database. Depending on the recovery strategy that you use, however, DB2 uses the logs in different ways. For a full discussion on how DB2 uses logs, refer to the *Administration Guide*. In the satellite environment, however, logs are also required if you want to use DB2 data replication. The description that follows describes how you should set the value of the *logretain* database configuration parameter.

If you are using DB2 data replication on the satellite and:

- You want to use forward recovery, set the *logretain* database configuration parameter to `Recovery` either at installation time, or later. If you want to set this configuration parameter:
  - At installation time, select the **The database will be recoverable** check box on the Create Database page when you install DB2 Satellite Edition.
  - After installation time, use the **update database configuration** command. For information about this command, refer to the *Command Reference*.

  When you set the *logretain* database configuration parameter to `Recovery`, the *backup_pending* database configuration parameter is activated. You must back up the database before it can be accessed. You back up the database using the **backup database** command. For information about this command, refer to the *Command Reference*.

When the *logretain* database configuration parameter is set to `Recovery`, the logs are available for forward recovery. In addition, the Capture program can write the updates recorded in the logs to the change data tables that are used in DB2 data replication.

When the *logretain* database configuration parameter is set to `Recovery`, consider where to prune the logs. You can use the **list backup** command to display both the complete list of database backup images, and the first log file associated with each backup image. When you delete a backup image, remember to use the **prune logfile** to delete the logs associated with the backup image, and the **prune history** command to remove the record for the backup from the history file. For more information about the **prune** commands, refer to the *Command Reference*.

- You do not want the database to be recoverable, set the *logretain* database configuration parameter to `Capture` either at installation time, or later.

  In this situation, the Capture program can write the updates recorded in the logs to the change data tables. When the Capture program completes, it calls the **prune logfile** command to delete log files when the Capture program completes. You should not set the *logretain* database configuration parameter to `Capture` if you want to perform forward recovery on the database.

If you are not using DB2 data replication on the satellite and:

- You set the *logretain* database configuration parameter to `Recovery`, consider where to prune the logs. You can use the **list backup** command to display both the complete list of database backup images, and the first log file associated with each backup image. When you delete a backup image, remember to use the **prune logfile** to delete the logs associated with the backup image, and the **prune history** command to remove the record for the backup from the history file. For more information about the **prune** commands, refer to the *Command Reference*.
- You do not want to enable forward recovery for the database, leave the *logretain* database configuration parameter set to `No`. In this situation, only crash recovery is supported.

Replication Control Servers and Sources

You will want to implement a recovery strategy both for your replication control server, and for any databases that are replication sources. The *replication control server* is the database location of the applicable subscription definitions and Apply program control tables. The sections that follow describe the implications if the replication control server is on the satellite. If the replication control server or the replication source is not on the satellite, refer to the documentation for the DB2 database product that serves either as

a replication control server or replication source for the recommendations and procedures for implementing a recovery strategy.

Replication Control Sever on the Satellite

If the replication control server is on the satellite, either the database that contains the user data contains the subscription definitions and the Apply program control tables, or you use a different database to contain the replication-related information.

For refresh, version, and point-in-time forward recovery, you must cold start the Capture program to refresh the replicated data on the satellite. The only situation in which you do not have to cold start the Capture program occurs when you roll forward to the end of the logs. In this situation, the database is completely restored. For information on cold starting the Capture program, refer to the description of how to start the Capture program in the Windows and OS/2 environments in the *Replication Guide and Reference*.

Note: If you roll forward to a point in time, it may not be possible to restore all the data that is unique to the satellite: log records that were written past the specified point in time may contain unique data.

Replication Control Server on a DB2 Server

The database that provides the replication control server function does not have to be a database on the satellite. Instead, the database can be on the DB2 control server, or on any other DB2 server. Your recovery strategy for this database should be forward recovery to the end of the logs. This method is the only way to restore all the replication-related data that indicates whether a satellite has replicated or not, and what data each satellite has replicated.

Neither forward recovery to a point in time, nor version recovery are recommended for the replication control server. With either method, you cannot re-create all the data that indicates whether a satellite has replicated, or what data it replicated. If you have to restore the database with point-in-time or version recovery, you should cold start the Capture program on all the satellites that use that replication control server to ensure that their data is up-to-date.

Chapter 10. Performing a Mass Deployment

When you finish your development phase, and have verified that your model office represents what you want to deploy in production, you are ready to begin a mass deployment. Facilities exist to ease the deployment of the DB2 Satellite Edition solution on hundreds and thousands of satellites. The techniques described here will help you achieve predictable results.

You can use two methods to achieve the mass deployment of DB2 Satellite Edition:

- Mass installation

  The mass installation method uses a traditional installation process to deploy DB2 Satellite Edition. Use this method if you cannot replace the contents of the entire hard disk on the satellite. This situation occurs:

  – If the equipment is owned by a user outside of your company. For example, an independent insurance agent would potentially have application software on his system from several insurance companies.
  – When other applications are already deployed on company-owned systems.

    For example, you are deploying a new application, the first one to use DB2 Satellite Edition, on systems that are already running other applications.

© Copyright IBM Corp. 1999

If you use the mass installation method, you must build your own customized installation application. Your application will have to invoke the DB2 installation application, and perform any additional customization that you require. You can use your installation application to install your application, in addition to DB2.

- Mass copy

  When the equipment is company owned and dedicated to a single application, the setup of the application and database often occurs at a central site. When the setup is complete, the system is given to the end user, either at an application training session, or by shipping it directly to the user. In this situation, the end user is most likely an employee of the company.

  If you perform the setup activities at a central site, this can bypass the traditional installation process for all but the first satellite. In this situation, you will find it more efficient to copy the disk image of a fully configured system and perform minor customization, rather than have to repeatedly install the operating system, database and application. After you copy the disk, you can customize the satellite by running scripts that tailor the system for a specific user, and load the users portion of the data.

Testing Your Deployment Method

If your deployment is large, it is critical that you test your deployment method rigorously before using it to create your entire production environment. Even after testing the method in the information systems test environment, consider deploying a set of pilot satellites before deploying a production environment.

Performing a Mass Installation

DB2 Satellite Edition, along with the other DB2 Universal Database Version 6 products, provides tools that you can use to perform a mass installation. These tools include:

- A response file generator that you can use to reverse engineer the DB2 Universal Database products and components installed on a system to produce a response file. You can use the response file to drive the installation of DB2 on additional systems.
- Client profile export and import tools that you can use to move catalog information, ODBC/CLI settings, database manager configuration values, and DB2 registry settings from one system to other systems.
- A program, **cpysetup.bat**, that you can use to copy the installation files for a specific language from the IBM distribution CD to your media.

The response file generator and the client profile exporter require a source system from which to extract the information that create their output files. You should use the model office as the source system for these utilities. See "Chapter 6. Working with the Model Office" on page 87 for information about how to use the model office for this purpose. More information on this topic is included in the sections that follow.

When you combine the output from these utilities with your custom installation application, you can create the media to support a mass installation. Figure 8 provides a representation of the mass installation model.

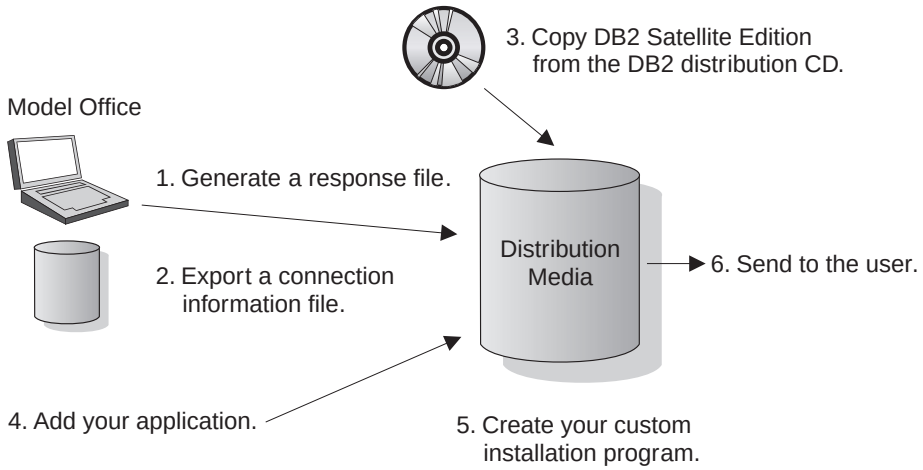


Figure 8. A Mass Deployment

Using the Model Office

As described in "Chapter 6. Working with the Model Office" on page 87, you can use the model office during the production-deployment phase as the source system for the utilities that can generate:

- Installed component information and other parameters that are required by the installation program
- Catalog information
- ODBC/CLI settings, database manager configuration values, and DB2 registry settings.

When the model office is set up and tested to ensure that it represents the production environment for a specific version of the end-user application, you can use the response file generator utility, **db2rspgn**, to generate a response

file that you can use to install your production satellites. In addition to generating a response file that can be used to perform installations, the response file generator calls the client profile export utility to generate a client profile file for each instance that contains:

- System-level values for DB2 registry settings
- Global CLI/ODBC settings
- Instance-level database manager configuration setting, DB2 registry settings, and catalog information for node and database directory entries
- CLI/ODBC settings for CLI/ODBC enabled databases.

Note: In most situations, only the DB2 instance will exist on the model office. One client profile will be generated for it.

The generated response file contains keywords that reference the generated client profile files. When the response file is used during a response-file driven installation, the installation program calls the client profile import utility so that the contents of the referenced client profile files can be imported to the new satellite to set it up. Using this method, you can replicate the setup of the model office to hundreds of satellites. For more information, see "Generating a Response File" on page 202.

Customizing the Generated Response File

The response file that the **db2rspgn** utility generates cannot extract all of the information that is required to install DB2 Satellite Edition from the model office. For example, the password for the user name that is used by the Remote Command Service cannot be extracted from Windows NT. So that you can easily specify this additional information, the generated file contains sample keywords. You must update some of these keywords before using the response file for the installation. Consider the following before setting these values in the response file:

- Satellite environment specific keywords:
  - The satellite ID must be unique on each satellite system. If the satellite ID is not set, the logon ID of the user of the satellite is used as the satellite ID. If you decide to set the satellite ID using a keyword in the response file, the satellite ID will be the same on all the satellites that are installed using the response file, unless you customize the response file for each satellite before it is used. See "Customizing the Operating Environment of Each Satellite" on page 151 for more information.
  - You must set the user ID and password for the Remote Command Service using the `ADMIN.USERID` and `ADMIN.PASSWORD` keywords.

  - You can set the user ID and password that the satellite uses to connect to the satellite control database for synchronization. See "Customizing the Operating Environment of Each Satellite" on page 151 for more information.
  - If you want the DB2 installation application to create a database, set the following keywords:
    - DB2_USERDB_NAME. Leave this keyword blank if you do not want to create a database at installation time.
    - DB2_USERDB_REP_SRC
    - DB2_USERDB_RECOVERY
  - If you set the application version on the satellite before running the **db2rspgn** utility, *do not* use the DB2.DB2SATELLITEAPPVER keyword to set the application version. The value specified by the keyword will be overwritten when the DB2 instance client profile is imported. You can set the application version on the satellite after the DB2 installation program completes. See "Customizing the Operating Environment of Each Satellite" on page 151 for more information.
- General DB2 installation keywords:
  - In most situations, you will want to set REBOOT = NO to prevent the DB2 installation program from rebooting the computer when the installation of DB2 Satellite Edition completes. If the computer is not rebooted, your installation application, which launched the DB2 installation program, will regain control when the DB2 installation completes. In this way, more customization of the satellite can be done. If you do not reboot the computer, be aware of the following:
    - All services added by the DB2 installation program will not be started.
    - All running processes will not have access to the updated path information that is created by the DB2 installation program. If you want to run any DB2 commands from your installation application after the DB2 installation completes, you must explicitly code the path information for all DB2 commands that you issue before the reboot.
  - Set CREATE_ICONS = NO if you do not want to create the DB2 for Windows folder in the Programs folder.
  - Set DB2.AUTOSTART = NO if you do not want the DB2 instance to be started when the computer is rebooted.
  - Do not add the following keywords to the generated response file:
    - DB2.PORT_NUMBER
    - DB2.SVCENAME

      Their values are overwritten when the DB2 instance client profile is imported.

For more information about these and other response file keywords, see "Response File Keywords" on page 200.

You may not be able to specify in the response file where on the computer you want DB2 to be installed. In this situation, your installation application can prompt the user for the location in which they want to install your application and DB2. With this information, your install application can edit the response file and replace the value of the FILE keyword with the drive and path in which DB2 is to be stored before the DB2 installation is performed. Similarly, you can change the values of other keywords before installing DB2.

If you are modifying the response file, you must first copy it to a disk drive on the target satellite before editing the file. If you copy the response file to a disk drive, copy the generated client profile files to the same drive and path. You must do this because the DB2 installation program only searches the directory location of the response file is for client profile files.

Preparing Your Distribution Media

When you prepare your distribution media, the layout of the files that will make up the various components of the DB2 installation image should be considered. For example, assume that you are preparing the files to install DB2 so that they can be used to burn a CD. Further assume that all the files required for the installation of DB2 Satellite Edition will be placed on the E drive of your system in a directory called \satellite. Into this directory, you will need to copy the generated and customized response file, the generated client profiles, and the files required to install DB2 Satellite Edition.

Copying DB2 Satellite Edition Installation Files to Your Media

The **cpysetup.bat** utility is provided on the DB2 Satellite Edition distribution CD. You use this utility to copy the files that are required to install DB2 Satellite Edition to your distribution media. See "Making DB2 Satellite Edition Files Available for Installation" on page 204 for information about the location of the utility on the distribution CD, and for information on how to use it.

Assume that your CD-ROM drive is the G drive. To copy the DB2 installation files to the e:\satellite directory, you would use the following command:

```
g:\db2\common\cpysetup.bat  e:\satellite   EN
```

The resulting file structure would be as follows:

```
e:\satellite\setup.exe
             \db2\common
```

Copying the Generated Response File and Client Profile Files to Your Media

When you copy the generated and customized response file and client profile files to your media, they must be in the same directory. If they are not, the DB2 installation program cannot locate the client profile files and import them.

For the example shown in "Invoking the DB2 Installation Program from Your Installation Application", it is assumed that you copied these files into a subdirectory of e:\satellite\response.

Note: If you want, the response file and the client profile files can be generated directly to the image of your distribution media.

Invoking the DB2 Installation Program from Your Installation Application

To invoke the DB2 Satellite Edition installation program from your application, follow the instructions in "Installing Using a Response File" on page 205. In the example that has been used, if the install image was built on the E: drive, the command to run the installation would be as follows (the command is over two lines for formatting reasons):

```
e:\satellite\setup  /u e:\satellite\response\db2udbse.rsp
  /l drive:\path\logfile
```

You must set the installation log file location appropriately for use on the target system.

If you require input from the user to further customize the response file, the response file and the generated client profile files must have been copied to the disk drive on the target system. In this situation, you must modify the **setup** command to point to the location of the updated response file.

Customizing the Operating Environment of Each Satellite

Each satellite has DB2 configuration values that must be set up so that the satellite can operate in the satellite administration environment. You can use your installation application to set or modify the configuration values:

- The satellite ID, user ID, and password required for authentication with the DB2 control server.

  The satellite ID must be unique on each satellite, but the ID does not have to be explicitly set. If the satellite ID is not set on the satellite, the user's logon ID is used to uniquely identify the satellite.

  The satellite ID is determined in the same way when the satadmin.aut file is created to contain the authentication credentials that are required to

connect to the satellite control database. If the satellite ID is not set, the user ID of the individual who installs DB2 is used to create the entry in the satadmin.aut file. If the satellite ID is subsequently set on the satellite, or the user who operates the synchronizing application is not the user who installed the satellite, the authentication credentials required for connecting to the satellite control database will not be found. You can use one of the following methods to fix this problem:

– Do not set the SATELLITE_ID keyword, or provide the user ID (DB2.SATCTLDB_USERNAME) and password (DB2.SATCTLDB_PASSWORD) that are required to connect to the satellite control database in the response file:
  - You can have your installation application set the SATELLITE_ID keyword by calling the **db2set** command after the installation of DB2 completes. For more information, see "Setting the DB2SATELLITEID Registry Variable" on page 72.
  - Before the user attempts to synchronize, instruct the user to run a synchronization test (**db2sync -t**). During the synchronization test, the user can specify the user ID and password that are required for synchronization. When this is done, the authentication credential (user ID and password) is stored under the SATELLITE_ID in the satadmin.aut file and is for subsequent synchronizations.
– Provide both the SATELLITE_ID, and the user ID and password required to connect to the satellite control database in the response file. Because the satellite ID must be unique on every satellite, you must customize the response file to set a unique value before the DB2 installation program is started. To perform this task, copy the response file to a disk drive on the satellite, and use a program to edit the file to set a unique value for DB2.SATELLITE_ID. If you decide to use this method, ensure that you:
  - Copy the client profile files into the same directory as the response file; otherwise, the DB2 installation program cannot find them
  - Modify or dynamically generate the command that invokes the DB2 setup program to set the drive and path where the modified response file is stored.

    Note: The satellite ID must be unique on each satellite. To prevent the ID from being overwritten when the client profile files are imported at the end of the installation process, the ID is not included in the exported client profile files.

• Satellite application version

  The value that is on the model office is exported when the client profile file is created. This value is imported when the client profile is imported. You can set the value again, either after DB2 is installed, or when the end-user application program is installed on the satellite. You can use either the **db2sync -s** command or the db2SetSyncSession API. For more information

about the **db2sync** command, see "Appendix C. db2sync - Start DB2 Synchronizer" on page 243. For more information about the db2SetSyncSession API, see "Setting the Application Version on the Satellite" on page 115.

Completing the Setup of the Satellite

The preceding sections described techniques that you can use to install DB2 Satellite Edition on a satellite, and customize DB2. In addition, you will need to install the application software and create the database (if you did not use the DB2 installation program to create it), and any database objects that are required to support the application (such as tables and indexes). You also need to bind the application to the database, unless it only uses CLI and ODBC commands to access data. You can accomplish these tasks as part of your custom installation program, or you can perform a second installation.

If you are using data replication to ensure that data on the satellite is synchronized with data in a corporate system, you can use data replication to load an initial copy of this data. See "Chapter 7. DB2 Data Replication" on page 95 for information about generalizing replication subscriptions from your model office, and modifying the generated subscription scripts so that each satellite receives its unique data when it synchronizes.

You must perform several other steps to fully enable the satellites that you are mass installing. See "Completing Your Mass Deployment" on page 155 for more information.

Mass Copy

Another technique that you can use to accomplish a mass deployment is to copy the entire hard drive on the model office, using a copy utility such as Ghost**. When the hard drive is copied, you will have to customize the resulting image before using it. For example, the satellite ID on the satellite will have to be set to a unique value, as will many operating system values, such as the TCP/IP host name.

For this technique to be suitable, the following must be true:

- The model office that you copy is operational. That is, it can synchronize and replicate data.
- The model office is already customized for the satellite administration environment.

To create the image, you can use the model office, or a copy of it.

After you copy the model office, you will have to perform customization to represent the unique environment that each user requires. The following sections outline the customization that you must perform on the DB2 portion of the satellite.

DB2 Considerations

If you perform a mass copy, you need to customize the following for DB2:

- General DB2 configuration:
  - Set the DB2SYSTEM registry variable to either the computer name or the TCP/IP host name of the system. DB2SYSTEM is a global DB2 registry variable, which you can set with the **db2set -g** command. Refer to the *Command Reference* for more information.
  - If the hardware you are deploying on and the model office system from which you generated the copy have significantly different CPU speeds, set the *cpuspeed* database manager configuration parameter to -1 . Refer to the *Administration Guide* for more information.
  - You can modify other DB2 registry values and database manager configuration values as required.
- Satellite specific configuration:
  - If you do not want to use the logon ID of the user of the satellite as the satellite ID, the DB2SATELLITEID registry variable must be set to a unique value. When you do this, the authentication credentials in the satadmin.aut file that are required to connect to the satellite control database will no longer be valid. Before the user attempts to synchronize, instruct the user to run a synchronization test (**db2sync -t**). During the synchronization test, the user can specify the user ID and password that are required for synchronization.
  - If the model office that is the source of the copied image already has your business application installed on it, the application version that is required for synchronization is most likely set. If, however, the application version is not set, you can specify the value by using either the **db2sync -s** command or the db2SetSyncSession API. For more information about the **db2sync** command, see "Appendix C. db2sync - Start DB2 Synchronizer" on page 243. For more information about the db2SetSyncSession API, see "Setting the Application Version on the Satellite" on page 115.

Application Data

When you copy the hard drive of the model office, you not only copy the database and the definitions for tables, indexes, and other database objects, you also copy the data. Most likely, this data will not be the right subset of the corporate data for the satellite. You will have to delete and replace the data. If you are using data replication, you can use it to populate the satellite

with its data after deleting the table data that was in the copy. If you are not using data replication, you can populate the tables on the satellite with the load or import utilities.

Data Replication

If you are using data replication to keep data on the satellite synchronized with data in a corporate system, you will need to set up the replication subscription for this satellite and load its initial data. You should:

- To set up replication subscriptions, see "Chapter 7. DB2 Data Replication" on page 95 for information about generalizing replication subscriptions from your model office, and modifying the generated subscription scripts so that each satellite receives its unique data when it synchronizes.
- Run the **dbcntl.sat** command file to drop and re-create the capture control tables.
- Run the synchronization process on the satellite to set up its replication subscriptions, and load its initial data.

Operating System Considerations

You will have to customize unique operating system values (such as the computer name) and communications information (such as the TCP/IP host name) for each satellite. The techniques that you use to accomplish the customization, and the list of characteristics that must be customized are beyond the scope of this manual.

You must perform several other steps to fully enable the satellites that you are mass installing. See "Completing Your Mass Deployment" for more information.

Completing Your Mass Deployment

Before your deployed satellites can synchronize, you must define and set them up in the DB2 control server. Assuming that you have been using the model office and test satellites to prepare for the mass deployment, the group, application version, and group batches for the application version already exist. You only need to add the deployed satellites to the group and promote the batches to production

Locate the group for which the mass deployment is being performed, and using the Create Satellite notebook, add the satellites. You must also enable the satellites to execute group batches. If your deployment is very large, using the Satellite Administration Center interface may be very time consuming. A

sample SQL script, which you can use to perform a mass insert of the required entries into the satellite control database is available on the Web at the following site:

`www.software.ibm.com/data/db2/library/satellite`

You should also ensure that the group batches that the deployed production satellites will execute when they synchronize are promoted to production. For information on performing this task, see "Promoting a Test Level to a Production Level" on page 29.

Installing a New Version of the End-User Application

During the life of your business application, you may find it necessary to upgrade the end-user application to address changing business needs. The upgrade of the end-user application to a new version can be performed either:

- When you install DB2. The installation can be either the initial installation of DB2, or an upgrade to a new release or version.
- As an application-only install, without any change to the level of DB2 on the satellite.

To install the end-user application at the same time you install DB2, follow the instructions described in either "Performing a Mass Installation" on page 146 or "Mass Copy" on page 153.

The sections that follow describe how to install a new version of the end-user application without re-installing DB2.

Installing a New Version of an Application

When the new version of the application is available for deployment to your satellites, you must perform several tasks before deploying the new version.

Setting the New Application Version on the Satellite

A new version of an end-user application often requires changes to the existing database definition. In addition to installing the new application code, you can use the installation application to perform any database definition migration and data migration that you require to support the new application. An alternative that you can also use is to drop the existing database, create a new database, and reload the required data. If the data for the initial deployment was loaded from information stored in a corporate database, you can use replication to reload the data. Use the same technique to load the data on the satellite as you used for the initial deployment.

When the new version of the application is installed on the satellite, you will not want it to execute the group batches of the old application version. If the satellite executes the batches for the previous application version, errors will likely occur. When your installation application installs the new version of the end-user application on the satellite, it should also update the application version that is used by the satellite for the purpose of synchronizing. You can specify the new application version by using either the **db2sync -s** command or the db2SetSyncSession API. For more information about the **db2sync** command, see "Appendix C. db2sync - Start DB2 Synchronizer" on page 243. For more information about the db2SetSyncSession API, see "Setting the Application Version on the Satellite" on page 115.

If the update of application version is not performed before the satellite next synchronizes, the satellite will upload its old application version to the DB2 control server, and any unexecuted batch steps for this application version will be sent to the satellite and executed. If any script in the group batches is incompatible with the new database definition, the results of the execution are unpredictable. For this reason, you must ensure that the application version is updated on the satellite *before* the end user initiates a synchronization session. Using the methods described above, reset the application version when you install the new version of the application.

Create the New Application Version for the Group

You must set up information about the group's new application version on the DB2 control server. Use the Satellite Administration Center to:

- Create a new application version for the group. The application version that you create using the Satellite Administration Center must match the new version of the end-user application.
- Create the setup, update and cleanup batches that you require for the new application version. The satellites will execute these batches the first time that they synchronize after the new version of the application is installed. Initially, these batches should be at the test level (that is, you should not promote them until you are sure that they produce the results that you want).

For the steps that you use to create the application version, see "Creating an Application Version" on page 78.

Create a Test System to Test the Deployment of the New Application

Before you deploy the new version of the application, we recommend that you create a test system that is at the same level as the existing production satellites. You can use this system to verify that the installation of the new version of the application provides the correct results, and that the setup,

update and cleanup batches produce the expected database definition and data for the end-user application. To create a test system, you can:

- Use the installation process that you used to create the production satellites of the previous application version to install the test system. The installation will set up the system to the level that was first deployed, that is, before the production satellites executed any of their group batches.
- Add the test system to the group by using the Satellite Administration Center. You should name this test system using the same conventions that you are using for your model offices. When you accomplish a clean test, you will most likely want to use this system as the model office for the new application version within the group.
- Have the test system synchronize so that it runs any batches that are already in production for the previous version of the application:
  - Use the Satellite Administration Center to set this satellite as a production satellite.
  - Run the **db2sync** application on the satellite. The satellite will download and execute all batches and batch steps that are in production for the previous application version.
  - Check the results of the synchronization to ensure that the satellite is not in the failed state, and that the batches were executed as expected.
  - If the results are as expected, use the Satellite Administration Center to set the system as a test satellite.

  This process will bring the test system up to the level of the production systems that are running the previous application version.
- Install the new version of the end-user application, using the process and media that you will use to upgrade your production satellites. You should use the installation process to accomplish any database migration and data migration steps, as described above.
- Ensure that the installation program performs its task correctly:
  - If the database definition changed for the new application version, or data was migrated, verify that the results are as expected
  - Verify that the new application version is set on the satellite. You can use the **db2sync -g** command to view the application version.
- Have the new model office synchronize, so that it runs the batches that you created for the new application version.
- Check the results:
  - You should answer the following questions:
    - Did the batches run successfully?
    - Is the satellite in the failed state?
    - Are there any unexpected results in the batch output logs?

- If there are no errors, is the resulting database definition and data correct?

– Correct any problems that occur, and repeat the synchronization test

See “Testing Group Batches” on page 80 for information about performing these tasks.

Repeat the process of synchronizing, checking results, and debugging the group batches until the satellite can synchronize without errors. The process should be able to set up the test satellite correctly before you deploy your production satellites.

When you are ready to begin deploying the production satellites, promote the setup, update and cleanup batches to the production level. For more information on performing this task, see “Promoting a Group Batch to Production” on page 84.

When you use the procedure described above and can build a test system that runs without errors, you should retain it, and use it as your model office for the new application version. The system is already upgraded with the new version of the end-user application, and has already executed the group batches that you promoted to production.

Deploy the New Application Version to the Production Satellites

Install the new application version on the production satellites, using the process and media that you tested with the test system. Using the same process should set the new application version on each satellite.

The next time that the production satellite synchronizes, it will upload its new application version to the DB2 control server, and download the batches for this new application version. The satellite will execute the batches, and upload the results of the synchronization to the DB2 control server.

Monitoring Which Satellites Implemented the New Application Version

You can use the satellite details view in the Satellite Administration Center to determine which application version a satellite is running. You can filter on the **Application version** column to determine which satellites have installed the new version, and which satellites are not yet upgraded.

Chapter 11. Problem Determination

The sections that follow describe some of the problems that can occur in the satellite environment. Topics that are covered include installation errors, and the configuration errors that can prevent a satellite from synchronizing. In addition, an overview is provided of how to fix a satellite that reports an error during a synchronization session.

Installation Problems

During the installation of either the DB2 control server or of DB2 Satellite Edition on a satellite, errors can occur. Information that you can use to diagnose problems is logged in several locations. The sections that follow provide more detailed information.

© Copyright IBM Corp. 1999

DB2 Control Server Installation

You install the DB2 control server by selecting the Control Server component of DB2 Universal Database Enterprise Edition. Installation-related messages are logged as follows:

- On Windows NT:
  - If you perform an interactive installation or a response file installation without redirecting the installation messages to a file using the /L option, the installation messages are written to the db2.log file as follows:

    ```
    x:\db2log\db2.log
    ```

    Where *x*: is the drive from which Windows NT is booted (the boot drive).
  - If you perform a response file installation using the /L option, the installation messages are written to the path and file that you specify.

  You can use any editor to examine the file that is created during the installation process.

  When the installation program creates the satellite control database, SATCTLDB, messages from this operation are logged in the *x*:\sqllib\misc\satctldb.log file, where *x*: is the drive on which you installed DB2.
- On AIX, the installation messages are written to the /temp/db2setup.log file. You can examine this file using any editor. On AIX, you create the satellite control database, SATCTLDB, after installing the DB2 control server. The messages for the create database operation are written to the log file that you specify.

Satellite Installation

Installation messages for DB2 Satellite Edition are logged as follows on Windows NT, Windows 95, and Windows 98:

- If you perform an interactive installation or a response file installation without redirecting the installation messages to a file using the /L option, the installation messages are written to the db2.log file as follows:

  ```
  x:\db2log\db2.log
  ```

  Where *x*: is the drive from which Windows is booted (the boot drive).
- If you perform a response file install using the /L option, install messages are written to the path and file you specify.

You can use any editor to examine the file that is created during the installation process.

If you create a database during installation, messages from the create database operation are logged in the *x*:\sqllib\misc\satdb.log file, where *x*: is the drive on which you installed DB2.

If you perform a response file installation, and the response file imports client profile files to configure the database manager instance, error messages from the import process are written to the db2diag.log for each instance.

Synchronization Configuration Problems

The sections that follow describe the configuration problems that can prevent either a test of the synchronization capability, or synchronization, from occurring. In general, the problems are caused by errors in the satellite configuration, information that is either missing or incorrect in the satellite control database, and authentication errors.

Synchronization Test Problems

You should use the **db2sync -t** command to open the DB2 Synchronizer application in test mode. You use the test mode to verify that the information that is required for the satellite to synchronize itself is correct both on the satellite, and in the satellite control database. The sections that follow describe the errors that can occur, and how to recover from them. For more information about the **db2sync** command, see "Appendix C. db2sync - Start DB2 Synchronizer" on page 243.

Satellite ID Is Not Set on Satellite

Typically, the satellite ID is the same as the logon ID for the user of the satellite. If this is not the case, you can set the value of the DB2SATELLITEID registry variable on the satellite to define the satellite ID. In either situation, the satellite ID on the satellite must be equal to the value specified for the satellite ID when it is created in the Satellite Administration Center using the Create Satellite notebook.

When the user initiates a synchronization session, the satellite ID is determined as follows:

1. If a value is specified for the DB2SATELLITEID registry variable, the satellite ID is this value.
2. Otherwise, the logon ID is used as the satellite ID.

If the satellite ID cannot be determined when the user initiates a synchronization session, the SQLCODE -3951N is returned. This error can occur if the DB2SATELLITEID registry variable is not set *and* the user is not logged on.

Depending on which method you use to specify the satellite ID, correct the problem as follows.

- If you use the logon ID as the satellite ID, ensure that the user logs on using the user ID that is defined in the satellite control database as the satellite ID.

  Either the user logged on to a user ID that is not the satellite ID that is stored in the satellite control database, or, on Windows 95 or Windows 98, the user cancelled the logon screen.
- If you use the DB2SATELLITEID registry variable to record the satellite ID, set the registry variable locally on the satellite as follows:

  ```
  db2set DB2SATELLITEID=satellite_id
  ```

  Note: The value specified for *satellite_id* must be equal to the value that is recorded as the satellite ID in the satellite control database. You can use the Satellite Administration Center to view the satellite ID from either the satellite details view, or the Edit Satellite notebook. For additional information about the satellite details view and the Edit Satellite notebook, refer to the online help that is available from the Satellite Administration Center.

Note: The satellite ID is case sensitive.

Satellite Control Database Not Cataloged on the Satellite

Before the synchronization test can occur, the satellite control database must be recorded in the database directory on the satellite. If the information about the satellite control database is either not available on the satellite, or is not correct, the SQLCODE -3955 is returned. In this situation, the Catalog Control Database window opens, in which you can specify the DB2 instance that contains the satellite control database. To specify the information, you can either use Discovery or type the TCP/IP host name and port number. For information about using the Catalog Control Database window, refer to the online help that is available from the window.

Authentication Credentials Are Not Available Or Not Correct on the Satellite

Before the synchronization test can occur, valid authentication credentials for the satellite control database must exist on the satellite and be stored in the *instance_path*\security\satadmin.aut file. If a synchronization test is initiated and either the satadmin.aut file does not exist, or it does not contain the user

ID and password required to connect to the satellite control database, the SQLCODE -3966 is returned with a reason code of 1. In this situation, the Connect to Control Database window opens, which you use to specify the authentication credentials to use to connect to the satellite control database. For information about using the Connect to Control Database window, refer to the online help that is available from the window.

Satellite Does Not Exist at the Satellite Control Database

During a synchronization test, the satellite uploads its unique satellite ID to the DB2 control server. The DB2 control server then checks that this value is recorded in the satellite control database. If the satellite ID is not recorded in the satellite control database, the SQLCODE -3931W is returned. The synchronization test ends.

Note: The satellite ID is displayed in the title bar of the DB2 Synchronizer application.

If this error occurs:

- The satellite ID that the satellite uploads to the DB2 control server is not correct.

  In this situation, use the **db2set** command to change the value of the DB2SATELLITEID registry variable, or ensure that the user logs on to the user ID that corresponds to the satellite ID that is recorded in the satellite control database.
- The satellite ID is not recorded correctly in the satellite control database.

  In this situation, you must drop the satellite and create it again. For information on performing these tasks, refer to the online help that is available from the Satellite Administration Center.
- The satellite is not yet created.

  If the satellite is not yet created, use the Create Satellite notebook to create it. For information about creating a satellite, refer to the online help that is available from the Satellite Administration Center.

Note: The satellite ID is case sensitive.

Incompatible Versions of DB2 on the Satellite and the DB2 Control Server

For synchronization to occur, the release level of DB2 on the DB2 control server and the release level of DB2 on the satellite must be compatible. The release level of DB2 on the satellite must be within the range of one level above to two levels below that of the DB2 on the DB2 control server. If the release levels are not compatible, the SQLCODE -3933W is returned.

If this error occurs, migrate the release level of DB2 that is on the satellite so that it is compatible with the release level of DB2 on the DB2 control server. For information on how to identify the release level of DB2 on the satellite, see "The Satellite Software Version" on page 175.

Synchronization Problems

The sections that follow describe problems that can occur during a synchronization session.

Satellite Cannot Synchronize More Than Once

The situation can occur that, after encountering an error during a synchronization session and subsequently being fixed, the satellite cannot synchronize again. What can occur is as follows. The satellite synchronized, successfully downloading its group batches, but encountered an error when executing these batches. After being fixed, the satellite attempted to synchronize again, and received the SQLCODE -3950, indicating that a synchronization session is already active. This SQLCODE is issued because the DB2 control server records the satellite as having downloaded its group batches, and is expecting the satellite to upload the results of the previous synchronization session. The fix to the satellite, however, has set the state on the satellite so that the satellite is again attempting to download its group batches.

To fix this problem, you must change the status of the satellite in the satellite control database to enable it to download the group batches:

1. Connect to the SATCTLDB database on the DB2 control server.
2. Issue the following SQL statement, where *satellite_id* is the satellite ID recorded in the SATADMIN.SATELLITES table:

```
UPDATE SATADMIN.SATELLITES SET sync_state='N' WHERE id='satellite_id'
```

Satellite is in the Failed State at the DB2 Control Server

If the user initiates a synchronization session and the SQLCODE -3935W is returned, the satellite is in the failed state. That is, the satellite reported an error to the DB2 control server during a previous synchronization session. When a satellite reports an error, it is automatically disabled from executing group batches. In this situation, you could create a fix batch to correct the problem on the satellite. For information about fix batches, see "Fix Batches" on page 43. "Identifying and Fixing a Failed Satellite" on page 169 describes how to identify and fix a satellite that is in the failed state.

Satellite Is Not Enabled at the DB2 Control Server

When a satellite is first created, it is disabled. The disablement allows, for example, the staging of the deployment of the group satellites.

Before a satellite can execute the group batches for its particular application version, it must be enabled to execute the group batches. If the user initiates a synchronization session and the satellite is not enabled to execute its group batches, the SQLCODE -3934W is issued.

The satellite may be disabled for the following reasons:

- The satellite is in the failed state. You must fix the satellite before enabling it to synchronize. See "Identifying and Fixing a Failed Satellite" on page 169 for more information.
- The satellite is disabled for administrative reasons.
- The satellite is newly created.

If the SQLCODE -3934W warning is issued and you want this satellite to begin synchronizing, use the Satellite Administration Center to enable the satellite at the DB2 control server. For information on how to perform this task, refer to the online help that is available from the Satellite Administration Center.

Satellite's Application Version Does Not Exist at the DB2 Control Server

When a satellite synchronizes, it passes its application version to the DB2 control server. The DB2 control server uses this information to determine which group batches and batch steps the satellite is to execute. If the DB2 control server cannot find this application version for the satellite's group in the satellite control database, the SQLCODE -3932W is returned. The synchronization session ends.

If this warning occurs, one of the following is possible:

- The application version set on the satellite is not correct.

  You can use the **db2sync -s** *application_version* command or the db2SetSyncSession API to set the application version on the satellite. To display the current value of the application version on the satellite, use the **db2sync -g** command. For more information about the **db2sync** command, see "Appendix C. db2sync - Start DB2 Synchronizer" on page 243. For more information about the db2SetSyncSession API, refer to the *Administrative API Reference*.
- The application version for the satellite's group is not yet created at the DB2 control server.

  In this situation, the satellite may have been modified to use the next version of the end-user application, but the application version is not yet

created for the satellite's group in the satellite control database. Use the Create Application Version window to create the application version for the group. For information about this window, refer to the online help that is available from the Satellite Administration Center.

Note: The application version is case sensitive.

Application Version Is Not Set at the Satellite

When a satellite synchronizes, it passes its application version to the DB2 control server. The DB2 control server uses this information to determine which group batches and batch steps the satellite is to execute. If the user initiates a synchronization session and the application version is not recorded locally on the satellite, the SQLCODE -3956N is returned. The synchronization session ends.

You can use the **db2sync -s** *application_version* command or the db2SetSyncSession API to set the application version on the satellite. For more information about the **db2sync** command, see "Appendix C. db2sync - Start DB2 Synchronizer" on page 243. For more information about the db2SetSyncSession API, refer to the *Administrative API Reference*.

Note: The application version is case sensitive.

Authentication Error Occurs at Satellite

If a satellite has not synchronized for some time and the password to the DB2 control server changed before the satellite had a chance to download the changes during a synchronization session, the SQLCODE -1403 may be issued when the user initiates a synchronization session. This SQLCODE indicates an authentication error. In this situation, the satellite does not have the correct password for the DB2 control server in its *instance_path*\security\satadmin.aut file. See "Re-Creating or Updating the satadmin.aut File" on page 176 for information on how to update this file.

An authentication error can also occur if the satellite ID is changed after the satellite has the correct authentication information in its satadmin.aut file. In this situation, the SQLCODE -3966 is issued with a reason code of 1. To recover from this error, you can:

- Use the **db2set** command to set the value of the DB2SATELLITEID registry variable back to its previous value.
- Use the **db2sync -t** command to start the DB2 Synchronizer application in test mode on the satellite. In this situation, the Connect to Control Database window opens, which you use to specify the authentication credentials to

use to connect to the satellite control database. For information about using the Connect to Control Database window, refer to the online help that is available from the window.

Note: For the **db2sync -t** solution to work, you must have already created a satellite using the Satellite Administration Center, and the ID of the new satellite must match the new value specified for the DB2SATELLITEID registry variable on the satellite.

Identifying and Fixing a Failed Satellite

The sections that follow describe how to use the Satellite Administration Center to identify and fix problems on failed satellites. For information about how to use the windows and notebooks of the Satellite Administration Center, and for information about the different views and icons that are available, refer to the online help that is available from the Satellite Administration Center.

The following sections provide information on:

1. "Identifying the Failed Satellite"
2. "Obtaining Information About the Failure" on page 171
3. "Assigning a Fix Batch to the Satellite" on page 172
4. "Debugging the Fix Batch" on page 173
5. "Returning the Repaired Satellite to Production" on page 174.

Identifying the Failed Satellite

The Satellite Administration Center uses roll-up views, which provides easy access to a quick, high-level snapshot of a group, or of an application version. The roll-up views display information about any failures that occur. That is, if a satellite reports a failure when it uploads its status during a synchronization session, you do not need to have the satellite details open to find out that a failure occurred. Instead, the icons that are displayed in the object tree (the left-hand side of the Satellite Administration Center) indicate if an error has occurred.

If you want to view basic information about the icons that are used in the Satellite Administration Center, use the Show/Hide Legend icon on the tool bar to open the Legend window. For information about the different possible states of these icons, refer to the legend, which is available from the Satellite Administration Center help.

The following example describes how to identify a production satellite that reports an error, and what the error is. To find the failed production satellite, and the error that it reported:

1. Expand the Groups folder in the object tree.

   The Group folder for the group that contains the failed satellite will have a red ″X″ superimposed over it to indicate that at least one production satellite reported an error while executing the group batches of a specific application version.

2. Select the Group folder.

   So that you can easily identify when a production satellite reports an error, the icons that represent both the Satellites folder and the Application Version folder associated with the group contain red within the folder.

   Note: The Application Version folder associated with the group only has red within the folder when one or more production satellites report an error while executing a group batch. The Satellites icon associated with the group has red within the folder if any satellite, either test or production, reports an error.

3. Within the Group folder:

   a. Click the Application Version folder to determine which application version was being executed by the production satellite (or satellites) that reported an error. The application version details view opens in the contents (the right-hand side of the display) pane of the Satellite Administration Center. For information about this view, refer to the online help that is available from the Satellite Administration Center.

      The icon representing the application version that was being executed by a production satellite that reported an error has a red ″X″ superimposed over it as follows:

   b. Click the Satellites folder to determine which production satellite (or satellites) reported an error. The satellite details view opens. For information about this view, refer to the online help that is available from the Satellite Administration Center.

      The icon the represents the production satellite that reported the error is as follows:

The X over the icon indicates that the production satellite has failed when executing group batches. The gradation on the sphere in the icon indicates that the satellite is disabled.

Note: If a test satellite reports an error, it is also disabled from executing group batches.

Obtaining Information About the Failure

When you identify the failed satellite, the next step is to obtain information about the failure that occurred. To perform this task, you view the logs for the satellite:

1. Right click on the failed satellite.
2. Select **Show Logs** from the pop-up menu.

   The Show Logs window opens. The logs in the window are organized by date in descending order.
3. Select the log that records the failure and right click. Because the logs are organized by date in descending order, typically, the log that records the failure is the first log in the list.
4. Select **View Details** from the pop-up menu.

   The Log Details window opens. This window displays the full set of information for the log that you selected. The information includes the batch that was being executed, as well as the batch step and the script that did not execute successfully. The log information also indicates whether the error resulted in an external or an internal return code. For more information, see "Internal and External Error Return Codes" on page 175.

You can also view the logs for the failed satellite by starting from the Logs folder in the object tree. To perform this task:

1. Select the Logs folder.

   The Log Details view opens in the contents pane. The logs in the view are organized by date in descending order. You can use the sort and filter facilities that are available from the Satellite Administration Center to modify the details view to reveal failed satellites. For example, you can filter the view to display the logs for a specific satellite, or only failed satellites.
2. Select the log that records the failure and right click.
3. Select **View Details** from the pop-up menu.

   The Log Details window opens. This window displays the full set of information for the log record that you selected. The information includes the batch that was being executed, as well as the batch step and the script that did not execute successfully. The log information also indicates

whether the error resulted in an external or an internal return code. For more information, see "Internal and External Error Return Codes" on page 175.

Assigning a Fix Batch to the Satellite

When you have determined the problem that caused the satellite to fail, the next step is to attempt to fix the problem. If the satellite reported an error that occurred on other satellites, you may already have a fix batch to fix that error. Otherwise, use the information from the log to create a fix batch. For information on creating a batch, refer to the online help that is available from the Satellite Administration Center.

To assign a fix batch to the satellite:

1. Open the Satellite Details view. For information on performing this task, refer to the online help that is available from the Satellite Administration Center.
2. Select the failed satellite in the Satellite Details view and right click.
3. Select **Fix** from the pop-up menu. The Fix Satellite window opens.

   Use the Fix Satellite window to assign the fix batch that you want the satellite to execute, and the batch step where you want the satellite to begin executing the batch. You can use the **...** push button to display the list of fix batches and unassigned batches that are available. If you do not have a batch that is suitable to fix the problem, create an unassigned batch and use it.

   Note: The satellite will only be able to execute fix batches. Because the satellite is in fix mode, it cannot execute group batches. If the user attempts to synchronize the failed satellite before you enable it to execute the fix batch, the SQLCODE -3934W is returned at the satellite.
4. Click **OK**.
5. Select the failed satellite in the Satellite Details view and right click.
6. Select **Enable** from the pop-up menu.

   When a satellite reports an error, its state changes to `Failed` in the Satellite Details view. In addition, the satellite is disabled. That is, the satellite cannot execute batches. The satellite must be enabled to execute the fix batch.
7. Click **OK**.
8. Have the user synchronize.
9. View the results of the execution of the fix batch.

   To perform this task, you should view the logs for the satellite to determine whether the satellite executed the fix batch successfully. In

addition, you can query the results of the fix batch. To perform the query, you can add a batch step to the fix batch that the satellite executed and have the satellite execute only that batch step, or you can use a different fix batch. If you are satisfied with the results of the fix, you are ready to promote the satellite back to executing its group batches. For details, see "Returning the Repaired Satellite to Production" on page 174. If you are not satisfied with the results of the fix, see "Debugging the Fix Batch".

Debugging the Fix Batch

If the satellite reported an error when it executed the fix batch, or the results of the fix batch are not satisfactory, you need to debug the fix batch.

To debug the fix batch:

1. Determine the problem with the fix batch.

   To perform this task, you should examine the logs for the satellite that executed the fix batch. If the log shows that an error occurred, you can begin the process of debugging the fix batch based on the error. If, however, an error did not occur, you can have the satellite execute a fix batch that queries the state of the satellite. You may have to try different queries to determine the problem.
2. Edit the fix batch to make the changes that you require.

   For information about this task, refer to the online help that is available from the Satellite Administration Center.
3. Open the satellite details view.
4. Select the satellite that you want and right click.
5. Select **Edit** from the pop-up menu.

   The Edit Satellite notebook opens.
6. On the Batches page, specify the fix batch that you want the satellite to execute, and the batch step where you want the satellite to begin executing the batch.
7. Enable the satellite to execute the fix batch, if necessary.

   This step is only required if the satellite reported an error when it executed the fix batch. When a satellite reports an error, that satellite is automatically disabled from executing batches. If the satellite successfully executed the fix batch, but the results of the fix batch are not satisfactory, the satellite remains enabled to execute fix batches. You can check the satellite details view in the Satellite Administration Center to determine whether the satellite is enabled or disabled.
8. Have the user synchronize.
9. View the results of the execution of the fix batch.

To perform this task, you should view the logs for the satellite to determine whether the satellite executed the fix batch successfully. In addition, you can have the satellite execute another fix batch to query the results of the previous fix batch. If you are satisfied with the results of the fix, you are ready to promote the satellite back to executing its group batches. For details, see "Returning the Repaired Satellite to Production". If you are not satisfied with the results of the fix, return to step 2 on page 173 and repeat the procedure.

Returning the Repaired Satellite to Production

When the fix that you apply to the satellite produces the results that you want, the satellite can return to production. That is, the satellite can return to executing its group batches when it synchronizes. To return the satellite to production:

1. Open the satellite details view.
2. Select the satellite that you fixed and right click.
3. Select **Promote** from the pop-up menu.

   The Promote Satellite window opens.
4. Depending on the fix that you applied, you may need to specify that the satellite resumes execution of one or more of its group batches at a different batch step than the next one to be executed. The fields of the Promote Satellite window indicate which group batches the satellite executes, and the batch step at which the satellite begins executing each group batch. Use the **...** push button to specify the batch step where the satellite is to begin executing its group setup, update, or cleanup batch, as required.
5. Click **OK**.

   If the satellite is already enabled (that is, it did not report an error when executing the fix batch), the next time that the user synchronizes, the satellite will download and execute its group batches, beginning execution at the batch steps that you specified. If the satellite reported an error when it executed the fix batch and the error is not important, you must enable the satellite before it can execute its group batches.
6. If the satellite is disabled, select it from the satellite details view and right click.
7. Select **Enable** from the pop-up menu.
8. Click **OK** when the Enable Satellite window opens to confirm that you want to enable this satellite to execute its group batches.

Using the DB2 Trace Facility

If requested to by DB2 Customer Service, you can use the DB2 trace facility (**db2trc**) on a satellite. You cannot issue this command from a remote DB2 CLP window. Instead, you can include the **db2trc** command in a fix batch, or you can use a product that enables you to enter commands from a remote console and keyboard.

Because the formatted or flowed output files are typically much larger than the dump file, it is recommended that you instruct the user to issue the **db2trc** command with the **dump** option. When the trace is dumped to a file, transmit the file to another DB2 Universal Database system. Then you can format the file before sending it to DB2 Customer Service. For more information about using the trace facility, refer to the *Troubleshooting Guide*.

The Satellite Software Version

The software version of a satellite is based on the release number used by DB2 Universal Database. You can find the official description of the release number in the C header file db2ApiDf.h. When a satellite synchronizes, it uploads its software version to the DB2 control server.

You can find the software version of the satellite in the satellite details view. For information about accessing this view, refer to the online help that is available from the Satellite Administration Center.

You can use release number for diagnostic purposes. In this situation, different release numbers may have different fixes to correct known problems. You can also use the release number to stage upgrades of DB2 on the satellites.

Internal and External Error Return Codes

The error return codes are associated with the execution of a batch step by a satellite. The return code can be either an external return code or an internal return code. You can determine the type of the return code by viewing the log details in the Satellite Administration Center.

If the event associated with a log record involves an internal code failure, the error return code is displayed under the **Internal code** column. For example, assume that a satellite, when attempting to execute a script, encounters a file system full condition. This error code will be displayed under the **Internal code** column when the satellite returns the results of the synchronization session to the satellite control server.

If the script of a batch step does not execute successfully (according to its associated success code set), the error return code is displayed under the **External code column.**

Satellite Progress File

Each satellite has its own log file to record information about its success or failure in executing the batch steps of the different batches that it executes during a synchronization session. The log file is called progress.log, and is located in the *instance_path*\satellite directory. The log file contains detailed information about the events that occurred on the satellite.

Re-Creating or Updating the satadmin.aut File

If the satadmin.aut file on a satellite is either destroyed or out-of-date, you can have the file either re-created or updated on the satellite.

Note: Before you can perform the following steps, you must either set the DB2SATELLITEID registry variable to the satellite ID, or you must be logged on as the user ID that will be used to synchronize. For additional information, see "Satellite ID Is Not Set on Satellite" on page 163.

To re-create or update the satadmin.aut file:

1. Start the DB2 Synchronizer application in test mode by issuing the following command on the satellite:

   ```
   db2sync -t
   ```

2. Click the **Test** push button.

   If the satadmin.aut file no longer exists on the satellite or is out-of-date, no authentication credentials exist that the satellite can use to authenticate with the DB2 control server. In this situation, the SQLCODE -3966 is issued, and the Connect to Control Database window opens. Use this window to specify the authentication credentials to use to connect to the satellite control database. For information about using the Connect to Control Database window, refer to the online help that is available from the window. When you provide the correct user ID and password, the DB2 control server will authenticate the satellite, and the satadmin.aut file will be either re-created or updated on the satellite.

3. Close the DB2 Synchronizer window.
4. Start another synchronization session via the application that you use to synchronize.

Part 2. DB2 Satellite Edition Installation and Migration

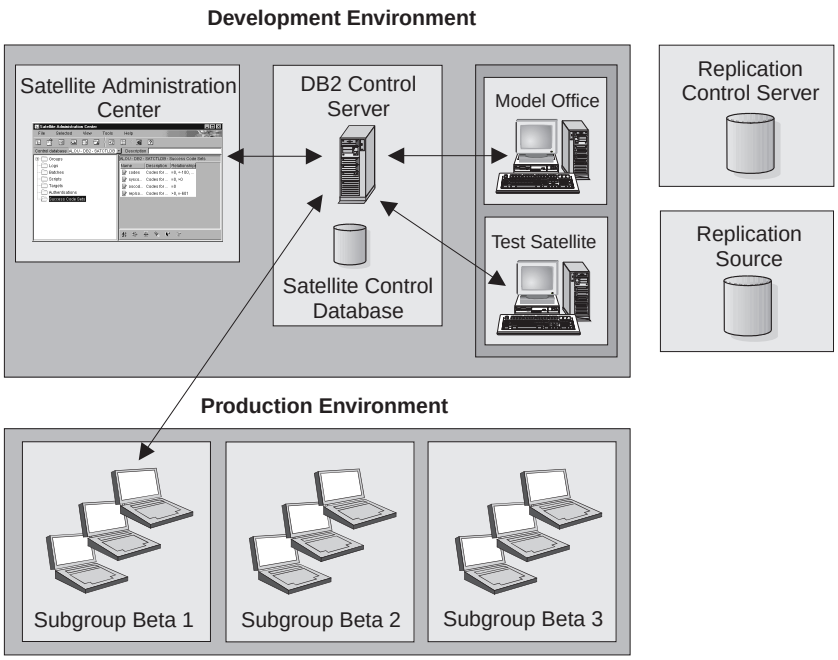
© Copyright IBM Corp. 1999

Chapter 12. An Introduction to Planning and Installation for the Satellite Environment

For a satellite environment, the installation scenarios will vary from one organization to another. At a minimum, you will need to install DB2 on the workstation that will act as the DB2 control server, and DB2 Satellite Edition on the workstations and laptops that will act as satellites.

You can also install the Control Center, which includes the Satellite Administration Center, on workstations for help desk staff and on workstations used for satellite environment administration. In large deployments, you can install the replication control server tables and a replication source database on a machine other than the DB2 control server.

The following diagram provides a schematic of the systems in your satellite environment:



© Copyright IBM Corp. 1999

Notes:

1. The Control Center, which includes the Satellite Administration Center, is assumed to be installed on workstations other than the DB2 control server. At a minimum, you will need to install the DB2 Administration Client on these workstations, although any other DB2 product that includes the Control Center can also be installed.
2. If you choose to administer the satellite environment from the same workstation on which you have installed the DB2 control server, select the Control Center component in addition to the Satellite control server component when you install DB2.
3. The replication control server, which contains the tables required to control replication, can be installed on any workstation that supports IBM DB2 products. In smaller installations, the replication control server can reside on the same workstation as the DB2 control server. For more information, see "Creating the Replication Environment" on page 97.

Chapter 13. DB2 Control Server Planning and Installation

This section describes the planning and installation information for the DB2 control server. To install the DB2 control server, you must install the optional Control Server component when installing DB2 Universal Database Enterprise Edition on the Windows NT or AIX platform. The DB2 control server allows you to set up and administer your satellite environment. For more planning information for the installation of DB2 Universal Database Enterprise Edition, refer to the appropriate DB2 *Quick Beginnings* book.

Memory Requirements

This section shows the *suggested* amount of memory that is required to run the DB2 control server. The memory requirements listed are estimates. Use this information to plan for systems with large numbers of synchronizing satellites.

Table 1. Memory Requirements for DB2 Control Server

Number of Concurrent Synchronizing Satellites	DB2 Control Server on Windows NT	DB2 Control Server on AIX
___ 5	32 MB	64 MB
___ 10	40 MB	80 MB
___ 25	48 MB	96 MB
___ 50	88 MB	186 MB
___ DB2 Administration Tools	30 MB	30 MB
Total Memory Requirements	__ MB	__ MB

© Copyright IBM Corp. 1999

Note: If you are running additional DB2 databases on the same workstation on which you have installed the DB2 control server, you will require additional memory to support these databases.

Disk Requirements

This section describes the minimum amount of disk space that is required to install the DB2 control server component, create the DB2CTLSV instance, and create an empty satellite control database (SATCTLDB). You should add these numbers to the base requirements for DB2, which can be found in the ″Planning for Installation″ section of the appropriate DB2 *Quick Beginnings* book. These estimates do not include the disk space required for the operating system, application, or communication products. Consult each product's documentation for these values.

Windows NT
The recommended minimum disk space for the DB2 control server, plus an empty satellite control database, is 13 MB.

AIX The recommended minimum disk space for the DB2 control server, plus an empty satellite control database, is as follows:

- Under the *INSTHOME*/db2ctlsv and *INSTHOME*/sqllib directory, is 23.2 MB, where *INSTHOME* represents the home directory of the instance.
- Under the /usr directory, is 0.4 MB

As you add groups, satellites, and scripts, and use the satellite control database, the satellite environment will consume more disk space. Formulas for estimating the size of your production SATCTLDB database can be found at the following URL: `www.software.ibm.com/data/db2/library/satellite`

Note: Synchronization logs can consume rather large amounts of disk space unless they are periodically purged.

Software Requirements

The DB2 control server requires that you must configure the DB2 control server instance, DB2CTLSV, to support TCP/IP, because it is the only communications protocol that DB2 Satellite Edition can use for synchronization.

The Satellite Administration Center, a component of the Control Center, will connect to the DB2 control server's satellite control database to set up and maintain satellites, groups, and the batches that the satellites execute when they synchronize. If the Control Center, and hence the Satellite Administration

Center, is running on a remote system, these connections can be made using NetBIOS, TCP/IP, IPX/SPX, Name Pipes or APPC. At a minimum, the appropriate TCP/IP communications software must be installed.

For more information on the software requirements for TCP/IP and the other protocols, refer to the "Planning for Installation" section of the appropriate DB2 *Quick Beginnings* book. "Installing the DB2 Control Server" on page 184 contains communication configuration information for the DB2 control server instance, DB2CTLSV.

Satellite Control Database Considerations

When the default satellite control database (SATCTLDB) is created, all its tables and indexes are created in the default tablespace, userspace1. For large deployments, where the DB2 control server will be managing thousands of satellites, you can change this design to allow more control over the SATCTLDB database's disk usage and performance. Disk tuning considerations for the SATCTLDB database are similar to those for any application database. For more information, refer to the *Administration Guide*.

The file satctldb.ddl in the sqllib\misc directory contains the DDL to define the SATCTLDB database. A copy of this file can be made and then modified to define the location and size of additional table spaces that will be used for the SATCTLDB database and to assign the SATCTLDB database's tables to these table spaces.

Note: When modifying this file, the table and index definitions should only be modified to add table space attributes. Do not make changes to the DDL, such as referential integrity definitions or trigger definitions, as the operation of the SATCTLDB database, the Satellite Administration Center and satellite synchronization may be adversely affected.

The installation process on Windows NT creates the SATCTLDB database when the Control Server component is selected. The installation process on AIX does not automatically create the SATCTLDB database. If you want to use additional table spaces for your SATCTLDB database, you should create a modified version of the satctldb.ddl file, and then use it to create the SATCTLDB database that will support your environment.

For more information on issuing the commands to create SATCTLDB database using the satctldb.ddl file, see "Installing the DB2 Control Server" on page 184.

Installing the DB2 Control Server

The DB2 control server is installed as an optional component of DB2 Universal Database on Windows NT and AIX during a custom installation. Select the Control Server component to install the DB2 control server. For basic component selection and installation information, refer to the appropriate DB2 *Quick Beginnings* book.

Setting up the DB2 Control Server on AIX

The SATCTLDB database is not automatically created during the DB2 control server installation on AIX. You should create a modified version of the satctldb.ddl file, and use it to create the SATCTLDB database that will support your environment. The file satctldb.ddl can be found in the sqllib\misc directory.

To setup the DB2 control server:

1. Create the DB2CTLSV instance.
2. Run the satctldb.ddl file to create the SATCTLDB database.

Creating the DB2CTLSV Instance

To create the DB2CTLSV instance:

1. Log on as user with `root` authority.
2. Change to the directory where the DB2 CD-ROM is mounted by entering the following command:

   ```
   cd /cdrom
   ```

   where */cdrom* represents the mount point of the CD-ROM drive.
3. Change to the `/cdrom/db2/aix` directory where the install image for the DB2 product that you want to install is located.
4. Enter the **./db2setup** command to start the DB2 installation program. The **DB2 Installer** window opens.
5. Use the **Tab** key to select the **Create** option. The **Create DB2 Services** window opens.
6. Select **Create a DB2 Instance**. The **DB2 Instance** window opens.
7. Rename the user name field with **db2ctlsv**. Use the default User ID and password. Change the home directory to `/home/db2ctlsv` to match the instance. Select **OK**. The **Fenced User** windows opens.
8. Select **OK** to accept the defaults. The **Create DB2 Services** window opens.
9. Select **OK**. The **Summary Report** window opens.
10. Select **Continue**.

Creating the SATCTLDB Database

To create the SATCTLDB database, perform the following steps:

1. Log in as db2ctlsv.
2. Ensure that the database server has been started; the **db2start** command has been issued.
3. If you do not want the default SATCTLDB database created, copy and edit the satctldb.ddl file to satisfy your requirements. For more information on customizing the DDL file, refer to the following URL: www.software.ibm.com/data/db2/library/satellite.
4. Enter the following command, from the sqllib/misc directory:

   ```
   db2 -tf prdctldb.ddl -z $HOME/prdctldb.log
   ```

   where prdctldb.ddl represents the modified version of the DDL file and is located in the sqllib/misc directory.
5. Check the prdctldb.log file for errors that may have been encountered during the creation of the SATCTLDB database.

Setting up the DB2 Control Server on Windows NT

For large deployments where thousands of DB2 satellites will be managed by the DB2 control server, you should use your own design, rather than the default, to allow more control over the SATCTLDB database's disk usage and performance. If you want to use your own design, you must first drop the default database and re-create the SATCTLDB database based on your customized DDL file.

To customize your design, perform the following steps:

1. Make a copy of the satctldb.ddl file and customize it to meet your needs. For the purposes of this example, prdctldb.ddl is used as the filename for the customized DDL file. For more information on customizing the DDL file, refer to the following URL: www.software.ibm.com/data/db2/library/satellite.
2. Open a command window by clicking on **Start**, then selecting **Programs –> DB2 for Windows NT –> Command Window**.
3. In the Command Window, ensure that the DB2INSTANCE environment variable is set to DB2CTLSV. To check the DB2INSTANCE environment variable enter the **set db2instance** command. The value returned must be DB2INSTANCE=DB2CTLSV. If the environment variable is not set to DB2CTLSV, enter the **SET DB2INSTANCE=DB2CTLSV** command to change its setting.
4. Drop the default SATCTLDB database that was created during the installation by entering the following command:

   ```
   DROP DATABASE SATCTLDB
   ```

5. Change to the directory where you have stored your customized prdctldb.ddl file.
6. Enter the following command:

   ```
   db2 -tf prdctldb.ddl  -z prdctldb.log
   ```

   where `prdctldb.ddl` represents your customized DDL file.

   Check the prdctldb.log file for errors that may have been encountered during the creation of the SATCTLDB database.

Other Considerations

When the default or customized version of SATCTLDB database is created, it is not backed up, nor is it enabled for forward recovery. For more information on defining a recovery strategy for this critical database, see the "Recovering the DB2 Control Server and Satellite Control Database" on page 132.

You can now proceed to create groups, satellites, and batches using the Satellite Administration Center.

Configuring DB2CTLSV for Inbound Connections

TCP/IP support is setup when the DB2 Universal Database Enterprise Edition is installed. For more information on TCP/IP configuration, refer to the appropriate DB2 *Quick Beginnings* book.

Chapter 14. Planning for DB2 Satellite Edition Installation

DB2 Satellite Edition has many components that can be used in your environment. Use the product planning information in this section to:

- Ensure that your system meets the prerequisites.
- Decide which components you want to install.
- Prepare information that will be required by the installation software to configure your satellite.
- Decide what authority with which to install.
- Decide which installation method to use: interactive or distributed.

Installation Authority on Windows NT

To fully use the satellite administration features, a user with local administrative authority should install DB2 Satellite Edition.

On Windows NT, DB2 Satellite Edition can be installed by a user with local administrative authority, or a local user without administrative authority. When a user with local administrative authority performs the install there are no installation restrictions; all components of DB2 Satellite Edition are available and DB2 Satellite Edition can be used by all system users.

When a local user without administration authority performs the installation, there are restrictions:

- The Control Server Synchronization component cannot be installed. This component installs the code required for synchronization. Without this support, the satellite cannot connect to the DB2 control server to report its status and receive administrative commands.
- Only the user who performed the installation will have DB2 Satellite Edition as part of their standard environment when they log on to Windows NT.

© Copyright IBM Corp. 1999

- When DB2 Satellite Edition is used as a client to access databases on other DB2 Universal Database host, or AS/400 machines, client authentication cannot be used. For more information on server authentication, refer to the *Administration Guide*.
- Services are not created for the DB2 instance and the DB2 Java applet server, but rather started as processes by adding them to the Windows registry run key.

As a result of this flexible installation ability, two installation scenarios arise:

- If a local user has installed DB2 Satellite Edition and then a user with local administrative authority installs an additional component of DB2 Satellite Edition or another DB2 product on the same machine, the administrator's installation replaces the installed DB2 Satellite Edition. The administrator's installation will remove all of the local user's services, shortcuts, and environment variables from the previous installation of DB2. The administrator's install will add its own shortcuts and environment variables.
- If a local machine user or a user with local administrative authority has installed DB2 Satellite Edition, and a second local machine user, without local administrative authority, attempts a DB2 product install on the same machine, the second user's install will fail. The second user will receive a message that they must be a local administrator to install the product.

  The second user's install would fail even for the addition of a DB2 Satellite Edition component.

Interactive Installation or Distributed Installation

You can install DB2 Satellite Edition using one of following two methods:

- Interactive installation. You will likely use an interactive installation for your initial satellite. You can also use this method to install your model office and test systems.
- Distributed installation using a response file. To perform a distributed installation, you can use a customized response file. This method is appropriate for mass deployments and the installation of hundreds or thousands of satellites, since the response file can be re-used to perform identical installations on different machines.

  You can use response files to install test satellites. You may need to perform many test satellite installations, and the response file install may be better suited for this task than an interactive install.

  Distributed installation also supports two unique features, which are not part of the interactive install. You can:

  – Specify to the installation program that the desktop shortcuts are *not* to be created. Use this feature if you want to initiate DB2 operations, such

as synchronization, from your business application. This feature will allow you to hide DB2 from your users.

– Specify the name of a client profile to be imported by the installation program. The installation will set up the connectivity information for the satellite. The information required to establish connections to the DB2 control server, the database containing the replication control tables, and the systems that the satellite will be replicating data with can be specified in the client profile. This feature is particularly important when performing mass deployments, because you do not have to configure each satellite separately.

To prepare for mass deployment, the response file generator utility, **db2rspgn**, can generate a response file based on installed, configured, and tested model office. The response file generator can export a client profile from an existing DB2 system, such as Control Center, and provide a keyword to import this file in the generated response file. This response file can be used to install systems that will have the same components and configuration. The model office is normally used as a source to generate a response file.

If you installed components on the model office that you do not want on the production or test satellites, the generated response file will install these components, unless you modify the response file to remove the selection of these components.

For more information on response files and the response file generator, see "Chapter 16. DB2 Satellite Edition Distributed Installation" on page 199. For more information on mass deployment, see "Chapter 10. Performing a Mass Deployment" on page 145.

Choosing Components to Install

Several DB2 Satellite Edition components provide specialized support for your satellite environment. You can install additional components on a satellite as you require them. Use the information in this section to make component choices during the initial installation of your satellite.

Inbound Remote Administration

The Inbound Remote Administration component provides support for remote DB2 Universal Database systems to connect to and administer this system. The DB2 Control Center, Command Line Processor (CLP), and any other vendor tool that uses the DB2 administrative API interfaces, can perform

administrative tasks. For more information about using the Control Center to administer a satellite, see "Appendix D. Administering DB2 Satellite Edition from the Control Center" on page 245.

The Inbound Remote Administration component only supports TCP/IP connections.

Replication - Apply and Capture

DB2 data replication replicates changes between relational tables. In the satellite environment, replication tasks will likely involve replicating changes between tables on the satellite and your corporate systems. The table that is the source of changes is called the *source table* and must reside on a non-satellite system. The table that is updated with source table changes is called the *target table* which resides on one or more satellites. All target table types including replica's which support changes at both the source and target table are supported on the satellite. The DB2 server that contains the replication control information is called the *replication control server*. For more information, see the *Replication Guide and Reference*.

The Apply component installs the Apply program which copies changes to target tables. The Capture component installs the Capture program, which captures changes made to source tables. In most satellite environments, you will install both replication components.

The Apply program then sends these changes to the corporate system. This program also receives changes made to the corporate customer data and then applies them to the local copy on the satellite.

For example, consider satellites used by a travelling sales force that require access to customer data, such as names and addresses. Rather than storing the data for all customers, each satellite only contains the system user's customers, a subset of the customers stored on the central corporate database. If the customer data can be updated on both the corporate system and the satellite, then changes must flow from the corporate system to the satellite, and from the satellite to the corporate system. Both the Apply and Capture programs are needed to keep both databases synchronized. The Capture program on the satellite captures the changes made to the local copy of the customer data.

If you will only be using replication to maintain tables containing reference data on a corporate database, data that is not updated on the satellite, you only need to install the Apply component on the satellite.

Replication will also need to be installed on your corporate systems if you are using replication on your satellites. For more information on replication, and the Capture and Apply programs, refer to the *Replication Guide and Reference*.

Control Server Synchronization

The Control Server Synchronization component installs the code required for DB2 satellite synchronization and replication. Using synchronization, the satellite can connect to the DB2 control server, using the **db2sync** command or a user-written application that uses the synchronization APIs, to report its status and receive administrative commands.

DB2 Connect Support

The DB2 Connect Support component allows the satellite to directly connect to a host or AS/400 system if:

- You are replicating data with a DB2 database on a host or AS/400 system.
- The tables for the replication control server reside on a DB2 database on a host or AS/400 system.

Alternatively, you can install DB2 Connect Enterprise Edition on a separate machine, and route the satellite's connections to the DB2 database on a host or AS/400 system through DB2 Connect. For more information, see "Connectivity Scenarios" on page 193.

Miscellaneous Tools

The Miscellaneous Tools component consist of tools that provide administration function. There are two subcomponents:

- Database tools used with databases on the satellite.
- Client tools used when the satellite is acting as a client to another system.

You should only install these tools on your test satellites, because these tools are designed for administration staff and IBM service.

Memory Requirements

This section shows the suggested amount of memory that is required to run DB2 Satellite Edition. The memory requirements listed here are estimates; the actual amounts required depend on the functions you are using and the size of your databases. You should have a minimum of 32 MB of memory to accommodate average databases.

Disk Requirements

This section shows the minimum amount of disk space that is required to install the DB2 Satellite Edition components. The disk space requirements do not include the operating system, application, and communications products. Consult each product's documentation for these values.

To estimate the disk requirements for a particular configuration, add the recommended minimum disk sizes for the components that you want to install. Include an allowance for your application data.

Table 2. Disk Requirements for DB2 Satellite Edition

	Recommended Minimum Disk Space
DB2 Universal Database Satellite Edition	
DB2 Satellite Edition (base)	30 MB
- ODBC Support	0.1 MB
- Control Server Synchronization	0.3 MB
- Replication - Apply and Capture	2.5 MB
- Java Enablement (JDBC & SQLJ support)	1.4 MB
- Inbound Remote Administration	n/a
- DB2 Connect Support	1 MB
- Miscellaneous Tools (Database & Client Tools)	1.4 MB
- Far-East Code Page Conversion Support	6.8 MB
Total Disk Space Required	___ MB

If you perform a compact install, only the base and ODBC component are installed. If you perform a typical install, the base, and the ODBC, Replication, Inbound Remote Administration and Control Server Synchronization components are installed.

Software Requirements

This section outlines the software required to run DB2 Satellite Edition.

Table 3. Software Requirements

Product	Hardware/Software Requirements	Communications
	Windows 32-bit Operating Systems	
DB2 Universal Database Satellite Edition	• Windows 95 Version 4.00.950 or later • Windows 98 • Windows NT Version 4.0 with Service Pack 3, or later	The Windows 32-bit operating systems provide TCP/IP connectivity. **Notes:** 1. If you plan to use the ADSTAR Distributed Storage Manager (ADSM) facilities for backup and recovery of your databases, you require the ADSM Client Version 3 or later. 2. If you have the IBM Antivirus program installed on your operating system, it must be Version 3.0 or later.

Connectivity Scenarios

TCP/IP is the only communication protocol supported by DB2 Satellite Edition. The following sections describe the connectivity scenarios that are supported.

Outbound Connections

DB2 Satellite Edition can support outbound connections to:

- The DB2 control server when the satellite synchronizes.
- The database containing the replication control tables that control the Apply program.
- Any system running a DB2 family product to replicate data.

DB2 Satellite Edition can also act as a client to any DB2 Universal Database server. Applications running on the satellite can access data stored in any DB2 Universal Database database.

If you are replicating data with a host or an AS/400, or if the tables for the replication control server reside on a host or an AS/400 database, you have two connectivity options:

- You can install and use the DB2 Connect Support component to connect directly to the host or AS/400 systems that your satellite will access. You must use the custom installation type to select this component.

  Note: The DB2 Connect Support component supplied with DB2 Satellite Edition can only be used for these purposes.
- You can install DB2 Universal Database Connect Enterprise Edition on a separate system, and route the satellite's connections to the host or AS/400 systems through it.

Inbound Connections

DB2 Satellite Edition can support inbound connections for satellite administration. Typical administrative activities include problem determination and correction by help desk or administrative staff. The CLP or Control Center can be used from a remote DB2 Universal Database to perform administrative tasks. Other vendor tools that use the DB2 administrative API interface, can also be used to perform administrative tasks. For more information about using the Control Center, see "Appendix D. Administering DB2 Satellite Edition from the Control Center" on page 245.

The Inbound Remote Administration component must be installed for inbound connection support. This component is installed by default when you use the typical installation, and can be selected when you use the custom installation. Inbound administrative connections are only supported over TCP/IP. DB2 Satellite Edition will be configured to use TCP/IP with default values for the service name and port number. You can override the default values during an interactive install, only if you have chosen the custom installation. The values for service name and port number can be overridden during any response file install.

Chapter 15. Installing DB2 Satellite Edition

This section describes the installation of DB2 Satellite Edition on Windows 32-bit operating systems using an interactive installation. To perform a distributed installation, see "Chapter 16. DB2 Satellite Edition Distributed Installation" on page 199.

Notes:

1. Before reading this section, ensure that you have read "Chapter 14. Planning for DB2 Satellite Edition Installation" on page 187. The instructions in this section are detailed from a local administrator's perspective. If you want to perform the installation as a local system user without administration authority, then you should log on with the local user account instead of the account belonging to the Local Administrator's group.
2. If you are migrating from a previous version of DB2, you must complete certain procedures before installing DB2 Universal Database Version 6. See "Chapter 17. Migrating DB2 Satellite Edition from Previous Versions and Releases" on page 207.

Before You Begin

Before you begin the installation, ensure that you have the following items and information:

__ 1. Ensure that your system meets all of the memory, hardware, and software requirements to install DB2 Satellite Edition.

__ 2. On Windows NT, you will need a user account to perform the installation. The account you specify should:

- Be defined on the local machine.
- Belong to the Local Administrator's group.
- Have the *Act as part of the operating system* advanced user right.

For information on how to grant user rights, refer to the Windows NT online help.

__ 3. By default on Windows NT, the setup program will create a user account using the username `db2admin` and password `db2admin`. You can accept these default values or create your own user account by

© Copyright IBM Corp. 1999

modifying the default values. If you create your own user account, ensure that it conforms to DB2's naming rules. For more information on DB2 naming rules, see "Appendix E. Naming Rules" on page 247.

Note: If you use the default user account `db2admin`, and do not change the default password on the setup screen, you should change this password immediately following the installation. The `db2admin` password is the default for any installation, and is well known.

Performing the Installation

To install DB2 Satellite Edition, perform the following steps:

Step 1. On Windows NT, log on to the system with the user account that you created to perform the installation.

Step 2. Stop any other programs, including all DB2 services and Netfinity service programs, so that the setup program can update files as required.

Step 3. Insert the CD-ROM into the drive. The auto-run feature automatically starts the setup program. The setup program will determine the system language, and launch the setup program for that language.

Note: To manually invoke the setup program, perform the following steps:

a. Click **Start** and select **Run**.

b. In the **Open** field, enter the following command:

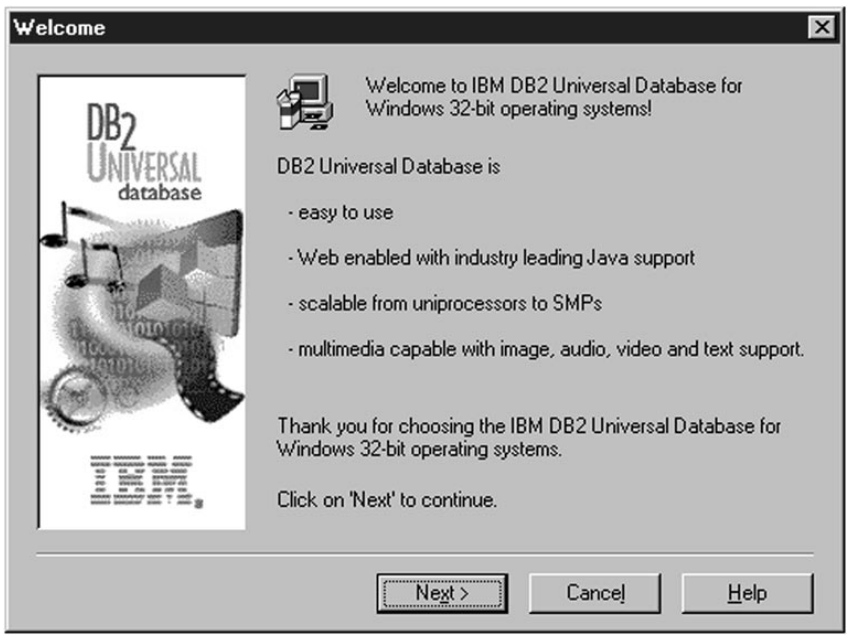
```
x:\setup /i language
```

where:

- *x:* represents your CD-ROM drive
- *language* represents the country code for your language (for example, EN for English). For more information about country codes, see "Code Page and Language Support" on page 251.

c. Click on **OK**.

Step 4. The Welcome window opens.



Step 5. Respond to the setup program's prompts. Online help is available to guide you through the remaining steps. Invoke the online help by clicking on **Help** or pressing **F1** at any time. You can click on **Cancel** at any time to end the installation.

Note: For information on errors encountered during installation, see the db2.log file. The db2.log file stores general information and error messages resulting from the install and uninstall activities. By default, the db2.log file is located in the x:\db2log directory, where x: represents the drive on which your operating system is installed.

For more information, see "Installation Problems" on page 161.

Based on your component selections, the installation program has:

- Created DB2 program groups and items (or shortcuts).
- Created the following services on Windows NT: DB2 JDBC Applet Server, DB2 Security Server, DB2 Remote Command Service.

  Note: These services are created if you are performing the installation using a user account without administration authority.
- Updated the Windows registry (Windows NT only).
- Created a default instance named DB2.

On Windows NT, the DB2 instance is added as a service if the installation is performed by a user with local administration authority. If you selected **Automatically start the DB2 instance at boot time**, the service's start-up type is set to **Automatic**; otherwise, it is set to **Manual**.

On Windows 9x, if you selected **Automatically start the DB2 instance at boot time**, the **db2start** command is added to the **Startup** folder.

After performing an interactive install, when you reboot the satellite, the DB2 Synchronizer application is automatically started in test mode.

In test mode, you can use the **db2sync -t** command to perform the following tasks:

- Catalog the DB2 control server (DB2CTLSV) and its satellite control database (SATCTLDB).
- If you did not specify a user ID and password for connection to the SATCTLDB database during the installation, specify them now. For information on group authentication credentials, see "Tables and Authorizations" on page 68.

Note: Before you enter the **db2sync –t** command, you must set up the DB2 control server by defining this specific satellite and an application version to a group.

You can run the DB2 Synchronizer application in test mode at any time by entering the following command:

```
db2sync -t
```

For more information on the **db2sync** command, see "Running a Test Synchronization" on page 74 and "Appendix C. db2sync - Start DB2 Synchronizer" on page 243.

Note: If the satellite is remote to the DB2 control server, or is running a different version of DB2, or runs a different operating system, you need to bind the database utilities on the satellite, including the DB2 CLI, to the SATCTLDB database on the DB2 control server. This binding task only needs to be performed once, it does not need to be performed from every satellite. See "Binding Utilities" on page 113 for details.

Chapter 16. DB2 Satellite Edition Distributed Installation

This section describes how to perform a distributed installation of DB2 Satellite Edition by using a response file. You can customize the response file by specifying keywords, or you can use the response file generator utility to generate a response file from an existing implementation of DB2 Satellite Edition.

Distributed installation using a response file is well suited for mass deployments. When you have finished reading through this section, see "Chapter 10. Performing a Mass Deployment" on page 145 for more information on response files and mass deployment.

What is a Response File?

The first step in any type of distributed installation is to create a response file. A response file is an ASCII file that can be customized with the setup and configuration data that will automate an installation. You would normally enter the setup and configuration data during an installation, but with a response file, the installation can proceed without any intervention.

A response file specifies such configuration and setup parameters as the destination directory, and the products and components to install. It can also be used to set up the following settings:

- Global DB2 registry variables
- Instance variables
- Database manager configuration settings

Available Sample Response Files

The DB2 CD-ROM includes a ready-to-use sample response file with default entries for DB2 Satellite Edition. The sample response file, db2udbse.rsp, is located in x:\db2\common directory, where `x:` represents the CD-ROM drive.

© Copyright IBM Corp. 1999

Response File Keywords

This section describes the most important keywords that you will specify when performing a distributed installation of DB2 Satellite Edition on Windows 32-bit operating systems.

COMP

Specifies the components you want to install. The setup program automatically installs components that are required for a product, and ignores requested components that are not available.

Note: Component selections have no effect unless you specify a custom installation (`TYPE = 2`).

DB2.AUTOSTART

Specifies whether or not to automatically start the DB2 instance each time the system is rebooted.

By default, the DB2 instance starts automatically unless this parameter is set to `NO`.

DB2.SATCTLDB_USERNAME and DB2.SATCTLDB_PASSWORD

Specifies the user ID and password that will be used by the satellite when connecting to the satellite control database (SATCTLDB) on the DB2 control server. The user ID and password are used to authenticate the connection to the database. It is not mandatory that you enter this information at installation time, but it is recommended that you do so if you have the information. The user ID and password cannot be authenticated at installation time.

If you choose not to provide this information at installation time, you can do so later by running the DB2 synchronization application in test mode by issuing the **db2sync -t** command. You will then be prompted for the user ID and password required to make the connection.

DB2.DB2SATELLITEID

Specifies the unique ID for the satellite and sets the DB2SATELLITEID registry variable on the satellite. The ID must be unique across all groups that are recorded at the DB2 control server. It must match an ID defined for a satellite at the control server. The satellite ID is used during the synchronization process to identify the satellite. The ID can be a maximum of 20 characters.

It is not recommended that you provide the DB2SATELLITEID in the response file as it must be unique, unless you are customizing the value of DB2SATELLITEID for each system the response file will be used on. The DB2SATELLITEID can be set after the installation using the **db2set** command.

If not specified, the Windows login ID is used in its place during the synchronization process.

DB2.DB2SATELLITEAPPVER
Specifies the satellite's application software version. The version can be a maximum of 18 characters and numbers. The value specified must match an application version defined for the group to which the satellite belongs as defined at the satellite control server. If it does, then scripts associated with this application version will be used to maintain the satellite during the synchronization process. A default version V1R0M00 has been supplied, but this value can be changed. These values can be set or changed after installation.

DB2.USERDB_NAME
Specifies the name for the database that DB2 can create during the installation of DB2 Satellite Edition. If no value is provided, the database is not created.

DB2.USERDB_REP_SRC
Specifies that the database will be used as a DB2 replication source. DB2 will configure the database so that changes to application data can be written to change tables by the Capture program. The Apply program will then use the captured changes to synchronize application data with other systems. In addition to configuring the database to capture data changes, you must define the application tables for which changes will be gathered. For more information on the *data capture changes* parameter of the `CREATE TABLE` statement, refer to *SQL Reference*. This configuration step can be done when the installation process is complete and the application tables are defined in the database.

DB2.USERDB_RECOVERABLE
Specifies that the database on the satellite is recoverable. DB2 will configure the database for forward recovery by setting the *logretain* parameter to `recovery`. You will be required to manage the database log files and take backups of the database. Before the database can be used, you will have to back it up. If this keyword is not specified, the database will not be configured for forward recovery. Database log files will be managed automatically by DB2. You will not be required to take a backup copy of the database before the database can be used. However, data can be lost if a disk failure occurs.

For more information on database recovery, see "Chapter 9. Recovering the Satellite Environment" on page 129.

DB2SYSTEM
Specifies a name for the system which is unique within a network. This parameter must be specified.

FILE Specifies the destination directory for a DB2 product.

PROD Specifies the product you want installed. You want to install `UDB_SATELLITE` for DB2 Satellite Edition.

REBOOT
Specifies whether to restart the system when the installation is complete.

TYPE Specifies the type of install.

The options are:

- 0 = Compact
- 1 = Typical (default)
- 2 = Custom

Note: A compact or typical install type will ignore any custom keywords (COMP).

Generating a Response File

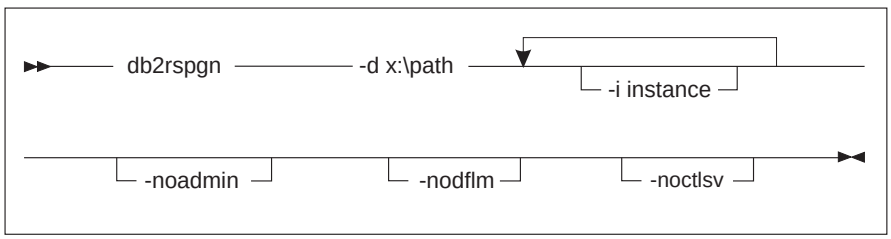
The response file generator utility creates a response file from an existing installed and configured DB2 product. You can use the generated response file to re-create the exact setup on other machines.

For example, you can install and configure a DB2 Satellite Edition model office to represent your production satellites. When you are satisfied with the configuration and behavior of the model office, then you can run the response file generator and create a response file and client profiles. For more information, see "Chapter 10. Performing a Mass Deployment" on page 145.

The response file generator creates a response file for the installation and client profiles for each instance that you specify. You can then use the response file to create identical satellites across your network.

The response file generator also gives you the option to create just the installation response file, without a client profile, which would allow you to create identical copies of your model office, without that configuration information, such as node and database directory entries.

The syntax for the **db2rspgn** command is as follows:



-d Destination directory for a response file and any supporting files. This parameter is required.

-i A list of instances for which you want to create a profile. The default is to generate an instance profile file for all instances. This parameter is optional.

-noadmin, –nodlfm, and –noctlsv
These parameters are not applicable to DB2 Satellite Edition.

For example, to create a directory called db2rsp at the base of the current drive, and have the response file generator place the response file and the client profiles for all the instances in this directory, enter the following command.

```
db2rspgn -d \db2rsp
```

Each instance would have a profile created.

You can create the same directory as the first example, but only include the client profiles for instances `inst1`, `inst2`, and `inst3`, by entering the following command:

```
db2rspgn -d \db2rsp -i inst1 inst2 inst3
```

If you are planning to set up and configure identical DB2 products, you only need to specify the installation response file when you perform the installation. The installation response file that was created by the response file generator will automatically call each instance profile. You only need to ensure that the instance profiles are located in the same drive and directory as the installation response file.

Making DB2 Satellite Edition Files Available for Installation

To copy the required files from the CD-ROM to the shared network drive that will act as the code server:

1. Insert the appropriate CD-ROM into the drive.
2. Create a directory on the code server by entering the following command:

   ```
   md c:\db2prods
   ```

3. Enter the **cpysetup.bat** command to copy the DB2 installation files to your code server. This command is located in the *x:*\db2\common\ directory, where *x:* represents your CD-ROM drive.

   The command syntax is as follows:

   ```
   cpysetup.bat directory language
   ```

   where:

   - *directory* represents the directory that you created in the previous step (for example, c:\db2prods).
   - *language* represents the two-character country code for your language (for example, `en` for English). Table 5 on page 251 lists the keywords for each available language.

   For example, to copy all of the English DB2 install files to the c:\db2prods directory, enter the following command:

   ```
   cpysetup.bat c:\db2prods en
   ```

Set up Shared Access

This section will allow you to grant your network workstations access to the code server. From the code server, perform the following steps:

1. Click on **Start** and select **Programs->Windows Explorer**.
2. Select the directory that you want to share. For example, c:\db2prods.
3. Select **File->Properties** from the menu bar. The properties window for the directory will open.
4. Select the **Sharing** tab.
5. Select the **Shared As** radio button.
6. In the **Share Name** field, enter a share name. For example, `db2nt`.
7. To specify *read access* for everyone:
   a. Click on the **Permissions** push button. The Access Through Share Permissions window opens.
   b. Ensure that the **Everyone** option is selected in the **Name** box.
   c. Click on the **Type of Access** drop down box and select the **Read** option.

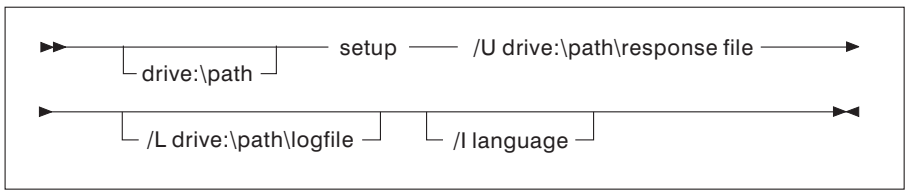
d. Click **OK**. You are returned to the properties window of the directory for which you want to set up shared access.
e. Click **OK**.

Installing Using a Response File

Before you begin any mass deployment or distributed installation using a response file, you should understand how the response file is used.

To run the response file setup program, perform the following steps:

1. Click on **Start** and select the **Run** option. The Run window opens.
2. In the **Open** field, enter the path to the setup program. The syntax of the setup command is an follows:



where:

/U Specifies the fully qualified response file name. If you changed and renamed the sample response file that is provided, make sure that this parameter matches the new name. This parameter is required.

/L Specifies the fully qualified log file name, where setup information and any errors occurring during setup are logged. This parameter is optional.

If you do not specify the log file's name, DB2 names it db2.log and stores it in the db2log directory on the drive on which your operating system is installed.

/I Specifies the two-character country code that represents your language. If you do not specify the language, setup will determine the system language, and launch the appropriate DB2 install for that language. This parameter is optional. For more information about country codes, see "Code Page and Language Support" on page 251.

For example, to install DB2 Satellite Edition using a custom response file that you created called satellite.rsp (located in the same directory as the DB2 install files), enter the following command:

```
x:\setup /U satellite.rsp
```

Note: If you are using a response file that was created using the response file generator, you must ensure that all the client profiles are located in the same drive and directory as the response file that you specify.

3. Click **OK** to start the setup program. The installation proceeds without further action on your part.

Chapter 17. Migrating DB2 Satellite Edition from Previous Versions and Releases

This section describes how to migrate previous versions of DB2 to the Version 6 format.

DB2 Universal Database Version 6 supports the migration of DB2 Version 2.x, Database Server Version 4.x, and DB2 Version 5.x to a format usable by DB2 Version 6.

Note: If you are migrating from a previous product you should review "Appendix B. DB2 Satellite Edition Differences" on page 241 to ensure that DB2 Satellite Edition will meet your functional requirements.

Planning the Migration to DB2 Satellite Edition

When you install DB2 Satellite Edition on a system that already has a production DB2 environment, the pre-Version 6 code is replaced, and the DB2 environment is migrated to Version 6. Because the system was already in production, it will be at a different state than a system that did not have DB2 on it before DB2 Satellite Edition was installed. For example:

- One or more databases may have been created on the system.
- Tables, indexes, and other objects to support a business application will have been created and loaded with data.
- The system may be replicating data with another DB2 Universal Database system or a DB2 for OS/390 system.
- Tuning of the system for optimal performance may have been completed; this could involve setting database manager and database configuration values.

© Copyright IBM Corp. 1999

By contrast, a newly installed system does not have this level of initial customization. For these new systems, you can use the Satellite Administration Center to define batches for the group to accomplish this customization when they first synchronize.

If the migrated systems and new satellites will be running the same end-user application, you will likely want them to be members of the same group. You do not, however, want the migrated systems to run the batch steps that a new system will run to perform the initial customization. The Satellite Administration Center supports this by allowing you to set an execution starting point for a satellite. In our example, you would set the execution starting point for the migrated satellite so that the initial customization scripts are bypassed.

In addition to the steps described in the "Post-Installation Steps" on page 219, there is one required migration step that is not completed by the Version 6 installation program; the databases created on a version of DB2 earlier than Version 6 are not migrated to the Version 6 format. Also, you may want to perform several optional post migration actions:

- Rebind packages. All packages are invalidated during database migration, and under normal operating conditions they will be rebound at first use. Rebinding, however, takes time. To avoid your users waiting for a rebind to complete, you can use the **db2rbind** command to rebind all existing packages.
- Adjust the database manager or database configuration parameter values to use specific DB2 Satellite Edition features such as automatic log management when utilizing replication, which can now be accomplished by setting `logretain = Capture`.

Because all migrated systems will have to have the same commands run on them, you can placed these commands in a one–time fix batch, that can then be run by all the migrated satellites. The fix batch would run the database migration command, and perform any post–migration steps that are required to get the migrated systems ready to start running group batches. When the satellite has run the fix batch, the satellite can be promoted from fix batch mode to group batch mode, during which time the execution starting point can be set.

A Sample Scenario

Consider a sample scenario to outline the process that will be required to successfully complete a sample migration. In this scenario, you will migrate a set of systems running a DB2 Personal Edition Version 5 environment to DB2 Satellite Edition, while at the same time supporting the implementation of new systems running DB2 Satellite Edition. Compare the state of these systems:

State of the Satellite	Satellite migrated from Version 5	Newly installed Satellite
Database needs migration	Yes	No
Database definitions require setup	No	Yes
Database and Database Manager configuration values require customization	Yes - to use new DB2 Version 6 features and DB2 Satellite Edition features.	The Database and Database Manager configuration values, which are optimized for DB2 Satellite Edition and its features, are set up when the system is installed.
Packages are invalid and need to be rebound	Yes	No
Bind application packages if needed	No	Possibly
Data replication has been defined	Yes	No

Assume that once both types of satellites have reached a common state, they will then run group batches to accomplish ongoing administration and to run the data replication process.

Every satellite migrated from a pre-Version 6 environment will need to have its database migrated, and you may want to perform other optional post migration activities. You can accomplish this in one of two ways and you will have to decide which method is more appropriate for your environment:

- "Using Your Own Scripts" on page 210. You can develop scripts and run them using your own methods when DB2 Satellite Edition has been installed on the system.
- "Using a Fix Batch" on page 210. A fix batch allows you to determine which systems have performed the migration and to determine its success by examining the script execution. When you have verified that a successful migration has been completed, you can then set the satellite to begin running group batches.

You should use a fix batch because the results on each system can be easily tracked using the Satellite Administration Center. When you have decided on the method you will use and read the appropriate section, continue to the next section "Migrating from Previous Versions of DB2" on page 214.

Using Your Own Scripts

Situations may exist where you will choose to use your own scripts, and an established technique to execute them, to migrate your databases to the Version 6 format, and to perform other customization in preparing the migrated satellite to join a group and start executing group batches. In this situation, the satellites do not have to run a fix batch. Instead you can:

1. Identify the group and application version to which the migrated satellites will belong.
2. Define the systems to be migrated in the SATCTLDB database using the Satellite Administration Center. Create these satellites in the group that they will belong to. You should specify a subgroup for these migrated systems, or have some way of easily identifying them. Using a subgroup, for example, `migrate`, you can filter the subgroup for multi-select actions, such as enable, or view these migrated satellites in the details view. When the satellites are defined, they are automatically placed in disabled state, and cannot run batches when they synchronize.
3. Set the Execution Starting Point for each of the migrated satellites to the step in the group's setup, update and cleanup batch at which these systems will start the script execution on first synchronization. A sample SQL script, which can be used to set the execution starting point for all the satellites in a subgroup (it will required customization before it is run), is provided on the web at the following URL: `www.software.ibm.com/data/db2/library/satellite`.
4. Enable the satellites to run batches by multi-selecting for the "migrate" subgroup and selecting the "Enable" action.

The migrated satellite will now run group batches each time it synchronizes. You may want to have the satellite's end user run the synchronizer application in test mode, by running **db2sync -t**, to ensure that the satellite is set up to synchronize before the first production synchronization.

Using a Fix Batch

Two approaches that would be valid:

- The migrated satellites could run fix batches and the new satellites could run group batches to reach the common state.
- The new satellites could run fix batches and the migrated satellites could run group batches to reach the common state.

You should consider the first approach. The satellites to be migrated are known, and can be tracked until all of them have run the fix batch. As the migrated satellites complete the execution of the fix batch, then promote the migrated satellite for group batch execution, and set the point at which it would being running group batches. You may want to place the migrated systems in a special subgroup, for example `migrated`, when they are created so that their progress can easily be tracked using the sort and filter capabilities of the Satellite Details view.

Using this technique to handle migrated satellites, new satellites can be added to the group at any time, as the group batches they run would always perform the initial customization for them, without intervention from an administrator to set them up to run a special fix batch.

Using the scenario, as described in "A Sample Scenario" on page 209, and assuming that you create the database during installation, the setup group batch would contain the following steps to accomplish the initial customization of a new system:

1. Customize the Database and Database Manager configuration values as required
2. Define replication - create the target tables, and setup subscriptions
3. Define indexes on the target tables

An application version would be created for the group, and the setup batch would be associated with the application version. An update batch should be associated with the application version, which would contain scripts to run the data replication process.

As new satellites are defined in the group and connect for synchronization, the setup batch would be run to customize them, after which they would continue to run the update batch on each synchronization, which runs the data replication process.

To handle the systems migrated from a pre-Version 6 environment to DB2 Satellite Edition, you would use a fix batch. The fix batch should include the following steps:

1. Run the **db2migr** command to migrate each database
2. Update the Database and Database Manager configuration values as required, to take advantage of new Version 6 function. For example, setting `logretain = Capture`.
3. Rebind the packages
4. Disable or remove the current method that executes data replication.

A complete list of the post–database migration activities that can be performed may be found in “Optional Post Migration Activities” on page 219.

As discussed in “Chapter 4. Cataloging Instances and Databases” on page 51, the targets of the script execution must be cataloged on the satellite. The instance and database names must match those for on the Control Center’s instance, as these will have been used to name targets when the targets are created using the Satellite Administration Center. In addition to other migration activities, the fix batch may have to catalog additional node and database directory entries so that all the targets of script execution are defined on the migrated satellite.

The systems migrated to DB2 Satellite Edition would have to be created using the Satellite Administration Center, and would be set up to run the fix batch on their initial connection. As outlined above, once this fix batch is run, these satellites can be promoted to run group batches and the initial step that they would run in each batch can be set.

Step by step, here is what you would do:

1. Identify the group and application version to which the migrated satellites will belong.
2. Define the systems to be migrated in the SATCTLDB database using the Satellite Administration Center. Create these satellites in the group that they will belong to. You should specify a subgroup for these migrated systems, for example, `migrated`, or have some way of easily identifying them. Using a subgroup, you can filter the subgroup for multi-select actions, such as enable, or view these migrated satellites in the details view. When the satellites are defined, they are automatically placed in disabled state, and cannot run batches when they synchronize.
3. Create a fix batch that contains the required steps. This fix batch should be thoroughly tested before it is made available for execution by the production satellites that are being migrated.
4. Put each satellite in the subgroup in fix mode and specify the fix batch that will be run. A sample SQL script, which can be used to perform this task, is provided on the web at the following URL:
   `www.software.ibm.com/data/db2/library/satellite`
5. Enable the satellites to run batches by multi-selecting for the ″migrate″ subgroup and selecting the ″Enable″ action.
6. Migrate each system to DB2 Satellite Edition.
   a. Before you perform the installation, please make sure that you have followed the steps in the “Pre-Installation Migration Steps” on page 214.

b. Install DB2 Satellite Edition on each of the systems. Each migrated system will not have a catalog entry for the satellite control database (SATCTLDB). Cataloging the SATCTLDB database can be done in two ways:
   1) You can run the **db2sync** application in test mode (`db2sync -t`). You will be prompted to enter the information to access the SATCTLDB database.
   2) A client profile file can be imported during the installation process to catalog the SATCTLDB database using the DB2.CLIENT_IMPORT_PROFILE keyword. The client profile should only contain the information to catalog the SATCTLDB database. You can produce this profile from the catalog information in the Control Center's instance using the Client Configuration Assistant. For more information, see "Chapter 4. Cataloging Instances and Databases" on page 51.

7. Have the user of each system synchronize.
   a. You may want to run the synchronizer application in test mode, by running the **db2sync -t** command to ensure that it is setup to synchronize prior to the first production synchronization.

      Note: Running the synchronization application in test mode will not cause the fix batch to run.

   b. Have the system user run the synchronizer application (in production mode), by running the **db2sync** command.

   After the fix batch has executed successfully, or produced satisfactory results, the migrated satellite is ready to run group batches. To have it start executing group batches, select the "Promote" action for the satellite, set its execution starting point as required and click on **OK**.

   Note: If the fix batch produced satisfactory results but still failed in execution, you will need to enable the satellite for batch in addition to promoting it.

The system will now execute group batches the next time it synchronizes.

Note: The information in the "Migrating from Previous Versions of DB2" on page 214 should be reviewed before beginning the process of migrating systems to DB2 Satellite Edition, and enabling these systems for group batch execution.

Migrating from Previous Versions of DB2

This section describes the process of migrating an existing system to DB2 Satellite Edition from previous releases. The following releases are supported for Version 6 migration: DB2 Common Server Version 2.x, Database Server Version 4.x, and DB2 Universal Database Version 5.x.

Note: The migration process for Database Server Version 4 is identical to that used for DB2 Common Server Version 2. Whenever Version 2 is mentioned in this section, the same information also applies to Version 4.

DB2 migration involves the following procedures:

- "Pre-Installation Migration Steps"
- "Post-Installation Steps" on page 219.

Pre-Installation Migration Steps

These steps will help you to ensure that all databases on your system can be migrated to the new DB2 Version 6 format. Before installing your new version of DB2, perform the following steps:

Step 1. Back up Databases

You should back up all databases before installing your new version of DB2. To back up the databases, perform the following steps:

1. Complete all database transactions.
2. Ensure all applications disconnect from each database.

   To list all applications that are connected to a database, enter the **db2 list applications** command. If all applications are disconnected, this command will return the following message:

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

   To force all applications to disconnect from the database, enter the **db2 force applications all** command.
3. Ensure all databases are cataloged. To view a list of all the cataloged databases in the current instance, enter the following command:

   ```
   db2 list database directory
   ```
4. Make a backup copy of all databases. For more information on backing up databases, refer to the *Administration Guide* for the DB2 version you are backing up. For the syntax of the backup command, refer to the *Command Reference* for the DB2 version you are backing up.

Note: Make sure that this is the most recent backup copy of the database before you start the next procedure.

5. Stop the database manager by entering the **db2stop** command.

Step 2. Verify that Databases Can Be Migrated

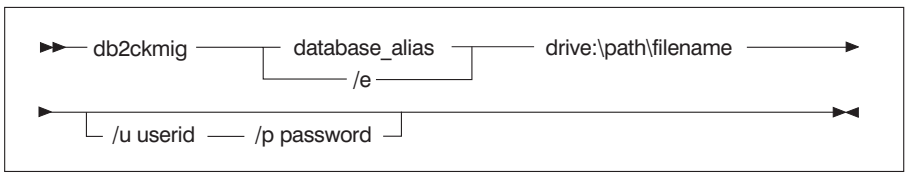
DB2 provides the **db2ckmig** command to check that databases can be migrated. This command must be run prior to installation. The command is located on the product CD-ROM.

To run the **db2ckmig** command:

1. Insert the CD-ROM into the drive.

   Note: If you are installing DB2 on a Windows 9x or Windows NT workstation, the setup program might be started automatically using the operating system's auto-run feature. In this case, do not proceed with the install. Instead, cancel and proceed to the next step.

2. Enter the **db2ckmig** command to verify that the databases on your system can be migrated. The syntax of the command is as follows:



database_alias Specifies a *database_alias* name of a database to be verified for migration. This parameter is required if the **/e** parameter is not specified.

/e Specifies that all cataloged databases are to be verified for migration. This parameter is required if the *database_alias* parameter is not specified.

/l *drive:\path\filename*
Specifies a drive, target path and filename to keep a list of errors and warnings generated for the scanned database. The *path* variable is optional; if you do not specify a path, the path from which you execute the **db2ckmig** command will be used. There is no default when installing from the CD-ROM. You must specify a *filename*.

/u *userid* Specifies the user account used to connect to the database. This parameter must be specified if the **/p** parameter is specified.

/p *password* Specifies the password of the user account used to connect to the database. This parameter must be specified if the **/u** parameter is specified.

For example, to check that all databases cataloged on your system can be migrated and to log all the messages from this command to the c:\temp\message.txt file, enter the following command:

```
x:\db2\common\db2ckmig /e /l c:\temp\message.txt
```

where `x:` represents your CD-ROM drive.

3. If any errors are found, the **db2ckmig** command generates a log file and places it in the path and file specified by the `/l` option. When the errors are corrected, enter the **db2ckmig** command again to ensure that the databases are ready to be migrated.

Table 4. Correcting Error Messages

Error	Action
A database is in backup pending state	Perform a backup of the database.
A database is in roll-forward pending state	Recover the database as required. Perform or resume a roll-forward database to end of logs and stop.
Table space ID is not in normal state	Recover the database and table space as required. Perform or resume a roll-forward database to end of logs and stop.
A database is in an inconsistent state	Restart the database to return it to a consistent state.
The Version 2 database contains database objects that have a schema name of SYSCAT, SYSSTAT, or SYSFUN	These schema names are reserved for the Version 6 database manager. To correct this error, perform the following steps: 1. Back up the database. 2. Export the data from the database object (catalogs or tables). 3. Drop the object. 4. Re-create the object with another schema name. 5. Import/Load the data into the object. 6. Run the **db2ckmig** command against the database again, ensuring that the database passes the **db2ckmig** check. 7. Make a backup copy of the database. For more information, refer to the *Administration Guide*.

Table 4. Correcting Error Messages (continued)

Error	Action
The Version 2 database contains database objects that have a dependency on the SYSFUN.DIFFERENCE function. Possible violated database objects are: • Constraint • Function • Trigger • View	The SYSFUN.DIFFERENCE function must be dropped and re-created during database migration. However, if there is a database object that is dependent on this function, migration will fail. To correct this error: **Constraint** Enter the **alter table** command to drop the constraint. **Function** Enter the **drop function** command to drop the function dependent on SYSFUN.DIFFERENCE. **Trigger** Enter the **drop trigger** command to drop the trigger. **View** Enter the **drop view** command to drop the view. **Note:** Any package dependent on the SYSFUN.DIFFERENCE function will be marked inoperative after migration. As a result, the **db2ckmig** command will not report any package that is dependent on the SYSFUN.DIFFERENCE function.For more information, refer to the *Administration Guide*.
The database contains user-defined distinct types (UDTs) that use the type name BIGINT, DATALINK, REAL or REFERENCE.	These data type names are reserved for the Version 6 database manager. To correct this error, perform the following steps: 1. Back up the database. 2. Export the data from any tables that are dependent on the data types. 3. Drop any tables dependent on the data types, and then drop the data types. These drops may drop other objects such as views, indexes, triggers, or functions. 4. Create data types with different type names and re-create the tables using the new data type names. Re-create any dropped views, indexes, triggers, or functions. 5. Import/Load the data into the object. 6. Run the **db2ckmig** command against the database again, ensuring that the database passes the **db2ckmig** check. 7. Make a backup copy of the database. For more information, refer to the *Administration Guide*.

Table 4. Correcting Error Messages (continued)

Error	Action
Structured type and function have the same name.	A structured type and function (with no arguments) belonging to the same schema cannot have the same name. The type or function and objects using the type or function have to dropped and re-created using another name. To correct this error, perform the following steps: 1. Back up the database. 2. Export the data from any tables that are dependent on the structured types or functions. 3. Drop any tables dependent on the structured types or functions, and then drop the structured types or functions. These drops may drop other objects such as views, indexes, triggers, or functions. 4. Create structured types or functions with different type or function names and re-create the tables using the new data type or function names. Re-create any dropped views, indexes, triggers, or functions. 5. Import/Load the data into the object. 6. Run the **db2ckmig** command against the database again, ensuring that the database passes the **db2ckmig** check. 7. Make a backup copy of the database. For more information, refer to the *Administration Guide*.

Migration Considerations for the DB2 Version 2.x User Exit Program:

Note: These instructions apply only to the DB2 Version 2.x **db2uexit** user exit program. If you are not using the Version 2.x **db2uexit** user exit program, skip this section and continue at "Installing DB2 Satellite Edition Version 6" on page 219.

DB2 Version 6 uses the **db2uexit** user exit program to archive and retrieve log files. For more information on the **db2uexit** interfaces, refer to the *Administration Guide*.

The following should be considered before migrating from Version 2.x to Version 6.

- If the Version 2.x user exit program, **db2uexit.exe**, located in the \sqllib\bin directory before installation, it will remain in this directory after the installation completes. The **db2uext2.exe** program will also be installed in this directory. Its function is to invoke the **db2uexit.cmd** or **db2uexit.exe** user exit programs using the Version 2.x interface. This allows the old user exit program to be used on Version 6.

- If **db2uexit.exe** is in a directory other than the sqllib\bin directory, it will remain there after installation, but **db2uext2.exe** will not be installed in the sqllib\bin directory. Following installation, if you want to use the old user exit program, you will have to copy it to the sqllib\bin directory, then copy **db2uext2.v2** from the sqllib\misc directory to the sqllib\bin directory, and rename it to **db2uext2.exe**.

If you are migrating from DB2 Version 2.x, you should modify your user exit program to use the DB2 Version 6 interfaces. The new user exit program **db2uexit** should replace **db2uext2** in the sqllib\bin directory.

Installing DB2 Satellite Edition Version 6

After you have successfully completed the pre-installation checks, you can now start installing DB2 Satellite Edition Version 6 using either the interactive or distributed method as described in "Interactive Installation or Distributed Installation" on page 188. During the installation of DB2 Satellite Edition Version 6, instance migration for instances created in previous versions of DB2 occurs.

Post-Installation Steps

After installing DB2 Version 6, you can now migrate databases and complete other migration activities. Do not make any changes to the database system before migrating all the databases; otherwise, the database migration will fail.

Migrating Databases

To migrate pre-Version 6 databases owned by an instance, perform the following steps:

1. Log in with a user account that has SYSADM authority.
2. Ensure that the databases you want to migrate are cataloged. To retrieve a list of all catalogued databases on your system, enter the **db2 list database directory** command.
3. Migrate the database using the **db2 migrate database** command. For more information, refer to the *Command Reference*.

Optional Post Migration Activities

After database migration is complete, you can perform the following post-migration activities:

- Migrate Unique Indexes
- Update Statistics
- Rebind Packages
- Update Database and Database Manager Configuration

- Migrate Explain Tables

You can also apply these activities to a back-level database backup that is restored to Version 6, since at the end of the restore the database is migrated to Version 6.

Migrate Unique Indexes (db2uiddl)

DB2 Versions 5 and 6 support deferred checking for duplicate index key values until the end of UPDATE statements. This ensures that temporary duplicate index key values which *may* be present in mid-UPDATE, but no longer are present at the end of the UPDATE, will not cause the statement to fail.

Note: You must install the Client Tools sub-component of the Miscellaneous Tools to use the **db2uiddl** command. Client Tools component can only be installed during a custom install.

With DB2 Version 2 the same UPDATE statement may fail because checking for duplicate key index values is performed row by row as the statement processes the table. For example, if a row with value 1 is changed to value 2, but a row with value 2 already exists, a duplicate value 2 will be detected causing the DB2 Version 2 UPDATE statement to fail.

Note: Version 2.x and 5.x unique indexes are not automatically migrated to Version 6 semantics for the following reasons:

- Converting unique indexes is a time-consuming operation.
- You may have applications that depend on the previous version's unique index semantics.
- You may want to manage the staged conversion of unique indexes on your own schedule, when needed, using the **db2uiddl** command.

All existing applications will continue to work even if the unique indexes are not converted to Version 6 semantics. You have to convert unique indexes to Version 6 semantics only if support for deferred uniqueness checking is required.

To convert unique indexes, you need to perform the following steps:

1. Log in with a user account that has SYSADM authority.
2. Start the database manager by entering the **db2start** command.
3. Run the **db2uiddl** command against your migrated database. Refer to the *Command Reference* for the syntax of this command.

   The **db2uiddl** command searches the database catalog tables and generates all the CREATE UNIQUE INDEX statements for user tables in an output file.

4. Review the output generated from the **db2uiddl** command. You should remove any unwanted indexes from the output file to reduce the time needed to execute it. Comments in the output will flag other situations that require your attention.
5. Connect to the database by entering the **db2 connect to** *database_alias* command, where *database_alias* is the alias of the database you are migrating.
6. Execute the output file, generated by the **db2uiddl** command, as a DB2 CLP command file, using a command similar to the following:

   ```
   db2 -tvf filename
   ```

   where `filename` represents the file generated by the **db2uiddl** command.

   Note: DB2 interprets the re-creation of an existing unique index using the **db2uiddl** command to signal that the index is ready to be converted to Version 6 semantics.
7. Disconnect from the database by entering the **db2 connect reset** command.

Update Statistics

When database migration is completed, the old statistics that are used to optimize query performance are retained in the catalogs. However, Version 6 of DB2 has statistics that are modified or do not exist in Versions 2.x or 5.x. To uses these statistics for better application performance, you may want to execute the **runstats** command on tables, particularly those tables that are critical to the performance of your SQL queries.

Refer to the *Command Reference* for the syntax of the **runstats** command. For details on the statistics, refer to the *Administration Guide*.

Rebind Packages

During database migration, all existing packages are invalidated. After the migration process, each package is rebuilt when it is used for the first time by the Version 6 database manager.

For better performance, you should run the **db2rbind** command to rebuild all packages stored in the database. In DB2 Version 6 this command has a new option, **all**, which, when specified, rebinds all packages (valid and invalid). If the **all** option is not specified with the **db2rbind** command, only those packages marked as invalid are rebound. Refer to the *Command Reference* for the syntax of this command.

Update Database and Database Manager Configuration

Some of the database configuration parameters are changed to Version 6 defaults or to other values during database migration. The same is true for database manager configuration parameters which may have changed to Version 6 defaults or to other values. For more information about configuration parameters, refer to the *Administration Guide*.

Note: You should run the DB2 Performance Monitor for suggestions in choosing appropriate configuration parameters. For more information about using the DB2 Performance Monitor, refer to *System Monitor Guide and Reference*.

Migrate Explain Tables

To migrate the explain tables in a database that has been migrated to Version 6, run the following command:

```
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 to be migrated. This parameter is required.
- *userid* and *password* represent the current user's ID and password. These parameters are optional.

Note: You must install the Database Tools sub-component of the Miscellaneous Tools to use the **db2exmig** command. Database Tools component can only be installed during a custom installation.

The explain tables belonging to the user ID that is running **db2exmig**, or that is used to connect to the database, will be migrated. The explain tables migration tool will rename the Version 2 or Version 5 tables, create a new set of tables, using the **EXPLAIN.DDL**, and copy the contents of the old tables to the new tables. Finally, it will drop the old tables. The migration utility, **db2exmig**, will preserve any user added columns on the explain tables.

Part 3. Appendixes

© Copyright IBM Corp. 1999

Appendix A. How the DB2 Library Is Structured

The DB2 Universal Database library consists of SmartGuides, online help, books and sample programs in HTML format. This section describes the information that is provided, and how to access it.

To access product information online, you can use the Information Center. You can view task information, DB2 books, troubleshooting information, sample programs, and DB2 information on the Web. See "Accessing Information with the Information Center" on page 236 for details.

Completing Tasks with SmartGuides

SmartGuides help you complete some administration tasks by taking you through each task one step at a time. SmartGuides are available through the Control Center and the Client Configuration Assistant. The following table lists the SmartGuides.

Note: Create Database, Index, and Configure Multisite Update SmartGuide are available for the partitioned database environment.

SmartGuide	Helps You to...	How to Access...
Add Database	Catalog a database on a client workstation.	From the Client Configuration Assistant, click **Add**.
Back up Database	Determine, create, and schedule a backup plan.	From the Control Center, click with the right mouse button on the database you want to back up and select **Backup->Database using SmartGuide**.
Configure Multisite Update SmartGuide	Perform a multi-site update, a distributed transaction, or a two-phase commit.	From the Control Center, click with the right mouse button on the **Database** icon and select **Multisite Update**.
Create Database	Create a database, and perform some basic configuration tasks.	From the Control Center, click with the right mouse button on the **Databases** icon and select **Create->Database using SmartGuide**.

© Copyright IBM Corp. 1999

SmartGuide	Helps You to...	How to Access...
Create Table	Select basic data types, and create a primary key for the table.	From the Control Center, click with the right mouse button on the **Tables** icon and select **Create->Table using SmartGuide**.
Create Table Space	Create a new table space.	From the Control Center, click with the right mouse button on the **Table spaces** icon and select **Create->Table space using SmartGuide**.
Index	Advise which indexes to create and drop for all your queries.	From the Control Center, click with the right mouse button on the **Index** icon and select **Create->Index using SmartGuide**.
Performance Configuration	Tune the performance of a database by updating configuration parameters to match your business requirements.	From the Control Center, click with the right mouse button on the database you want to tune and select **Configure using SmartGuide**.
Restore Database	Recover a database after a failure. It helps you understand which backup to use, and which logs to replay.	From the Control Center, click with the right mouse button on the database you want to restore and select **Restore->Database using SmartGuide**.

Accessing Online Help

Online help is available with all DB2 components. The following table describes the various types of help. You can also access DB2 information through the Information Center. For information see "Accessing Information with the Information Center" on page 236.

Type of Help	Contents	How to Access...
Command Help	Explains the syntax of commands in the command line processor.	From the command line processor in interactive mode, enter: ? *command* where *command* is a keyword or the entire command. For example, `? catalog` displays help for all the CATALOG commands, while `? catalog database` displays help for the CATALOG DATABASE command.

Type of Help	Contents	How to Access...
Control Center Help *Client Configuration Assistant Help* *Event Analyzer Help* *Command Center Help*	Explains the tasks you can perform in a window or notebook. The help includes prerequisite information you need to know, and describes how to use the window or notebook controls.	From a window or notebook, click the **Help** push button or press the F1 key.
Message Help	Describes the cause of a message, and any action you should take.	From the command line processor in interactive mode, enter: `?` *XXXnnnnn* where *XXXnnnnn* is a valid message identifier. For example, `? SQL30081` displays help about the SQL30081 message. To view message help one screen at a time, enter: `?` *XXXnnnnn* `| more` To save message help in a file, enter: `?` *XXXnnnnn* `>` *filename.ext* where *filename.ext* is the file where you want to save the message help.
SQL Help	Explains the syntax of SQL statements.	From the command line processor in interactive mode, enter: `help` *statement* where *statement* is an SQL statement. For example, **help** SELECT displays help about the SELECT statement. **Note:** SQL help is not available on UNIX-based platforms.
SQLSTATE Help	Explains SQL states and class codes.	From the command line processor in interactive mode, enter: **?** *sqlstate* or **?** *class-code* where *sqlstate* is a valid five-digit SQL state and *class-code* is the first two digits of the SQL state. For example, `? 08003` displays help for the 08003 SQL state, while `? 08` displays help for the 08 class code.

DB2 Information – Hardcopy and Online

The table in this section lists the DB2 books. They are divided into two groups:

Cross-platform books
These books contain the common DB2 information for all platforms.

Platform-specific books
These books are for DB2 on a specific platform. For example, there are separate *Quick Beginnings* books for DB2 on OS/2, on Windows NT, and on the UNIX-based platforms.

Cross-platform sample programs in HTML
These samples are the HTML version of the sample programs that are installed with the SDK. They are for informational purposes and do not replace the actual programs.

Most books are available in HTML and PostScript format, or you can choose to order a hardcopy from IBM. The exceptions are noted in the table.

On OS/2 and Windows platforms, HTML documentation files can be installed under the doc\html subdirectory. Depending on the language of your system, some files may be in that language, and the remainder are in English.

On UNIX platforms, you can install multiple language versions of the HTML documentation files under the doc/%L/html subdirectories. Any documentation that is not available in a national language is shown in English.

You can obtain DB2 books and access information in a variety of different ways:

View See "Viewing Online Information" on page 235.

Search See "Searching Online Information" on page 238.

Print See "Printing the PostScript Books" on page 238.

Order See "Ordering the Printed Books" on page 239.

Name	Description	Form Number File Name for Online Book	HTML Directory
	Cross-Platform Books		

Name	Description	Form Number File Name for Online Book	HTML Directory
Administration Guide	*Administration Guide, Design and Implementation* contains information required to design, implement, and maintain a database. It also describes database access using the Control Center(whether local or in a client/server environment), auditing, database recovery, distributed database support, and high availability. *Administration Guide, Performance* contains information that focuses on the database environment, such as application performance evaluation and tuning. You can order both volumes of the *Administration Guide* in the English language in North America using the form number SBOF-8922.	Volume 1 SC09-2839 db2d1x60 Volume 2 SC09-2840 db2d2x60	db2d0
Administrative API Reference	Describes the DB2 application programming interfaces (APIs) and data structures you can use to manage your databases. Explains how to call APIs from your applications.	SC09-2841 db2b0x60	db2b0
Application Building Guide	Provides environment setup information and step-by-step instructions about how to compile, link, and run DB2 applications on Windows, OS/2, and UNIX-based platforms. This book combines the *Building Applications* books for the OS/2, Windows, and UNIX-based environments.	SC09-2842 db2axx60	db2ax
APPC, CPI-C and SNA Sense Codes	Provides general information about APPC, CPI-C, and SNA sense codes that you may encounter when using DB2 Universal Database products. **Note:** Available in HTML format only.	No form number db2apx60	db2ap

Name	Description	Form Number File Name for Online Book	HTML Directory
Application Development Guide	Explains how to develop applications that access DB2 databases using embedded SQL or JDBC, how to write stored procedures, user-defined types, user-defined functions, and how to use triggers. It also discusses programming techniques and performance considerations. This book was formerly known as the *Embedded SQL Programming Guide*.	SC09-2845 db2a0x60	db2a0
CLI Guide and Reference	Explains how to develop applications that access DB2 databases using the DB2 Call Level Interface, a callable SQL interface that is compatible with the Microsoft ODBC specification.	SC09-2843 db2l0x60	db2l0
Command Reference	Explains how to use the command line processor, and describes the DB2 commands you can use to manage your database.	SC09-2844 db2n0x60	db2n0
Data Movement Utilities Guide and Reference	Explains how to use the Load, Import, Export, Autoloader, and Data Propogation utilities to work with the data in the database.	SC09-2858 db2dmx60	db2dm
DB2 Connect Personal Edition Quick Beginnings	Provides planning, installing, and configuring information for DB2 Connect Personal Edition.	GC09-2830 db2c1x60	db2c1
DB2 Connect User's Guide	Provides concepts, programming and general usage information about the DB2 Connect products.	SC09-2838 db2c0x60	db2c0
Connectivity Supplement	Provides setup and reference information on how to use DB2 for AS/400, DB2 for OS/390, DB2 for MVS, or DB2 for VM as DRDA application requesters with DB2 Universal Database servers, and on how to use DRDA application servers with DB2 Connect application requesters. **Note:** Available in HTML and PostScript formats only.	No form number db2h1x60	db2h1
Glossary	Provides a comprehensive list of all DB2 terms and definitions. **Note:** Available in HTML format only.	No form number db2t0x50	db2t0

Name	Description	Form Number File Name for Online Book	HTML Directory
Installation and Configuration Supplement	Guides you through the planning, installation, and set up of platform-specific DB2 clients. This supplement contains information on binding, setting up client and server communications, DB2 GUI tools, DRDA AS, distributed installation, and the configuration of distributed requests and access methods to heterogeneous data sources.	GC09-2857 db2iyx60	db2iy
Message Reference	Lists messages and codes issued by DB2, and describes the actions you should take.	GC09-2846 db2m0x60	db2m0
Replication Guide and Reference	Provides planning, configuration, administration, and usage information for the IBM Replication tools supplied with DB2.	SC26-9642 db2e0x60	db2e0
SQL Getting Started	Introduces SQL concepts, and provides examples for many constructs and tasks.	SC09-2856 db2y0x60	db2y0
SQL Reference, Volume 1 and Volume 2	Describes SQL syntax, semantics, and the rules of the language. Also includes information about release-to-release incompatibilities, product limits, and catalog views. You can order both volumes of the *SQL Reference* in the English language in North America with the form number SBOF-8923.	SBOF-8923 Volume 1 db2s1x60 Volume 2 db2s2x60	db2s0
System Monitor Guide and Reference	Describes how to collect different kinds of information about databases and the database manager. Explains how to use the information to understand database activity, improve performance, and determine the cause of problems.	SC09-2849 db2f0x60	db2f0
Troubleshooting Guide	Helps you determine the source of errors, recover from problems, and use diagnostic tools in consultation with DB2 Customer Service.	S10J-8169	db2p0

Name	Description	Form Number File Name for Online Book	HTML Directory
What's New	Describes the new features, functions, and enhancements in DB2 Universal Database, Version 6.0, including information about Java-based tools.	SC09-2851 db2q0x60	db2q0
	Platform-Specific Books		
Administering Satellites Guide and Reference	Provides planning, configuration, administration, and usage information for satellites.	GC09-2821 db2dsx60	db2ds
DB2 Personal Edition Quick Beginnings	Provides planning, installation, migration, and configuration information for DB2 Universal Database Personal Edition on the OS/2, Windows 95, and Windows NT operating systems.	GC09-2831 db2i1x60	db2i1
DB2 for OS/2 Quick Beginnings	Provides planning, installation, migration, and configuration information for DB2 Universal Database on the OS/2 operating system. Also contains installing and setup information for many supported clients.	GC09-2834 db2i2x60	db2i2
DB2 for UNIX Quick Beginnings	Provides planning, installation, migration, and configuration information for DB2 Universal Database on UNIX-based platforms. Also contains installing and setup information for many supported clients.	GC09-2836 db2ixx60	db2ix
DB2 for Windows NT Quick Beginnings	Provides planning, installation, migration, and configuration information for DB2 Universal Database on the Windows NT operating system. Also contains installing and setup information for many supported clients.	GC09-2835 db2i6x60	db2i6
DB2 Enterprise - Extended Edition for UNIX Quick Beginnings	Provides planning, installation, and configuration information for DB2 Enterprise - Extended Edition for UNIX. Also contains installing and setup information for many supported clients.	GC09-2832 db2v3x60	db2v3

Name	Description	Form Number File Name for Online Book	HTML Directory
DB2 Enterprise - Extended Edition for Windows NT Quick Beginnings	Provides planning, installation, and configuration information for DB2 Enterprise - Extended Edition for Windows NT. Also contains installing and setup information for many supported clients.	GC09-2833 db2v6x60	db2v6
DB2 Connect Enterprise Edition for OS/2 and Windows NT Quick Beginnings	Provides planning, migration, installation, and configuration information for DB2 Connect Enterprise Edition on the OS/2 and Windows NT operating systems. Also contains installation and setup information for many supported clients. This book was formerly part of the *DB2 Connect Enterprise Edition Quick Beginnings.*	GC09-2828 db2c6x60	db2c6
DB2 Connect Enterprise Edition for UNIX Quick Beginnings	Provides planning, migration, installation, configuration, and usage information for DB2 Connect Enterprise Edition in UNIX-based platforms. Also contains installation and setup information for many supported clients. This book was formerly part of the *DB2 Connect Enterprise Edition Quick Beginnings.*	GC09-2829 db2cyx60	db2cy
DB2 Data Links Manager for AIX Quick Beginnings	Provides planning, installation, configuration, and task information for DB2 Data Links Manager for AIX.	GC09-2837 db2z0x60	db2z0
DB2 Data Links Manager for Windows NT Quick Beginnings	Provides planning, installation, configuration, and task information for DB2 Data Links Manager for Windows NT.	GC09-2827 db2z6x60	db2z6
DB2 Query Patroller Administration Guide	Provides administration information on DB2 Query Patrol.	SC09-2859 db2dwx60	db2dw
DB2 Query Patroller Installation Guide	Provides installation information on DB2 Query Patrol.	GC09-2860 db2iwx60	db2iw
DB2 Query Patroller User's Guide	Describes how to use the tools and functions of the DB2 Query Patrol.	SC09-2861 db2wwx60	db2ww

Name	Description	Form Number File Name for Online Book	HTML Directory
Cross-Platform Sample Programs in HTML			
Sample programs in HTML	Provides the sample programs in HTML format for the programming languages on all platforms supported by DB2 for informational purposes (not all samples are available in all languages). Only available when the SDK is installed. See *Application Building Guide* for more information on the actual programs. **Note:** Available in HTML format only.	No form number	db2hs/c db2hs/cli db2hs/clp db2hs/cpp db2hs/cobol db2hs/cobol_mf db2hs/fortran db2hs/java db2hs/rexx

Notes:

1. The character in the sixth position of the file name indicates the language of a book. For example, the file name `db2d0e60` indicates that the *Administration Guide* is in English. The following letters are used in the file names to indicate the language of a book:

Language	Identifier
Brazilian Portuguese	b
Bulgarian	u
Czech	x
Danish	d
Dutch	q
English	e
Finnish	y
French	f
German	g
Greek	a
Hungarian	h
Italian	i
Japanese	j
Korean	k
Norwegian	n
Polish	p
Portuguese	v
Russian	r
Simp. Chinese	c
Slovenian	l
Spanish	z
Swedish	s

Trad. Chinese	t
Turkish	m

2. For late breaking information that could not be included in the DB2 books:
   - On UNIX-based platforms, see the Release.Notes file. This file is located in the DB2DIR/Readme/%L directory, where %L is the locale name and DB2DIR is:
     - /usr/lpp/db2_06_01 on AIX
     - /opt/IBMdb2/V6.1 on HP-UX, Solaris, SCO UnixWare 7, and Silicon Graphics IRIX
     - /usr/IBMdb2/V6.1 on Linux.
   - On other platforms, see the RELEASE.TXT file. This file is located in the directory where the product is installed.
   - Under Windows `Start` menu

Viewing Online Information

The manuals included with this product are in Hypertext Markup Language (HTML) softcopy format. Softcopy format enables you to search or browse the information, and provides hypertext links to related information. It also makes it easier to share the library across your site.

You can view the online books or sample programs with any browser that conforms to HTML Version 3.2 specifications.

To view online books or sample programs on all platforms other than SCO UnixWare 7:

- If you are running DB2 administration tools, use the Information Center. See "Accessing Information with the Information Center" on page 236 for details.
- Select the `Open Page` menu item of your Web browser. The page you open contains descriptions of and links to DB2 information:
  - On UNIX-based platforms, open the following page:

    ```
    file:/INSTHOME/sqllib/doc/%L/html/index.htm
    ```

    where *%L* is the locale name.
  - On other platforms, open the following page:

    ```
    sqllib\doc\html\index.htm
    ```

    The path is located on the drive where DB2 is installed.

If you have not installed the Information Center, you can open the page by double-clicking on the **DB2 Online Books** icon. Depending on the system you are using, the icon is in the main product folder or the Windows Start menu.

To view online books or sample programs on the SCO UnixWare 7:

- DB2 Universal Database for SCO UnixWare 7 uses the native SCOhelp utility to search the DB2 information. You can access SCOhelp by the following methods:
  - entering the "scohelp" command on the command line,
  - selecting the `Help` menu in the Control Panel of the CDE desktop or
  - selecting `Help` in the Root menu of the Panorama desktop

  For more information on SCOhelp, refer to the *Installation and Configuration Supplement*.

Accessing Information with the Information Center

The Information Center provides quick access to DB2 product information. The Information Center is available on all platforms on which the DB2 administration tools are available.

Depending on your system, you can access the Information Center from the:

- Main product folder
- Toolbar in the Control Center
- Windows Start menu
- Help menu of the Control Center

The Information Center provides the following kinds of information. Click the appropriate tab to look at the information:

Tasks	Lists tasks you can perform using DB2.
Reference	Lists DB2 reference information, such as keywords, commands, and APIs.
Books	Lists DB2 books.
Troubleshooting	Lists categories of error messages and their recovery actions.
Sample Programs	Lists sample programs that come with the DB2 Software Developer's Kit. If the Software Developer's Kit is not installed, this tab is not displayed.
Web	Lists DB2 information on the World Wide

Web. To access this information, you must have a connection to the Web from your system.

When you select an item in one of the lists, the Information Center launches a viewer to display the information. The viewer might be the system help viewer, an editor, or a Web browser, depending on the kind of information you select.

The Information Center provides some search capabilities, so you can look for specific topics, and filter capabilities to limit the scope of your searches.

For a full text search, click the `Search` button of the Information Center follow the *Search DB2 Books* link in each HTML file.

The HTML search server is usually started automatically. If a search in the HTML information does not work, you may have to start the search server by double-clicking its icon on the Windows or OS/2 desktop.

Refer to the release notes if you experience any other problems when searching the HTML information.

Note: Search function is not available in the Linux and Silicon Graphics environments.

Setting Up a Document Server

By default, the DB2 information is installed on your local system. This means that each person who needs access to the DB2 information must install the same files. To have the DB2 information stored in a single location, use the following instructions:

1. Copy all files and subdirectories from \sqllib\doc\html on your local system to a Web server. Each book has its own subdirectory containing all the necessary HTML and GIF files that make up the book. Ensure that the directory structure remains the same.
2. Configure the Web server to look for the files in the new location. For information, see the NetQuestion Appendix in *Installation and Configuration Supplement*.
3. If you are using the Java version of the Information Center, you can specify a base URL for all HTML files. You should use the URL for the list of books.
4. Once you are able to view the book files, you should bookmark commonly viewed topics. Among those, you will probably want to bookmark the following pages:

- List of books
- Tables of contents of frequently used books
- Frequently referenced articles, such as the *ALTER TABLE* topic
- The Search form

For information about setting up a search, see the NetQuestion Appendix in *Installation and Configuration Supplement* book.

Searching Online Information

To search for information in the HTML books, you can do the following:

- Click on **Search the DB2 Books** at the bottom of any page in the HTML books. Use the search form to find a specific topic. This function is not available in the Linux or Silicon Graphics IRIX environments.
- Click on **Index** at the bottom of any page in an HTML book. Use the index to find a specific topic in the book.
- Display the table of contents or index of the HTML book, and then use the find function of the Web browser to find a specific topic in the book.
- Use the bookmark function of the Web browser to quickly return to a specific topic.
- Use the search function of the Information Center to find specific topics. See "Accessing Information with the Information Center" on page 236 for details.

Printing the PostScript Books

If you prefer to have printed copies of the manuals, you can decompress and print PostScript versions. For the file name of each book in the library, see the table in "DB2 Information – Hardcopy and Online" on page 228. Specify the full path name for the file you intend to print.

On OS/2 and Windows platforms:

1. Copy the compressed PostScript files to a hard drive on your system. The files have a file extension of .exe and are located in the x:\doc*language*\books\ps directory, where x: is the letter representing the CD-ROM drive and *language* is the two-character country code that represents your language (for example, EN for English).
2. Decompress the file that corresponds to the book that you want. Each compressed book is a self-extracting executable file. To decompress the

book, simply run it as you would run any other executable program. The result from this step is a printable PostScript file with a file extension of .ps.

3. Ensure that your default printer is a PostScript printer capable of printing Level 1 (or equivalent) files.
4. Enter the following command from a command line:

   ```
   print filename.ps
   ```

On UNIX-based platforms:

1. Mount the CD-ROM. Refer to your *Quick Beginnings* manual for the procedures to mount the CD-ROM.
2. Change to /cdrom/doc/%L/ps directory on the CD-ROM, where */cdrom* is the mount point of the CD-ROM and *%L* is the name of the desired locale. The manuals will be installed in the previously-mentioned directory with file names ending with .ps.Z.
3. Decompress and print the manual you require using the following command:
   - For AIX:

     ```
     zcat filename | qprt -P PSPrinter_queue
     ```

   - For HP-UX, Solaris, or SCO UnixWare 7:

     ```
     zcat filename | lp -d PSPrinter_queue
     ```

   - For Linux:

     ```
     zcat filename | lpr -P PSPrinter_queue
     ```

   - For Silicon Graphics IRIX:

     ```
     zcat < filename | lp -d PSPrinter_queue
     ```

   where *filename* is the full path name and extension of the compressed PostScript file and *PSprinter_queue* is the name of the PostScript printer queue.

   For example, to print the English version of *DB2 for UNIX Quick Beginnings* on AIX, you can use the following command:

   ```
   zcat /cdrom/doc/en/ps/db2ixe60.ps.Z | qprt -P ps1
   ```

Ordering the Printed Books

You can order the printed DB2 manuals either as a set or individually. There are two sets of books available. The form number for the entire set of DB2 books is SBOF-8926-00. The form number for the books listed under the heading "Cross-Platform Books" is SBOF-8924-00.

Note: These form numbers only apply if you are ordering books that are printed in the English language in North America.

You can also order books individually by the form number listed in "DB2 Information – Hardcopy and Online" on page 228. To order printed versions, contact your IBM authorized dealer or marketing representative, or phone `1-800-879-2755` in the United States or `1-800-IBM-4YOU` in Canada.

Appendix B. DB2 Satellite Edition Differences

Before you migrate DB2 Personal Edition Version 5 to DB2 Satellite Edition Version 6, you should review the following list of functions and features that are either not supported on DB2 Satellite Edition, or are unique to DB2 Satellite Edition. Reviewing this list should help you ensure that your applications will continue to operate.

DB2 Satellite Edition does not provide the following:

- Communications using APPC, IPX/SPX, NetBios, and NamedPipes. Only TCP/IP is supported.
- DRDA Application Server functionality for DRDA Application Requesters.
- A DB2 Administration Server (DAS). If you want to use the Control Center to administer a satellite, not all of the Control Center functionality is available. For more information, see "Appendix D. Administering DB2 Satellite Edition from the Control Center" on page 245.
  - Satellites cannot be discovered using the DB2 discovery function. To connect to a satellite, you must create the node and database directory entries for that satellite using the Control Center, the Client Configuration Assistant, or manually, using the CLP.
  - You cannot schedule scripts for execution on a satellite.
  - You cannot use the browse function of a remote Control Center to view the satellite's directory structure or files.
  - You cannot grant or revoke privileges on a satellite from the Control Center.
  - You cannot use the full function of some SmartGuides.
- The DB2 graphical user interface tools, including:
  - The Control Center
  - The Client Configuration Assistant
  - The Event Analyzer
  - The Command Center
  - The Information Center.

    Note: If you require these tools on a satellite, you can install the DB2 UDB Administrative Client, in addition to DB2 Satellite Edition, on the satellite.
- The ability to create the SAMPLE database.
- The DB2 Governor.

© Copyright IBM Corp. 1999

- The coding samples that are typically used by database administrators or systems staff.
- Lightweight Directory Access Protocol (LDAP).
- Distributed Computing Environment (DCE).
- Simple Network Management Protocol (SNMP).
- DB2 documentation in HTML, PostScript, or online help.
- Remote monitoring.

The following functions are specific to DB2 Satellite Edition:

- Administration from the script-based centralized satellite administration solution. This solution includes the Satellite Administration Center.
- The **asnsat** command that is used to call the Capture and Apply programs in DB2 data replication.
- The DB2 Synchronizer application.

Some default configuration parameter values are different for DB2 Satellite Edition than for other DB2 Universal Database instances and databases. For the configuration parameters and their DB2 Satellite Edition-specific values, refer to the *Administration Guide*. Configuration parameters that apply to DB2 Satellite Edition are identified by Satellite Database Server with local clients in the table at the beginning of each relevant parameter.

Appendix C. db2sync - Start DB2 Synchronizer

Starts a graphical application that can be used to start, stop, and monitor the progress of a synchronization session for a satellite. This command can be invoked in test mode, wherein the ability of a satellite to synchronize can be validated.

This command can also be used to either set or display the application version for a satellite.

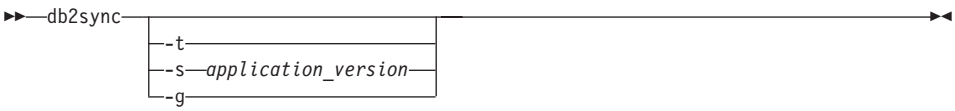
Authorization

None

Required Connection

None

Command Syntax



Command Parameters

-t Specifies that the synchronizer is to be started in test mode, in which the ability of a satellite to synchronize with its DB2 control server can be verified.

-s *application_version*
Sets the application version on the satellite to *application_version*.

-g Displays the application version currently set on the satellite.

© Copyright IBM Corp. 1999

Appendix D. Administering DB2 Satellite Edition from the Control Center

DB2 Satellite Edition does not include a DB2 Administration Server (DAS). Because of this, and for other reasons, if you use the Control Center to administer a DB2 Satellite Edition system, not all of the Control Center functionality is available. The functions that follow cannot be used. All other functions of the Control Center can be used on DB2 Satellite Edition for administration, and for problem determination and correction.

Notes:

1. You can only use the Control Center to administer a satellite system if you select the inbound remote communications component when you install the satellite.
2. For late breaking information on this subject, refer to the following Web site:

   `www.software.ibm.com/data/db2/library/satellite`

General Restrictions

Job Scheduling is not available.

Function that requires information from the file system on the satellite is not available. For example, pull-down lists for drive letters are not available. In most situations, you can enter information for drive, path, and directory data manually.

General Folder and Object Restrictions

The following actions are not available when they appear in the pop-up menu for a folder or for an object:

- Performance monitoring cannot be performed on any object.
- When DB2 Satellite Edition is running on Windows NT, you can select **Privileges** from a pop-up menu to both view the current user and group privileges and to update them, but the **Add User** and **Add Group** push buttons of the Privileges notebook do not work.

© Copyright IBM Corp. 1999

Specific Folder and Object Restrictions

On the Add System window, you cannot use the **Retrieve** and **Refresh** push buttons to add a satellite to the Control Center.

On the Add Instance and the Add Database windows, you cannot use the **Refresh** push button to add a satellite to the Control Center.

You cannot perform the following actions from the pop-up menu for a named instance:

- Start
- Stop
- Import a server profile
- Setup communications
- Multisite Update
- Applications.

From a database folder, you cannot use the Create Database SmartGuide to create a database that uses Database Managed Storage (DMS) objects.

You cannot perform the following actions from the pop-up menu for a named database:

- Configure using SmartGuide
- Backup Database using SmartGuide
- Generate DDL.

You cannot perform Generate DDL from the pop-up menu for a named table.

You cannot perform Create Index using SmartGuide from the pop-up menu for an index folder.

You cannot open the Connections folder for a database.

Appendix E. Naming Rules

Go to the section that describes the naming rules that you require information on:

General Naming Rules

Unless otherwise specified, all names can include the following characters:

- A through Z

  Note: When used in most names, characters A through Z are converted from lowercase to uppercase.
- 0 through 9
- @, #, $, and _ (underscore)

Unless otherwise specified, all names must begin with one of the following characters:

- A through Z
- @, #, and $

Do not use SQL reserved words to name tables, views, columns, indexes, or authorization IDs.

For a list of SQL reserved words, refer to *SQL Reference*.

Database, Database Alias, and Catalog Node Name Rules

Database names are the identifying names assigned to databases in the database manager.

© Copyright IBM Corp. 1999

Database alias names are synonyms given to remote databases. Database aliases must be unique within the System Database Directory in which all aliases are stored.

When naming a database or database alias, see "General Naming Rules" on page 247.

In addition, the name you specify can *only* contain 1 to 8 characters.

Note: To avoid potential problems, do not use the special characters @, #, and $ in a database name if you intend to have a client remotely connect to a host database. Also, because these characters are not common to all keyboards, do not use them if you plan to use the database in another country.

On Windows NT systems, ensure that no instance name is the same as a service name.

Object Name Rules

Database objects include:

- Tables
- Views
- Columns
- Indexes
- User-defined functions (UDFs)
- User-defined types (UDTs)
- Triggers
- Aliases
- Table spaces
- Schemas

When naming database objects, see "General Naming Rules" on page 247.

In addition, the name you specify:

- Can contain 1 to 18 characters *except* for the following:
  - Table names (including view names, summary table names, alias names, and correlation names), which can contain up to 128 characters; and
  - column names, which can contain up to 30 characters

Note: The names of all tables and columns that are used for data replication can contain up to 18 characters.

- Cannot be any of the SQL reserved words that are listed in the *SQL Reference*.

Note: Using delimited identifiers, it is possible to create an object that violates these naming rules; however, subsequent use of the object could result in errors.

For example, if you create a column with a + or – sign included in the name and you subsequently use that column in an index, you will experience problems when you attempt to reorganize the table. To avoid potential problems with the use and operation of your database, *do not* violate these rules.

Username, User ID, Group Name, and Instance Name Rules

Usernames or *User IDs* are the identifiers assigned to individual users. When naming users, groups, or instances, see "General Naming Rules" on page 247.

In addition, the name you specify:

- Can contain 1 to 8 characters
- Cannot be any of the following:
  - USERS
  - ADMINS
  - GUESTS
  - PUBLIC
  - LOCAL
- Cannot begin with:
  - IBM
  - SQL
  - SYS
- Cannot include accented characters
- In general, when naming users, groups, or instances:

  UNIX Use lowercase names.

  Windows 32-bit operating systems
  Use any case.

Workstation Name (nname) Rules

A *workstation* name specifies the NetBIOS name for a database server or satellite that resides on the local workstation. This name is stored in the database manager configuration file. The workstation name is known as the *workstation nname*. When naming workstations, see "General Naming Rules" on page 247.

In addition, the name you specify:

- Can contain 1 to 8 characters
- Cannot include &, #, and @
- Must be unique within the network

DB2SYSTEM Naming Rules

DB2 uses the *DB2SYSTEM* name to identify a physical DB2 machine, system, or workstation within a network. On Windows 32-bit operating systems, you do not need to specify a *DB2SYSTEM* name; the DB2 setup program detects the NT Computer name and assigns it to *DB2SYSTEM*.

When creating a *DB2SYSTEM* name, see "General Naming Rules" on page 247.

In addition, the name you specify:

- Must be unique within a network
- Can contain a maximum of 21 characters

Password Rules

When determining passwords, consider the following rules:

UNIX A maximum of 8 characters.

Windows 9x or Windows NT
A maximum of 14 characters.

Appendix F. National Language Support (NLS)

This section contains information about the National Language Support (NLS) provided by DB2, including information about supported languages and code pages. For information on developing applications that use NLS, refer to the *Application Development Guide*.

Code Page and Language Support

During installation of DB2, the country, codepage, and regional settings are established. However, you can change these settings after installing DB2: including regional settings such as code page, country language (for monetary, date, and numeric formatting), and time zone. When a new connection to a database is made, the database manager uses these new values.

Note: You must ensure that your regional settings are set correctly. DB2 may not produce the expected results if the country, code page, or regional settings are incorrect for the intended language.

Table 5 shows the languages into which the DB2 messages are translated.

Note: The code page values in the table that follows are also used as directory names on DB2 CD-ROMs. For example, a reference to *x*:*language*\win32\install would be *x*:\en\win32\install for English. For more detailed information on the languages and code pages support, refer to the *Administration Guide*.

Table 5. Languages and Code Pages

Country Code	Language
bg	Bulgarian
br	Brazilian Portuguese
cn	Simplified Chinese (PRC)
cz	Czech
de	German
dk	Danish
en	English
es	Spanish
fi	Finnish
fr	French

© Copyright IBM Corp. 1999

Table 5. Languages and Code Pages (continued)

Country Code	Language
gr	Greek
hu	Hungarian
il	Hebrew
it	Italian
jp	Japanese
kr	Korean
nl	Dutch
no	Norwegian
pl	Polish
pt	Portuguese
ru	Russian
se	Swedish
si	Slovenian
tr	Turkish
tw	Traditional Chinese (Taiwan)

Appendix G. Notices

Any reference to an IBM licensed program in this publication is not intended to state or imply that only IBM's licensed program may be used. Any functionally equivalent product, program or service that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program, or service. Evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, is the user's responsibility.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the

IBM Director of Licensing
IBM Corporation, North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Canada Limited
Office of the Lab Director
1150 Eglinton Ave. East
North York, Ontario
M3C 1H7
CANADA

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

This publication may contain examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

© Copyright IBM Corp. 1999

Trademarks

The following terms are trademarks or registered trademarks of the IBM Corporation in the United States and/or other countries:

ACF/VTAM
ADSTAR
AISPO
AIX
AIXwindows
AnyNet
APPN
AS/400
CICS
C Set++
C/370
DATABASE 2
DataHub
DataJoiner
DataPropagator
DataRefresher
DB2
DB2 Connect
DB2 Universal Database
Distributed Relational Database Architecture
DRDA
Extended Services
FFST
First Failure Support Technology
IBM
IMS
LAN Distance
MVS/ESA
MVS/XA
OS/400
OS/390
OS/2
PowerPC
QMF
RACF
RISC System/6000
SP
SQL/DS
SQL/400
S/370
System/370
System/390
SystemView
VisualAge
VM/ESA
VSE/ESA
VTAM
WIN-OS/2

Trademarks of Other Companies

The following terms are trademarks or registered trademarks of the companies listed:

C-bus is a trademark of Corollary, Inc. in the United States, other countries, or both.

HP-UX is a trademark of Hewlett-Packard.

Java, HotJava, Solaris, Solstice, and Sun are trademarks of Sun Microsystems, Inc.

Microsoft, Windows, Windows NT, Visual Basic, and the Windows logo are trademarks or registered trademarks of Microsoft Corporation in the United States, other countries, or both.

PC Direct is a trademark of Ziff Communications Company in the United States, other countries, or both and is used by IBM Corporation under license.

ActionMedia, LANDesk, MMX, Pentium, and ProShare are trademarks of Intel Corporation in the United States, other countries, or both.

UNIX is a registered trademark in the United States, other countries or both and is licensed exclusively through X/Open Company Limited.

Other company, product, or service names, which may be denoted by a double asterisk (**), may be trademarks or service marks of others.

Index

A

© Copyright IBM Corp. 1999

E

F

G

I

L

M

O

P

Q

R

S

T

U

V

W

Contacting IBM

This section lists ways you can get more information from IBM.

If you have a technical problem, please take the time to review and carry out the actions suggested by the *Troubleshooting Guide* before contacting DB2 Customer Support. Depending on the nature of your problem or concern, this guide will suggest information you can gather to help us to serve you better.

For information or to order any of the DB2 Universal Database products contact an IBM representative at a local branch office or contact any authorized IBM software remarketer.

Telephone

If you live in the U.S.A., call one of the following numbers:

- 1-800-237-5511 to learn about available service options.
- 1-800-IBM-CALL (1-800-426-2255) or 1-800-3IBM-OS2 (1-800-342-6672) to order products or get general information.
- 1-800-879-2755 to order publications.

For information on how to contact IBM outside of the United States, see Appendix A of the IBM Software Support Handbook. You can access this document by accessing the following page:

`http://www.ibm.com/support/`

then performing a search using the keyword "handbook".

Note that in some countries, IBM-authorized dealers should contact their dealer support structure instead of the IBM Support Center.

World Wide Web

http://www.software.ibm.com/data/
http://www.software.ibm.com/data/db2/library/

The DB2 World Wide Web pages provide current DB2 information about news, product descriptions, education schedules, and more. The DB2 Product and Service Technical Library provides access to frequently asked questions, fixes, books, and up-to-date DB2 technical information. (Note that this information may be in English only.)

Anonymous FTP Sites

ftp.software.ibm.com

© Copyright IBM Corp. 1999

Log on as anonymous. In the directory /ps/products/db2, you can find demos, fixes, information, and tools concerning DB2 and many related products.

Internet Newsgroups

comp.databases.ibm-db2, bit.listserv.db2-l

These newsgroups are available for users to discuss their experiences with DB2 products.

CompuServe

GO IBMDB2 to access the IBM DB2 Family forums

All DB2 products are supported through these forums.

To find out about the IBM Professional Certification Program for DB2 Universal Database, go to http://www.software.ibm.com/data/db2/db2tech/db2cert.html

Part Number: CT644NA

Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.

CT644NA

GC09-2821-00