CA RC/Update™ for DB2 for z/OS

User Guide

Version 17.0.00, Fourth Edition

CA Technologies Product References

This document references the following CA Technologies products:

- CA Database Analyzer™ for DB2 for z/OS (CA Database Analyzer)
- CA Fast Check® for DB2 for z/OS (CA Fast Check)
- CA Fast Load for DB2 for z/OS (CA Fast Load)
- CA Fast Recover™ for DB2 for z/OS (CA Fast Recover)
- CA Fast Unload® for DB2 for z/OS (CA Fast Unload)
- CA Quick Copy for DB2 for z/OS (CA Quick Copy)
- CA Rapid Reorg® for DB2 for z/OS (CA Rapid Reorg)
- CA RC/Migrator™ for DB2 for z/OS (CA RC/Migrator)
- CA RC/Query® for DB2 for z/OS (CA RC/Query)
- CA RC/Update™ for DB2 for z/OS (CA RC/Update)

Contact CA Technologies

Contact CA Support

For your convenience, CA Technologies provides one site where you can access the information that you need for your Home Office, Small Business, and Enterprise CA Technologies products. At http://ca.com/support, you can access the following resources:

- Online and telephone contact information for technical assistance and customer services
- Information about user communities and forums
- Product and documentation downloads
- CA Support policies and guidelines
- Other helpful resources appropriate for your product

Providing Feedback About Product Documentation

If you have comments or questions about CA Technologies product documentation, you can send a message to techpubs@ca.com.

To provide feedback about CA Technologies product documentation, complete our short customer survey which is available on the CA Support website at http://ca.com/docs.

Documentation Changes

The following documentation updates have been made in the fourth edition:

- Temporal Table Support (see page 377)—Updated the information about using temporal tables with unique constraints.

The following documentation updates have been made in the third edition:

- Temporal Table Support (see page 377)—Updated the information about using temporal tables with indexes and added an example.

The following documentation updates have been made in the second edition:

- Temporal Table Support (see page 377)—Updated the information about using temporal tables.

The following documentation updates have been made since the last release of this documentation:

- Known Issues that were previously documented in the Release Notes are now documented in the "Operational Considerations" chapter.
- Temporal Table Support (see page 377)—Added this new topic that describes considerations for implementing temporal tables.
- RC/Edit Component (see page 117)—Added support for TIMESTAMP with TIMEZONE.
- RC/Browse Component (see page 193)—Added support for TIMESTAMP with TIMEZONE.

Contents

Chapter 6: Drop Recovery 103

Chapter 7: RC/Edit 117

Chapter 9: RC/Browse 193

Chapter 10: RC/Copy 203

Chapter 11: Data Compare 217

Chapter 1: Introduction

This section contains the following topics:

Overview of CA RC/Update

CA RC/Update is a DB2 object and data management tool that lets you effectively manage and maintain DB2 objects and data, as well as support the application development area.

With CA RC/Update, you can:

- Create and maintain DB2 objects
- Edit, browse, and copy DB2 data
- Compare data between tables or between the same tables at different times
- Test embedded SQL in programs
- Execute SQL in batch or online mode
- Recover information contained in dropped DB2 objects

A complete application development environment is provided for the application developer, an editor and data copy feature for the end user, and sophisticated object management facilities for the database administrator. It is a multi-purpose tool that satisfies a variety of users.

Components

CA RC/Update is composed of the following components:

- RC/Objects (see page 24)
- RC/Alter (see page 24)
- Drop Recovery (see page 103)
- RC/Edit
- RI Editor (see page 25)
- RC/Browse

- RC/Copy (see page 203)
- ISQL (see page 25)
- Batch Processor (see page 25)
- Data Compare (see page 217)
- Join/Edit (see page 25)
- Space Calculator (see page 33)

RC/Objects Component

The RC/Objects component is a comprehensive facility for creating, altering, and dropping DB2 objects. Online screens are provided for each function and an online help and tutorial system is always available. The template option can be used to create objects based on existing objects. Supported DB2 object options include referential integrity, secondary IDs, and aliases. The generated DDL can be executed in online mode, batch mode, or RC/Alter mode.

RC/Alter Component

The RC/Alter component is automatically invoked whenever an object alteration not supported by DB2 is requested. DB2 permits only certain alterations to be made to a DB2 object using the DB2 ALTER command. For example, DB2 does not permit changing a column data type, inserting a new column anywhere other than the end of a table definition, or deleting a column from an existing table. RC/Alter creates the job to unload all the necessary data, drops the object, re-creates the object with the new definition, and then re-creates all dependent objects and authorization chains with no user intervention. It also allows recovery of the original version and form of an altered object.

RC/Objects automatically determines if the requested alterations to an object can be performed using the DB2 ALTER command. If they cannot, RC/Objects automatically invokes the RC/Alter facility to perform a drop and recreation of the object with the specified changes.

This is a very powerful feature and is transparent to the end user, who only makes the change. An impact analysis report can be reviewed to determine whether to implement the alteration.

To make a non-DB2 supported change to an object, the object must be dropped and re-created. Without RC/Alter, when an object is dropped, all data, its dependents, and authorizations are lost.

RI Editor Component

RI Editor makes it simpler to edit tables that have referential integrity relationships. With this component, all of the editing operations and commands available with RC/Edit are available, plus the timesaving features for working with multiple tables of a referential integrity (RI) set. The simulation of a referential integrity relationship between the current table and another table based on the unique constraint of the current table can also be done.

ISQL Component

ISQL is a facility for executing SQL statements contained within a data set. While using the ISPF editor, any lines within the data set can be designated for execution. A screen appears prompting for any host variables and the extracted SQL statements can then be executed in batch or online mode.

Note: For more information about the SQL Editor, see the *CA Database Management Solutions for DB2 for z/OS General Facilities Reference Guide*.

Batch Processor

The Batch Processor executes SQL statements, DB2 utilities, IDCAMS requests, and application programs in batch mode (not TSO batch) or interactively. A full audit trail is produced, and users can also request to perform authorization ID switching for creating objects for other users. The Batch Processor is also fully restartable.

Join/Edit Component

Join/Edit simplifies the setting up of SQL queries that incorporate joins. These queries can be made “on-the-fly” or stored in DB2 views for later processing.

Benefits

The benefits of CA RC/Update include the following:

Increases Productivity

With RC/Objects, DB2 objects are easily created and maintained using online screens. There is no need to remember object names, options, and DDL syntax. There are extensive space estimation and VSAM facilities available for defining the correct space allocations and underlying data sets.

Enhances DB2 Availability

When making a non-DB2 supported change to a DB2 object, the object must be dropped, re-created with a new definition, its dependent objects and authorization chains re-created, and then its data reloaded. This process is error-prone and difficult.

The RC/Alter component automatically performs the necessary steps to make the changes and re-create the object with all its data and dependents in place. This significantly reduces the time the DB2 objects are unavailable.

Lowers Development Costs

Using the ISQL feature, SQL can be tested from within the standard ISPF editor. This reduces the time, cost, and effort required to develop DB2 application programs. The RC/Edit and RC/Copy components can also easily populate test and production tables with data. The Batch Processor can execute whole job streams of SQL, DB2 utilities, and application programs in true batch mode.

Reduces Database Administrator Workload

Using the RC/Objects and RC/Alter components, the DB2 database administrator workload is significantly reduced. The advanced Space Calculation features for tablespaces and indexspaces help the database administrator estimate the necessary space requirements. The VSAM Define Facility automatically performs any explicit VSAM defines.

Promotes DB2 Usage

Users can become apprehensive when confronted with new and sophisticated technology. Users and management can quickly use DB2 and see the benefits. New users can easily browse and edit data using RC/Edit and RC/Copy, and can readily create tables using the Table Create and Template options.

Logs Activity

All activity within CA RC/Update is logged. Reports can be produced from the log for reviewing and auditing object creation, alteration, and deletion.

Separates Responsibilities

EXECUTE authority on the CA RC/Update plan is required. All DDL requests can be generated without holding the necessary authorities to execute the DDL. The data set with the DDL can be reviewed and then executed by another user with the appropriate authorities, using the Batch Processor. This provides a separation of responsibilities: request, approval, and implementation.

Helps Ensure Recoverability

CA RC/Update provides full recoverability from object alterations and drops. The Alteration Recovery feature allows recovery of the original version and form of an object altered through RC/Alter. The Recoverable Drop option in RC/Drop enables the recovery of a dropped object and its dependents by means of information stored in a recovery table.

Chapter 2: Operational Considerations

This section contains the following topics:

Product Authorization

You can grant or revoke authorization to the following CA RC/Update functions:

Product Auths

Grants user the authority to use CA RC/Update.

Auth. Options

Enter one of the following to browse, update or copy additional product authorizations for a user:

- E—Browse
- Y—Update
- C—Copy

Note: For specific instructions, see the *General Facilities Reference Guide.*

Set Authorities for a Default User

To create a default user ID that has limited access to CA RC/Update options, use Product Authorizations. For example, to give a group of users access to the RC/Browse option, but not RC/Edit, use the CA RC/Update Auth Option.

Follow these steps:

1. Enter **A** in the Option line on the CA Database Management Solutions Main Menu.

   The Product Authorizations panel appears.

2. Make the following specifications:

   a. Specify the user ID that represents all default users in the To ID field.

   b. Type **E** in the CMD field next to RC/Update.

   Press Enter.

   Two subordinate fields appear below the RC/Update line.

3. Enter **Y** in the Auth. Options field.

   The CA RC/Update Authorization panel appears.

4. Indicate which options the default user is not allowed to access, by entering **N** in the option fields.

   Note: The default entry for all fields on this panel is Y.

   Access settings now reflect your choices.

Copy Authority Options from One User to Another User

You can copy authority options from one user to another user.

Follow these steps:

1. Enter the user ID you want to copy authorities from in the From ID field on the Product Authorizations screen, and then enter the user ID you want to copy authorities to in the To ID field.

2. Enter **C** in the Cmd field next to the CA RC/Update Auth. Options function.

   The authority options are copied to the user ID in the To ID field.

Subcomponents

The following describes the CA RC/Update subcomponents:

RC/Edit

Lets the user edit table data.

RC/Browse

Lets the user view the table data, but not edit the data.

RC/SQL

Lets the user enter and execute SQL.

RC/Copy

Lets the user copy table data to another table.

RC/Data Compare

Lets the user compare data from two tables.

Administrative Privileges

The following privileges indicate the user's authority to view and change object definition defaults. These privileges are shared with CA RC/Migrator.

System Administrator

Lets the user view and change object definition defaults for any user.

Personal Defaults

Lets the user view and change object definition defaults for the user's ID only. This is the default.

Restrict Delete Function

Indicates what objects a user can delete from the recovery table. The recovery table stores the DDL for a recoverable dropped object. Once the recoverable DDL is deleted, the object and its dependents cannot be recovered. Valid values are:

Y

Indicates that the user can delete recovery DDL for only objects dropped by the user. Authority for this option is the default.

N

Indicates that the user can delete the DDL from the recovery table for any object.

A user with SYSADM authority has automatic authority to delete any object's recoverable DDL.

Model Service Privilege

Indicates whether the user can access Model Services to create, update, or delete the model IDs or model members used to generate Batch Processor statements for utilities.

Note: To execute the Batch Processor, the user must also be granted EXECUTE authority on the Batch Processor. For more information, see the *CA Database Management Solutions for DB2 for z/OS General Facilities Reference Guide*.

Valid values are:

Y

Activates Model Services. The user can select the Model Services option from the CA RC/Update Profile Menu.

N

Deactivates Model Services. The Model Services option does not appear on the CA RC/Update Profile Menu.

Other Authorizations

To execute the Batch Processor, you must grant the user authority to execute the Batch Processor (found in the General category).

Note: For more information, see the *CA Database Management Solutions for DB2 for z/OS General Facilities Reference Guide*.

Space Calculator Component

The Space Calculator facility can calculate the primary and secondary space quantities needed for your tablespaces and indexes. This eliminates time-consuming calculations.

The Space Calculator facility is especially important to users who want to calculate the space needed for segmented tablespaces and for multiple tables within one tablespace. Multiple table calculations require an average row size to calculate accurately the total space needed for non-segmented tablespaces. The average row size approximates multiple tables' row length to compensate for the rows being interleaved on one page.

Note: For more information about Space Calculator, see the *CA Database Management Solutions for DB2 for z/OS General Facilities Reference Guide*.

DB2 9 Clone Support Limitations

The RC/Object interface in CA RC/Update does not provide capabilities to Alter a Table to Add or Drop a clone. These operations can be performed in ISQL by entering the appropriate DDL commands. If a change is made to the base table that requires it to be dropped and recreated, the clone is preserved. RC/Alter preserves the clone, its data, and any objects or security defined against the clone and its dependent objects.

To add or drop a clone table, enter the appropriate DDL commands in ISQL.

Trusted Context Support

Trusted context and roles support is provided for alter, create, template, and drop options.

An internal error occurs when attempting to alter or template a trigger, sequence, or stored procedure under a trusted context.

To circumvent this problem:

1. Enter the **PROF** primary command on the CA RC/Update Main Menu.
2. Select option **4** (RC/Object Profile Variables).
3. Set the Current SQLID Confirmation Pop-Up option to **N**.
4. Press F3 until you return to the CA Database Management Solutionsfor DB2 for z/OS Main Menu.
5. Select option **3** (RC/Update) and alter or template the object again.

Index on Expression Toleration Limitations

Index on expression (IOE) toleration support has the following limitations:

- You cannot explicitly change an IOE into another kind of index by using the Index Alter or Index Template panels.
- You cannot explicitly create an IOE by using the Index Create panel. To create an IOE, you must template an existing IOE.
- There is no MATCHCASE or CAPS support for the expression fields. The fields are always maintained in mixed case.

User-Defined Distinct Types

You cannot create, alter, or drop user-defined distinct types (UDTs).

LOB Support Limitations

When the Batch Processor interface is called to unload or load tables with LOB objects, the size of the LOB data cannot exceed 32 KB.

User-Defined Functions

You cannot create, alter, and drop user-defined functions (UDFs).

UPDATE TRIGGER Column Considerations

When tables are updated during a typical edit session, the dynamic SQL UPDATE statement includes all table columns. This can cause an UPDATE TRIGGER column to appear to be updated even if it was not explicitly modified. This action will fire the trigger, which may be of concern to you.

Exceptions to this processing are as follows:

- ROWID and IDENTITY (generated always) columns are always excluded from the update.
- ROWID and IDENTITY (generated by default) columns are excluded only if the column is not updated.
- LOB columns are excluded from the update.

- If EQF was used to exclude the column, the column is not included in the update.
- PROTECT command listed columns are excluded from the update.

Security Label Restrictions

The following restrictions apply when you create and alter tables with columns defined with the security label attribute (AS SECURITY LABEL):

- The column must not be part of a unique constraint.
- The table must not have editproc, validproc, audit, obid, data capture, ccsid, volatile or notvolatile checked.

  Note: SQL error -20240 is generated if errors occur.
- Ensure that the user executing the product does not have authority (explicitly or through the .AUTH command) to drop and recreate a table without the appropriate security label when the table data is unloaded and loaded again.

  Note: The .AUTH command does not affect the security label.

To create a table with a security label column, specify **L** in the D column on the CA Table Create panel.

DB2 Routines Restriction

CA RC/Update does not support procedures that are created by off-platform tools, such as Data Studio from IBM, that introduce NL or NF characters in the text. A sample error message follows:

```
R0603E CSMTEXT ERROR: MOD=ROAPRAQ2 TYPE=PARSE
      R15-R1=0000000C 00000030 00000190
```

Double-Byte Character Set Support

Double-byte character set (DBCS) support is provided. DBCS uses two bytes per character.

As part of the support of DBCS, you can edit graphic-type columns. If you use the EXPLODE command, the ISPF editor is called. You can edit one column of a row at a time.

Note: When you first try to EXPLODE on a graphic-type column, the ISRFORM member of your ISPF profile is updated with the RCEDBCS format provided in *hlq*.CDBATBL0(RCEFORM). Do not change the RCEDBCS format. You can browse this format using the Format Utility option on the ISPF menu.

Chapter 3: Security

This section contains the following topics:

DB2 Security

CA RC/Update follows the same security considerations as DB2. Except where noted, if the DDL statement is embedded in a program, the privilege set consists of the privileges designated by the owner of the plan. If the statement is executed dynamically, the privilege set is the union of the privileges designated by each authorization ID of the process (primary and secondary IDs.)

Important! CA RC/Update never lets you bypass DB2 security. It provides additional security through the authorization exit and Product Authorization facilities.

Drop Security

The following apply for drop security:

- Users can drop a DB2 table, tablespace, or index if they have:
  - Ownership (created) of the object
  - SYSADM authority
  - SYSCTRL authority
  - DBADM authority on the database that contains the object
- Users can drop a DB2 storage group, alias, or view if they have:
  - Ownership (created) of the object
  - SYSADM authority
  - SYSCTRL authority

- Users can drop a database if they have:
  - SYSADM authority
  - SYSCTRL authority
  - DBADM or DBCTRL authority for the database
  - DROP privilege on the database
- Users can drop a synonym if they have ownership (created) of the synonym.
  - For static SQL (in a plan), the synonym must be owned by the owner of the plan.
  - For dynamic SQL, the synonym must be owned by the current SQL ID (a primary ID or one of several secondary IDs).
  - An authorization ID with SYSADM authority can drop synonyms on behalf of other users by changing the value of the CURRENT SQLID register. A SYSADM user can change the SQLID to any user ID.

Alter Security

The following apply for alter security:

- Users can alter an object if they created the object or have:
  - The ALTER privilege
  - SYSADM or SYSCTRL authority
  - DBADM authority on the database that contains the object (does not apply to storage groups)
- Users can also alter a database if they have DBCTRL or DROP authority on the database.
- If you are altering a table by dropping the parent key, the ALTER privilege must be held on all dependent tables. If a table is altered by dropping the foreign key, the ALTER privilege must be held on the parent tables. SYSADM or SYSCTRL authority is needed to alter ROSHARE.

Create and Template Security

The following apply for create and template security:

- Users can create any DB2 object except for an alias or a view if they have:
  - SYSADM or SYSCTRL authority
  - DBADM authority on the database that will contain the object (does not apply to storage groups because they are outside of the database)
  - The correct CREATE privilege for the object, as follows:
    - CREATEDBC or CREATEDBA privilege to create databases
    - CREATESG privilege to create storage groups
    - CREATETAB privilege on the database to create tables
    - CREATETS privilege on the database to create a tablespace
    - INDEX privilege or creator of the table to create an index on a table
- Users can create a view if they have SYSADM or SYSCTRL authority or SELECT privilege on every table or view named.
- Users can create an alias if they have SYSADM or SYSCTRL authority or a CREATEALIAS privilege on the database.

Note: No authorizations are needed to create a synonym. Tables or tablespaces can also be created in a database if the user has DBMAINT or DBCTRL authority on the database.

RC/Copy Security

Users can copy table data if they have SELECT authority for the table from which they are copying; or have insert authority for the table into which they are copying.

RC/Edit Security

Actions within an edit session are controlled by the authorities held on the table. The security restrictions are placed on update, column update, insert, delete, select, and referential integrity authority.

- Update—If a user does not hold UPDATE authority, all data will be protected on the edit screens, except for data in newly inserted rows (if the user holds INSERT authority on the table).
- Column Update—If users have specific column UPDATE authorizations, they will be allowed to edit data in those columns, but all other columns will be protected. Newly inserted rows will have all fields unprotected (if the user holds INSERT authority).

- Insert—Users without INSERT authority for a table will not be permitted to perform the line commands I (Insert) or R (Replicate).
- Delete—Users without DELETE authority will not be permitted to issue the D (Delete) line command, but will be allowed to delete rows inserted during the edit session.
- Select—Users without SELECT authority will not be permitted to access the table with RC/Edit.
- Referential Integrity—The unique constraint of a table with a parent key defined is automatically protected within the edit session so it cannot be updated.

RC/Browse Security

Actions within a browse session are controlled by the authorities held on the table. With RC/Browse, users without select authority are not permitted to access a table.

ISQL Security

The ISQL component uses a separate plan from CA RC/Update. However, when granting execute authority to CA RC/Update, execute authority is granted on both plans. The required security for executing ISQL is based on the statements entered. If there is no DB2 authority to execute the entered statements, an SQL error panel appears.

Batch Processor Security

To perform an alter using the RC/Alter facility, users might need all the authorities described in the *CA Database Management Solutions for DB2 for z/OS Batch Processor Reference Guide* because RC/Alter can execute the authorized or restricted commands and programs, such as the load utility, statistics collection, and REBIND, as part of the alteration.

The authorities required for ALTER are also needed, as described previously in Alter Security, to execute the DDL contained within the RC/Alter job.

Note: For more information, see the *CA Database Management Solutions for DB2 for z/OS Batch Processor Reference Guide*.

External Security Considerations

You can use an exit point that DB2 uses when processing security requests through an external security system, such as CA ACF2 or IBM RACF. If this exit point has been activated, RC/Edit is unable to determine SELECT, INSERT, UPDATE, and DELETE authority. Additionally, RC/Alter might be unable to create and alter objects.

Authorization IDs

Authorization IDs play a major role because they determine the creator of the object. Extensive facilities are provided to help ensure correct object creators during online or batch execution.

- Primary ID—Whenever a connection is established between DB2 and a process (CA RC/Update), DB2 uses the TSO logon ID or the USER parameter on the job statement as the authorization ID. This authorization ID is called the primary ID. Every process has exactly one primary ID.
- Secondary ID—DB2 passes the primary ID to an authorization exit. The exit can return a list of one or more authorization IDs, and this list of IDs constitutes the authorization IDs of the process. However, as stated earlier, there is only one primary ID. Any other process authorization IDs are called secondary IDs. A primary ID can have a maximum of 255 secondary IDs.

At any time, the SQL authorization ID of a process (CA RC/Update) is the value of a special register called CURRENT SQLID. This special register can be initialized by the authorization exit. If it is not, its initial value is the primary ID.

Manually Changing the SQLID

The value of CURRENT SQLID can be changed by the execution of a SET CURRENT SQLID statement. Unless some authorization ID of the process has SYSADM authority, the new ID must be one of the authorization IDs of the process. Thus, CURRENT SQLID usually contains either the primary ID or one of its secondary IDs. The SQLID value is saved in the user's ISPF Profile.

RC/Objects issues a SET statement each time users enter the product so that the current SQL ID equals the SQL ID saved in the last session. Use the SET CURRENT SQLID command to change the value to a different authorization ID.

Generated SQLIDs

A SET CURRENT SQLID statement is generated along with an object's creation DDL at generation time. When the generated DDL is executed through the Batch Processor, the Batch Processor establishes a new thread to DB2. The value of this new thread's authorization ID initially equals the user's primary authorization ID.

The SET CURRENT SQLID statement is generated because the user's primary ID might not have the proper authorizations to create the object. The SET CURRENT SQLID statement will be generated according to the following guidelines.

- For objects not having explicit creators (database, storage group, tablespace, synonym), RC/Objects generates a SET CURRENT SQLID statement to set the current SQL ID to the value in the Creator field on the product screen.
- For objects with explicit creators (table, view, index, alias), RC/Objects generates a SET CURRENT SQLID statement equal to the value of the user's current SQL ID. The command, SET CURRENT SQLID authid, is different from the default SQLID option that can be specified for an object's recovery (drop or alter recovery).
- SET CURRENT SQLID—When the SET CURRENT SQLID statement is issued in a CA product session, the product checks the DB2 authorizations against the user's current SQL ID. The authorizations for a user's current SQL ID are checked during the RC/Edit sessions. For objects with explicit creators, a SET CURRENT SQLID statement is also generated before any CREATE statements are generated during the session.
- Default SQLID—The Default SQLID recovery option generates SET CURRENT SQLID statements before the CREATE statements for objects with explicit creators in an object's recovery file. If a default SQLID is not specified, no SET CURRENT SQLID statements are issued.

AUTH Command

The Batch Processor AUTH command dynamically changes the authorization ID of the process. Once the authorization ID has been changed by the AUTH command, DB2 considers it the new primary ID. The AUTH command also requires that the Batch Processor be installed in an APF library.

Note: For more information, see the “Operational Considerations“ chapter.

This command can be used online and in batch mode.

Note: For a complete description of this command, see the *CA Database Management Solutions for DB2 for z/OS Batch Processor Reference Guide*.

Authorization ID Summary

The object owner is the value of the current SQLID and the object creator is the primary ID.

Object Owner

The following authorization ID statement is generated for the object owner. A SET CURRENT SQLID is issued for the ID entered in the creator column of the object for objects with implicit creators. A SET CURRENT SQLID is issued equal to the value of the current SQL ID for objects with explicit creators. DB2 stores the object owner in the CREATOR field of the object.

Object Creator

The following authorization ID statements are generated for the object creator.

- For online mode, no special statements are issued. The current user ID is the creator ID.
- For batch mode, an AUTH statement is created for the current user ID. Regardless of who executes the batch data set, the correct authorization ID is used when the object is created.

In both instances, DB2 stores the object creator in the CREATEDBY field of the object. The following example illustrates the commands issued to create a tablespace in batch mode.

.AUTH CREATEDBY

Sets up the primary ID for the batch execution.

SET CURRENT SQLID

Generated automatically so the correct Creator ID is used.

CREATE OBJECT DDL

The actual CREATE statement.

.AUTH

Resets authorization ID.

SET CURRENT SQLID is generated because a tablespace is an object with an implicit creator. Databases, synonyms, and storage groups also have implicit creators. Objects with explicit creators (aliases, views, tables, and indexes) have a SET CURRENT SQLID generated based on the value of the current SQL ID.

Special Considerations

In addition to the previous information, note the following special considerations:

- For objects that permit the object owner to be specified as part of the DDL (tables, indexes), a SET CURRENT SQLID statement is generated that is equal to the value of the current SQL ID.
- DB2 does not keep the CREATEDBY column for views. Therefore, there is no way to find out who created the views. The creation of the view is done under the authority of the user executing the DDL (batch or online).
- When an object is altered through RC/Alter, the original CREATEDBY ID is maintained.

If a user is generating the batch DDL for a different authorized user to execute, the AUTH statement will not switch to the other user's ID. In this case, the submitter of the job should edit the batch data set and perform a global change on the AUTH command to change it to their ID.

Chapter 4: Using CA RC/Update

This section contains the following topics:

Main Menu

The Main Menu displays available options, object codes, table data options, and operation control options. Although the main menu is the only screen that lists the options available, options and objects can be selected from any screen that includes the standard header. You can return to the main menu from any screen by entering the MAIN command.

The only difference between the main menu and any other screen that contains the standard header is that the main menu provides a map to the other available options.

Note: For information about the fields, press F1 (Help).

Request a Subsystem ID Listing

You can access a listing of remote locations and available subsystem IDs from the CA Database Management Solutions for DB2 for z/OS main menu, and from any product screen displaying a Location or DB2 SSID field.

To request a subsystem ID listing, enter a question mark (**?**) or blanks in the SSID field, and press Enter.

DB2 SSID/Location Selection Panel

The DB2 SSID/Location selection panel displays a list of remote locations, available subsystem IDs, load libraries, and install system administrators (SYSADM) for those subsystems.

Note: The values that appear on this screen are specified in the SETUP member of *high-level*.CDBAPARM. For more information, see the “Operational Considerations“ chapter.

Enter **S** in the selection line to select a local or remote SSID/location. Only one S can be entered. Selection of a location results in the automatic selection of its corresponding SSID.

Exit by pressing Enter or F3 (End). The selection is saved.

Object Options

Object Options are selected by entering the key letter in the Option field in the header. Although the options are displayed only on the Main Menu, an option can be entered in the Option field at any time. A list of valid entries and a short description of their functions follows.

A (Alter)

Changes an existing object. Alterations that are not allowed by the DB2 alter statement (such as altering column types within a table) are fully supported.

C (Create)

Creates a new object. All object types can be created. New objects can be created based on an existing object using the Template function.

D (Drop)

Drops any object from a selection list. Two types of object drops are available: standard and recoverable.

T (Template)

Creates a new object based on an existing object. The characteristics of the existing object are displayed, and the changes necessary to create the new object can then be made.

R (Drop Recovery)

Recovers an object's DDL from the recovery table. The object must have been dropped with a recoverable drop. An object selection panel permits quick selection or deletion of an object's recoverable DDL.

DB2 Objects

The object options must be selected with a DB2 object. To select an object, enter one of the following values in the Object field. The option and object selected determine the function selected:

SG (Storage Group)

Is a named collection of direct access volumes within the same device type. Each simple tablespace, simple indexspace, partition of a partitioned tablespace, and partition of a partitioned indexspace normally has an associated storage group.

Instead of using a storage group, storage can be explicitly defined using VSAM data set definitions. The definition of the VSAM data sets is permitted when creating the tablespace or indexspace.

DB (Database)

Is a collection of logically related objects. It is a collection of stored tables that are related, together with their associated indexes and the various spaces containing those tables and indexes.

TS (Tablespace)

Is a logical address space on secondary storage used to hold one or more stored tables. As the amount of data in those tables grows, storage is acquired from the appropriate storage group or VSAM data set.

TG (Trigger)

Defines the set of actions executed by DB2 when a delete, insert, or update action is applied to a specific table.

SQ (Sequence)

Is an object that you can use to generate a specific sequence of numeric values. The sequence of numeric values is generated in a monotonically ascending or descending order.

RI (Referential Integrity)

Is not a stored object. It describes the process of ensuring that a foreign key has a value within the domain or is null. Referential integrity can be created or updated as part of the Table option, making table alteration easier.

T (Table)

Is a collection of rows having the same columns. The table is the primary object type within DB2.

I (Index)

Provides efficient access to data and help ensure that key values are unique. If referential integrity rules are defined for a table, an index must be created on the columns that compose the unique constraints of the table.

V (View)

Is a virtual table that defines an alternate way of arranging and presenting data from one or more DB2 tables. A view appears as a table to the user although it does not directly exist in physical storage.

S (Synonym)

Is an alternate name for a local table or view. The creation of the same synonym for multiple users is permitted.

A (Alias)

Is an alternate name for a remote or local table or view. The creation of the same alias for multiple users is permitted.

J (Join/Edit)

Refers to a special type of DB2 view produced by the Join/Edit function.

Operation Control Options

As its name implies, operation control options control the way CA RC/Update operates. Valid values are:

L (Log Display)

Displays the CA Log. The log is updated whenever any function is performed, and can be used as an audit trail.

P (Profile)

Defines the print parameters for the QPRINT and PPRINT commands as well as the location of the load libraries for batch submission.

Note: For more information, see the *CA Database Management Solutions for DB2 for z/OS General Facilities Reference Guide*.

BP (Batch Processor)

Invokes the Batch Processor. The Batch Processor executes data sets containing Batch Processor Commands in batch or online mode.

Note: For more information, see the *CA Database Management Solutions for DB2 for z/OS Batch Processor Reference Guide*.

The BP option is used to execute object alterations generated by RC/Alter and DDL generated by the batch mode.

IS (Interactive SQL)

Lets you enter any SQL statement using the Online SQL Execution screen (Control Panel) and the Editor screen. ISQL is invoked to execute the entered SQL statement in batch or online mode. Use the IS option to enter DSN commands such as DISPLAY, STOP, and START DATABASE.

Note: For more information about interactive SQL, see the *CA Database Management Solutions for DB2 for z/OS Value Pack Reference Guide*.

TU (Tutorial)

Provides an online tutorial to introduce the most used features.

Table Data Options

Table data options work with the data in a table and no other objects. Enter one of the following values in the Option field.

Note: The object is assumed to be a table.

B (Browse)

Displays table information on the Browse Options screen.

E (Edit)

Lets you edit DB2 table data on the Edit Option screen.

CO (Copy)

Invokes the RC/Copy facility for copying data between tables.

RB (RI Browse)

Lets you browse tables that have referential integrity relationships. Data is viewed in the same format as the browse and edit options.

FB (Fast Browse)

Provides access to the browse functions without viewing the Browse Options screen.

FE (Fast Edit)

Provides access to the edit functions without viewing the Edit Options screen. RC/Edit uses the standard defaults or the last defaults saved for the selected table.

DC (Data Compare)

Invokes the Data Compare facility for comparing data between tables.

RE (RI Edit)

Lets you edit tables with referential integrity relationships. All of the editing operations available with RC/Edit can be performed, in addition to timesaving features for working with multiple tables of an RI set.

Dynamic Main Menu Facility

The dynamic main menu facility permits denial of user authority and at the same time removes the corresponding option or object from the Main Menu. This simplifies the items that appear on the main menu and increases the security of CA RC/Update use at your site. Even if a function that is not available to a user appears on the main menu, DB2 security prevails.

From the Authorization screen, authorities for managing DB2 objects can be granted or denied. When denying (indicated by N) a user authority, the Main Menu display can be automatically changed for that user. The user must be denied authority for all objects and all options if the corresponding object or option is not to appear on the Main Menu.

For example, users might not want other users to know that they can alter DB2 objects. Previously, user authorities could be denied, such as not granting DB2 ALTER or RC/Alter privileges, from the Authorization screen. When the Main Menu appeared, the user knew that altering an object was an option.

Note: For details about how to access the Authorization screen from the Product Authorizations screen, see the "Operational Considerations" chapter.

Other Header Functions

The standard header is the control center for performing all available functions. It is present on all primary screens to allow easy access to other functions.

Whenever information is entered in the header, the new function is performed. If a user is in the middle of a function, it is canceled and the new request is processed. Users always have the following options. They can:

- Enter new item name or creator ID information to change the object selection lists.
- Change the operation mode at any time by entering O, B, or A.
- Use the extended query at any time by entering Y or S at the Where prompt to control the data displayed in an object selection list.

Remember that although the list of options and objects are displayed only on the Main Menu, the control center in the standard header can be used any time it appears.

Standard Screen Header

The standard header is the control center. The header determines the execution mode, the object type, option, and the selection criteria for selecting the objects. The header is active on all screens except within a function (for example, while in the editor).

The header is always active when displayed allowing easy product navigation. For example, after creating a table, users can change the option to **CO** and jump to the RC/Copy facility. Once the data has been copied, enter **FE** to jump to RC/Edit.

Note: For information about the fields, press F1 (Help).

The first line of the screen identifies the ISPF panel name, software release number, title, and the current date and time.

This version does not support the use of CASE expressions in data queries.

Selection Panels

When altering, dropping, or creating (by template) a view, a selection panel is presented to select the view if not already specified. Make the view selection list more specific by specifying selection criteria for the Item (view) Name and Creator ID prompts in the header.

Use the EQF facility to further specify the views displayed, and use scrolling commands to scroll through information.

Note: For more information about EQF and scrolling commands, see the *CA Database Management Solutions for DB2 for z/OS General Facilities Reference Guide*.

The View Selection screen displays all the fields available in the SYSIBM.SYSTABLES table. This information is available to make selection of the views easier.

To choose items from the listing, type **S** next to the items to select. Press Enter to register the selection.

Note: For information about the fields, press F1 (Help).

Select an Object from a Selection Panel

You can select an item from a selection panel.

To select an item from a selection panel, enter **S** in the column next to the object name.

Note: If you have CA RC/Query, you can also enter a question mark (?) next to an item to display the CA RC/Query Main Menu where you can generate a report. If the code of the report is known, enter the code and skip the menu. Or, you can access CA RC/Query and display the report for the entered option. For lists of report options per object type, see the *CA RC/Query User Guide*.

Object Queues and Shrink Mode

An Object selection panel appears whenever you enter selection criteria for an object prompt on a create, template, or alter screen. When you press Enter, the resulting Object selection panel appears.

On some screens, the object prompt permits multiple object retrieval. For example, the VOLSER prompt on the Storage Group Create screen permits multiple volume retrieval for inclusion in the object definition because a storage group can contain multiple volumes.

If the object prompt for which you requested object help supports the selection of multiple object names, the Object selection panel queues your object selections. Every time you make object selections and press Enter, the objects are placed into a queue and the following message appears:

```
Selected objects have been queued, enter S (Shrink) to view the queue.
```

The selected objects have been queued and you can:

- Continue to make more object selections.
- Enter new selection criteria to receive a new list of objects.
- Press F3 (End) to return to the object screen with the selected objects.
- Enter the S (Shrink) command to view the current list of selected objects.

The S (Shrink) command acts as a toggle between the selection panel and the current queue. While on the selection panel, users can continue to enter new selection criteria and select objects. All selected objects are placed into the queue.

The object queue permits entry of new selection criteria in the header to display a new list of objects without affecting current selections. Whenever objects are selected and Enter is pressed, they are placed into the queue.

To view selections, enter the **S** (Shrink) command on the command line. A Selected Object Queue screen displays a listing of all objects currently selected. Objects can be removed from the list by blanking out the S next to the name. To return to the selection panel, re-enter **S** (Shrink) on the command line.

```
PTVOL3  -------------- Selected Volume Queue -------------
COMMAND ===>                                                SCROLL ===> PAGE

 This is a list of the volumes that you have selected so far,
      removing the 'S' will delete the volume from the selected queue.

Sel  VOLUME
 S_  DB3030
 S_  DB3031
 S_  DEV015
******************************** BOTTOM OF DATA ********************************
```

Entering S (Shrink) again redisplays the selection panel, where more selections can be made or the selection criteria can be changed. The following screen shows new criteria for the selection prompt along with a new list of objects.

```
PTVOL1  -------------- Volume Selection List -------------
COMMAND ===>                                                SCROLL ===> PAGE

Volume Name  ===> UTL%
SSID: D81A ------------------------------------------------------------- USER01

   Enter 'S' to select the volume(s) that you wish to use.

Sel  VOLUME
 __  UTL001
 __  UTL002
 __  UTL003
 __  UTL004
 __  UTL005
```

Remember, to return to the object screen with your selections, press F3 (End). To view the objects in the current queue, enter the **S** (Shrink) toggle to shrink the objects down to the selected objects.

Processing Considerations for Selection Panels

Primary selection panels function differently based on the selected object option. To use EQF, specify **Y** or **S** in the Where prompt.

Note: For more information, see the *CA Database Management Solutions for DB2 for z/OS General Facilities Reference Guide*.

Alter, Drop, or Template Options

When requesting the Alter, Drop, or Template option, a Selection panel appears unless a specific object name is entered. Multiple objects can be selected on the Selection panel. For the Template and Alter options, each selected object appears in succession.

When an object is dropped, you must specify the type of drop from the Selection panel. Valid values are S, SO, R, and RO. Simple standard drops (S) or recoverable drops (R) will immediately display their confirmation screens. Standard option drops (SO) or recoverable option drops (RO) will display their drop options screen before displaying their drop confirmation screen.

Table Options (Edit, Browse, Copy, RI/Edit, and RI/Browse)

If a specific object is entered, the object selection panel does not appear. Otherwise, the selection panel appears listing all objects that meet the entered selection criteria.

Object Selection

Object selection panels always work the same way. Some Object selection panels support the selection of multiple objects and therefore support object queues and the shrink command (see Object Queues and Shrink Mode).

When requesting object help (by entering selection criteria in the object prompt), a selection panel is presented listing the matching objects. The header on object selection panels lacks the option and object prompts.

View and Invoke Available Primary Commands

From any product panel, you can specify commands in the primary command line. These *primary commands* instruct the product to perform various activities. Some commands are available on specific panels only, while other commands are available in many places. Some commands also have an abbreviated version that you can use.

If you are not sure what commands you can use on a given panel, you can display a list of available commands. By viewing available commands (and browsing details), you can understand the purposes and capabilities of commands in a particular product area.

Follow these steps:

1. Enter **?** in the command line on any panel.

   The Primary Command Display panel appears. This panel shows valid panel-specific commands, commands for the product, and commands that are also valid for other CA Database Management Solutionsfor DB2 for z/OS products.

   Note: You can use the scrolling keys F7 (Up) and F8 (Down) to scroll through the list of commands.

2. (Optional) View detailed information about a command:

   a. (Optional) Enter **?** next to a command.

      The product displays details about command description, syntax, and so on.

   b. Press the END key.

      The Primary Command Display panel appears.

3. Do *one* of the following:

   - Enter **S** next a command to invoke the command.

     The product executes the command.

   - Press the END key to exit the Primary Command Display panel.

     The product returns you to the panel from which you invoked the Primary Command Display panel. After you return to the original panel, you can specify a primary command in the command line if needed.

Line Commands

You can use line commands in object screens and in Interactive SQL (IS option) to manipulate multiple line entries. The line commands are similar to the line commands supported in the ISPF editor. Several of the object screens support line commands, including storage groups for the volume list, tables for the column list, synonyms for the user list, tablespaces and indexspaces for the partition list, and views for the view text.

A special section is defined on the screen for managing the multiple entries. A CMD header appears over the line command column. The CMD column accepts the standard ISPF editor line commands.

Standard Line Commands

Line entries can be copied, deleted, and moved using line commands. Any number of new line entries can be inserted or previous operations undone. This section describes the function of each line command and demonstrates the functions through examples.

A list of valid line commands can be displayed by entering a question mark in the line command area. Because the question mark (?) is an invalid line command, the Line Command help screen appears.

What follows is a list of standard line commands. Although not all commands are valid for all objects, the following line commands are valid in most situations. These are referred to throughout this guide as Standard ISPF Line Commands.

D

Deletes the line entry.

R

Repeats the line entry.

I

Inserts a line entry following the line entry.

C

Copies the line entry to a location marked A or B.

M

Moves the line entry to a location marked A or B.

A

Places copied or moved line entries *after* this entry.

B

Places copied or moved line entries *before* this entry.

CC

Provides a block copy. Use CC/CC to delimit a block of entries to copy.

MM

Provides a block move. Use MM/MM to delimit a block of entries to move.

DD

Provides a block delete. Use DD/DD to delimit a block of entries to delete.

RR

Provides a block repeat. Use RR/RR to delimit a block of entries to repeat.

RES

Resets an entry back to its old definition.

U

Undoes the action, resetting an entry back to its old definition (same function as RES).

For example, if you want to create a partitioned tablespace, you can define the first partition line and enter an R# command (where # is the number of remaining partitions to define). Any of the newly inserted partition entries can then be made. The line commands also provide flexibility when defining columns within a table. You can create or move column entries to quickly create the column list that you want.

Examples of Line Commands

These examples illustrate how to use line commands to move blocks of lines and insert lines when working with table columns. The concepts apply to all object types that support line commands.

Example: Move a Block of Table Column Lines

This example shows an excerpt of a Table Alter panel with line commands that will execute a block move for columns:

```
CMD ### COLUMN NAME         COLUMN TYPE       SIZE    N  D  FORDATA  PK UK FK
MM_ 1   CHAR1_NNULL         CHAR              10      N
___ 2   CHAR2_DEF           CHAR              10      N  Y
MM_ 3   CHAR3_NULL          CHAR              10      Y  N
___ 4   INT1_NNULL          INTEGER           4       N
___ 5   INT2_DEF            INTEGER           4       N  Y
A__ 6   INT3_NULL           INTEGER           4       Y  N
___ 7   SMINT1_NNULL        SMALLINT          2       N
___ 8   SMINT2_DEF          SMALLINT          2       N  Y
___ 9   SMINT3_NULL         SMALLINT          2       Y  N
```

The two MM line commands select a block of lines to be moved (column numbers 1 through 3 in this example). The A line command selects the line after which to move the selected block.

After you specify all commands and press Enter, the updated panel shows the moved columns:

```
CMD ### COLUMN NAME         COLUMN TYPE        SIZE   N  D  FORDATA  PK UK FK
___ 1   INT1_NNULL          INTEGER            4      N
___ 2   INT2_DEF            INTEGER            4      N  Y
___ 3   INT3_NULL           INTEGER            4      Y  N
___ 4   CHAR1_NNULL         CHAR               10     N
___ 5   CHAR2_DEF           CHAR               10     N  Y
___ 6   CHAR3_NULL          CHAR               10     Y  N
___ 7   SMINT1_NNULL        SMALLINT           2      N
___ 8   SMINT2_DEF          SMALLINT           2      N  Y
___ 9   SMINT3_NULL         SMALLINT           2      Y  N
```

Example: Insert Table Column Lines

This example shows an excerpt of a Table Alter panel with a line command that will insert three lines:

```
CMD ### COLUMN NAME         COLUMN TYPE          SIZE   N  D  FORDATA  PK UK FK
___ 1   INT1_NNULL          INTEGER   4      N
___ 2   INT2_DEF            INTEGER   4      N  Y
___ 3   INT3_NULL           INTEGER   4      Y  N
___ 4   CHAR1_NNULL         CHAR      10     N
___ 5   CHAR2_DEF           CHAR      10     N  Y
I3_ 6   CHAR3_NULL          CHAR      10     Y  N
___ 7   SMINT1_NNULL        SMALLINT  2      N
___ 8   SMINT2_DEF          SMALLINT  2      N  Y
___ 9   SMINT3_NULL         SMALLINT  2      Y  N
```

After you specify the command and press Enter, the product inserts three lines below the line where you entered the command. You can add the new column names, column types, sizes, and so on by typing on the empty lines.

Note: As you add columns, Row Size increases. You can use the C (Compare) primary command to display a comparison of the old and new table definitions. The comparison will identify any columns that you added.

Special Line Commands

Some line commands are valid for only specific objects. Each of these special line commands is discussed in detail in the corresponding chapter of this guide. For example, the E command (Column Explode Detail) is valid only when the line entries are table columns.

DB2 Commands

While using CA RC/Update, you might want to execute DB2 (DSN) commands. Some of the executable DSN commands include START DATABASE; STOP DATABASE; DISPLAY DATABASE; DISPLAY UTILITY; and TERM UTILITY. If you have authority to execute these commands, there are several methods for their use.

- Execute the TSO DSN processor. Because TSO commands are accepted, the DSN CLIST can be executed by entering the following command on the command line:

  `TSO DSN SYSTEM(`*`sysid`*`)`

  Replace *sysid* with the correct DB2 subsystem ID. The DSN prompt appears and any DSN commands can be entered. To return to CA RC/Update, enter the END command at the DSN prompt.

- Enter the DSN commands using the IS option. The Batch Processor commands necessary to invoke DSN must be entered. These are documented in the *CA Database Management Solutions for DB2 for z/OS Batch Processor Reference Guide*.

  Once the DSN commands have been entered, they can be executed interactively or in batch using ISQL.

- Enter the DSN commands in a data set. Whenever executing the DSN commands (or any other Batch Processor commands), execute the data set using the BP option.

Note: The Command Processor can be used to execute any DB2 command. For more information, see the *CA Database Management Solutions for DB2 for z/OS Value Pack Reference Guide*.

Scrolling and Expanding Fields

When your data values are longer than the display area, you can scroll and expand the fields and columns to view long names or input data as follows:

- For fields that cannot be scrolled, like header fields, place your cursor in the field to be expanded and specify the EXPAND command (typically, using a PF key setting).
- For scrollable fields, the greater than sign (>) appears for fields with truncated values when the column is not wide enough to display the long value. You can view or edit this data using scroll commands (or the EXPAND and SETWIDTH commands).

Note: For more information about these commands, enter a question mark (**?)** on the command line or see the *CA Database Management Solutions for DB2 for z/OS General Facilities Reference Guide.*

DDL Execution

When F3 (End) is pressed from the Create, Alter, Template or Drop screen, a confirmation screen appears before the DDL is executed. Accept, edit, or reject the DDL to be used to complete the request.

Note: For more information, see the “Using CA RC/Update” chapter.

If your changes are complex, require dropping and recreating objects (and their dependencies), or the current CA RC/Update Operation Mode is set to A (RC/Alter), you can perform a complete analysis of your changes that includes rebuilding the DDL and optionally including utility statements and extended DDL generation and analysis options.

Online Operation Mode

Online operation mode is active when the MODE prompt in the standard header is set to O (Online).

For all Creates and Templates, most Drop requests and many kinds of alterations to DB2 objects, the online operation mode generates and executes the DDL for you, all online.

During online execution, an Audit Message File screen appears, showing the results of each executed DDL and CA Batch Processor command.

Each executed command appears along with the resulting return code. Page through the execution messages using the standard ISPF up/down paging keys. Enter END or press Enter to exit the display and return.

The following PBP commands are always executed at the start of execution.

.LIST TERM

List the output to the terminal.

.CONNECT sysid

Connects to the DB2 subsystem.

.OPTION NOERRORS

Informs PBP to perform a rollback if an error is encountered during execution.

At the end of the execution, the Batch Processor displays information about the PBP control options used for the execution.

Note: For more information about these commands, see the *CA Database Management Solutions for DB2 for z/OS Batch Processor Reference Guide*.

Batch Operation Mode

Batch operation mode is active when the MODE prompt in the standard header is set to B (Batch).

For all Creates and Templates, most Drop requests and many kinds of alterations to DB2 objects, the batch operation mode generates the DDL for you online and then writes the DDL to a data set (specified at the time batch operation mode was requested) for you so that it may be executed later, either online or through a batch job using the Batch Processor facility.

Request Batch Operation Mode

You can request batch operation mode at any time.

Follow these steps:

1. Enter B (Batch) for the Mode prompt in the standard header.
2. Enter the name of the data set to contain the generated DDL statements (a partitioned data set or sequential file name). Standard ISPF member selection criteria (for example, MEMB*) can also be entered to list members meeting the criteria.

   If no member is specified for a partitioned data set (PDS), press Enter to display a member selection list. From the member selection list, enter one of the following:

   - S (Select)
   - E (Edit)
   - B (Browse)

3. Specify the Data Set Options for the specified data set.

   - For a PDS member, the member can be replaced if it already exists. If selection criteria was entered for the Member field, Y must be entered to replace the member. Otherwise, the member selection list does not appear. Users can also specify the data set disposition using one of the following commands.

     OLD

     Allocates the data set but do not allow other users or jobs to access the data set while it is allocated. The whole PDS is locked out.

     SHR

     Allocates the data set but allow other users or jobs to access the data set (PDS).

   - For a sequential data set, specify the data set disposition using one of the following commands:

     OLD

     Allocates the data set and do not allow other users or jobs to access the data set while it is allocated. The previous contents of the data set are overwritten.

     SHR

     Performs the same as OLD but lets other users and jobs access the data set.

     MOD

     Adds the generated DDL to the end of the data set.

   Press Enter (after entering all data set information and options).

   The specified PBP data set is allocated and opened. The previous screen appears with a message indicating that the data set has been allocated.

4. To cancel from the Batch Processor Specification screen without allocating and opening a data set, enter **CANCEL**.

   The previous screen appears, and the Operation Mode returns to its previous setting.

Exit Batch Operation Mode

To exit batch operation mode, do one of the following:

- Exit the product.
- Unallocate the data set.

- Change the operation mode from B (Batch) to O (Online). When changing from batch operation mode to online operation mode, the CA RC/Update Online Specification screen displays so the disposition of the data set can be specified.
- Change the operation mode from B (Batch) to A (RC/Alter). When changing from batch operation mode to RC/Alter operation mode, the CA RC/Update Online Specification screen displays so the disposition of the data set can be specified.

Executing the Batch Generated Data Set

To execute a data set generated while in Batch Operation Mode, use the Batch Processor (BP) option. The BP option executes the CA Batch Processor to execute DDL in a data set. In this case, the data set is the data set generated while in Batch Operation Mode.

There are two options for execution.

- Execute the DDL online or in batch (JES).
- Request to write the Batch Processor job to a data set, and submit the job at any time. This data set contains all the necessary JCL to invoke the Batch Processor to execute the specified data set.

Note: For more information, see the *CA Database Management Solutions for DB2 for z/OS Batch Processor Reference Guide*.

RC/Alter Operation Mode

RC/Alter operation mode is active when the MODE prompt in the standard header is set to A (RC/Alter).

In this mode, RC/Alter is unconditionally invoked when you enter END from any Alter, Create, or Template screen or select any objects to drop from any CA RC/Update Drop Selection panel.

All activities that are normally managed by CA RC/Update are passed directly to RC/Alter, including:

- All change management analysis
- All drop management and processing
- All DDL generation.

RC/Alter then uses CA RC/Migrator to perform a complete analysis of the Alter, Create, Template or Drop request.

After the analysis, CA RC/Update no longer manages recovery DDL. The recovery DDL is generated and saved by CA RC/Migrator to the data set specified on the CA RC/Update Recovery Options panel at the time when the CA RC/Migrator generated DDL is actually executed.

Note: If you want to recover your dropped objects using the CA RC/Update Recover Dropped Objects facility, do not use execution mode A or respond with A to any DDL execution confirmation or warning pop-up window.

Catalog PDS Unload Program

Every time an object alteration requiring RC/Alter is submitted, RC/Alter builds an internal structure of the DB2 subsystem. The PDS Option lets users with large DB2 systems access a partitioned data set (PDS) that contains the internal structure. This provides significant improvements in performance when making object alterations. To use this option, you must:

- Update the MIGRATOR member of *high-level*.CDBAPARM so it identifies the PDS to use for each DB2 subsystem.

  Note: For more information about updating the MIGRATOR member, see the *CA Database Management Solutions for DB2 for z/OS Implementation Guide*.
- Run the Catalog PDS Unload program to create the PDS for each subsystem.
- Elect to use the PDS option on the RC/Alter Alteration Analysis screen.

Notes:

- When you run the Catalog PDS Unload program, the PDS used by both programs (CA RC/Update and CA RC/Migrator) is updated for a particular subsystem.
- The PDS option must not be in use to use the VSAM read option.

More information:

Run the Catalog PDS Unload Program (see page 64)
VSAM Read Option (see page 65)

Run the Catalog PDS Unload Program

RC/Alter builds an internal structure every time it must perform an object alteration. The Catalog PDS Unload program builds this internal structure and stores it in a partitioned data set. If using the PDS option, RC/Alter then uses the information in the PDS to perform the alteration.

The PDSUNLD command runs the Catalog PDS Unload program. The Catalog PDS Unload program must be run for each subsystem on which the PDS option is needed. This program permits setting of execution specifications through the Catalog Unload JCL Interface screen, and then it generates JCL to unload subsystem information to a partitioned data set. Once the information has been unloaded, it is accessed by RC/Alter when using the PDS Analysis option.

Note: We recommend that the Catalog PDS Unload program be executed on a regular basis (nightly) to help ensure that the PDS contains current DB2 System Catalog information.

Catalog Unload JCL Interface Screen

The Catalog PDS Unload program must be run for each subsystem on which the PDS option is needed. The Catalog Unload JCL Interface, shown in the following example, prepares the JCL for batch execution. To access the Catalog Unload JCL Interface screen, enter **PDSUNLD** at the command line.

Note: For information about the fields, press F1 (Help).

After entering values for these fields, press F3 (End) to generate the JCL to unload your catalog. The Batch JCL Specification screen appears.

Note: For more information about the Batch JCL Specification screen, see the *CA Database Management Solutions for DB2 for z/OS Batch Processor Reference Guide*.

VSAM Read Option

The VSAMID install parameter lets us read DB2 information directly from the DB2 data sets. Using the VSAM Read option helps ensure that CA RC/Update is always using up-to-date catalog information, whereas with the PDS option, the DB2 information is current only up to the point when you last ran the Catalog PDS Unload program.

Note: The VSAM Read option is valid only with DB2 V8 (running in Compatibility Mode) and prior releases of DB2.

To activate the VSAM Read option on a subsystem, the following requirements must be met:

- ACM option must be turned off.

  Note: For more information about ACM, see the *CA Database Management Solutions for DB2 for z/OS Value Pack Reference Guide*.

- PDS profile option must not be in use.

- PTLTSRB must be installed correctly.

  Note: For more information, see the *CA Database Management Solutions for DB2 for z/OS Implementation Guide*.

- The VSAMID (ID) parameter of the UPDATE parmlib member must be specified for the subsystem. This parameter must be specified for every subsystem on which this option is to be used. Valid values are:

  - *—Specify an asterisk to indicate that everyone who uses CA RC/Update has control interval access to the DB2 catalog data sets.
  - *user-ID*—Identifies a specific user that has control interval access to the DB2 catalog data sets. This user ID's authorization will be used to read the DB2 catalog information.

  A user ID can be created that has only this access, control interval access is not wanted for a real user.

  If the VSAMID parameter is not specified in the UPDATE parmlib member, current DB2 information is not obtained from the VSAM data sets.

Note: The VSAM Read Option is only used when RC/Alter is invoked or when requesting a Recoverable Drop or Drop Impact Analysis.

Chapter 5: Defining Profile Variables

This section contains the following topics:

Overview of Defining Profile Variables

You can define global profile variables and product-specific profile variables. You can set the following profile variables from the Profile screen:

- Global profile variables
- CA RC/Update profile variables
- RC/Edit profile variables
- RC/Objects profile variables
- Model services

The global profile lets you specify profile values that are applied to all CA Database Management Solutions for DB2 for z/OS products.

Note: For detailed information about defining the global profile variables, see the *CA Database Management Solutions for DB2 for z/OS General Facilities Reference Guide*.

Access the Profile Menu

The profile menu shows variables that are available.

To access the profile menu, enter **PROFILE** in any command line.

You can enter an option number in the Option field.

Set CA RC/Update Profile Variables

You can define the following defaults for CA RC/Update:

- Whether to save header field values at the end of a session and display them as defaults in the next session.
- Whether to return or process object input as uppercase, lowercase, or mixed case using the MATCHCASE setting.

Follow these steps:

1. Enter **2** in the Option field on the Profile menu.

   The CA RC/Update Operational Parameters screen appears.

2. Enter data in the fields as applicable at your site.

   Note: For information about the fields, press F1 (Help).

3. Press Enter to process your entries.

Set RC/Edit Profile Variables

You can define the following defaults for RC/Edit:

- Whether authorization should be checked before a user enters RC/Edit
- The date and time formats
- Comma formatting for numeric data
- Capitalization conversion for character data
- The edit method for RC/Edit and RC/Browse
- How often to commit changes during a Fast Edit
- Whether the Edit Table selection panel appears when you exit an RI/Edit session
- The way you edit views
- String delimiter characters for varying character columns of tables
- Asynchronous background fetch options

Note: You can also set parameters in the RCEDIT member of *high-level*.CDBAPARM. These parameters affect RC/Edit and RC/Browse. For more information, see the *CA Database Management Solutions for DB2 for z/OS Implementation Guide*.

Follow these steps:

1. Enter **3** in the Option field on the Profile menu.

   The RC/Edit Parameters screen appears.

2. Enter data in the fields as applicable at your site.

   Note: For information about the fields, press F1 (Help).

3. Press Enter to process your entries.

Set RC/Objects Profile Variables

You can set the following RC/Objects variables:

- Whether SET CURRENT SQLID statements will be generated for object create statements.
- Whether .AUTH statements will be suppressed for object create statements.
- Whether the Audit File will be displayed automatically after DDL processing when you are in online mode.
- Whether to force all object drops to use the recovery option.
- Whether to display DDL execution confirmation or warning pop-up windows from the CA RC/Update alter, create, template and drop confirmation screens.
- Whether to display the Use ASIS Confirmation pop-up window or how to interpret symbols embedded in the name or creator of an object's attribute.

Follow these steps:

1. Enter **4** in the Option field on the Profile menu.

   The RC/Objects & DDL Generation Profile Parameters screen appears.

2. Enter data in the fields as applicable at your site.

   Note: For information about the fields, press F1 (Help).

3. Press Enter to process your entries.

Setting Model Services Profile Variables

Model IDs are used to generate CA Batch Processor statements for utilities. These model IDs contain symbolic variables. Model Services lets you change the values of these symbols, create new model IDs, or alter the contents of the models, and thereby automatically include your customized changes.

Note: Model JCL is also used to prepare JCL for batch execution. The Batch Processor does not use these model JCL members. They reside in *high-level*.CDBAMDL and their names begin with MJ.

Using model services, you can create model IDs to routinely do the following tasks:

- Allocate disk space proportionately based on object size.
- Allocate files to disk or tape, depending on the size of the object. (You decide how big is “too big” to be on disk.)
- Use a third party utility to perform a function. (You can substitute CA Rapid Reorg for the IBM Reorg function if you want to.)
- Specify various utility options.

Note: If you are satisfied with the Batch Processor statements that we generate as they are, you never need to use model services. It is only when you want to customize the output that you use model services. Batch Processor, CA RC/Migrator, and RC/Alter are completely functional without making changes through model services.

Model IDs

Model IDs let you specify a set of utility names. The model ID is used to generate Batch Processor statements for those utilities. (Model JCL members, such as member MJBPMFL, used by the Batch Processor, are not included in model IDs.)

Note: For more information about how to specify models, see the “RC/Alter” chapter.

You can browse, template, update, or delete model IDs. Default model IDs are provided, which you can customize for your site. Multiple model IDs can exist for a single site. Each model ID also contains replacement values for any user-defined symbolic variables.

Note: When you install a new version, it optionally updates the @DEFAULT model ID. It will not change any user model libraries. The @DEFAULT model ID does not need to be updated. It will work without any changes and can be used as a template for new model IDs. A list of model utility names referenced by the @DEFAULT model ID is provided later in Utilities Referenced by @DEFAULT.

Utility Model Services Privilege

To define model services variables, you must be authorized through the Product Authorization Facility.

Note: For more information about defining authorizations, see the *CA Database Management Solutions for DB2 for z/OS General Facilities Reference Guide*.

Access Model Services

To access Model Services, enter **5** in the Option field on the Profile menu.

The General Model Services screen appears.

The header information displays the following information about the model IDs. All fields can be updated to change the model IDs listed.

Note: For information about the fields, press F1 (Help).

Create a Model ID

You can create a model ID.

Follow these steps:

1. Enter **T** in the CMD column next to the model ID you want to use as a template.

   The General Model Utilities panel appears.
2. Specify the following information for the new model ID:
   a. Enter a new name and description on the General Model Utilities panel.
   b. Add or delete the utilities you want in the model ID.
   c. (Optional) Specify the source of space statistics.
   d. (Optional) Specify symbolic variable information.

   Press F3 (End) to save the model ID.

Update Model IDs

You can update an existing model ID.

Follow these steps:

1. Enter **U** on the General Model Services screen next to the model ID you want to update.

   The General Model Utilities panel appears.
2. Make any necessary changes, and then press F3 (End).

   The General Model Services screen appears.

Browse Model IDs

To browse a model ID, enter **B** (browse) in the line command next to an existing model ID on the General Model Services screen. The General Model Utilities panel appears.

Its fields will appear as display only; however, you will be able to use the **S** (Symbols) and **B** (Browse) line commands from the General Model Utilities panel.

Delete Model IDs

To delete a model ID, enter **D** (delete) in the line command next to an existing model ID on the General Model Services screen. The model ID is automatically deleted.

You can only delete model IDs that you have created or that have a Share Option value of U.

General Model Utilities Panel

The General Model Utilities panel is where you specify information about the model ID. On this panel, you indicate:

- The model ID name, creator, description, and share option
- The utilities to be included in the model ID

You can enter the SYMBOLS command to display the symbolic variables for all the utilities. The General Model Symbols screen appears.

The header information displays the following information about the model IDs. All fields can be updated to change the model IDs listed.

Note: For information about the fields, press F1 (Help).

More information:

Model Symbols Screen (see page 87)

Excluding Utilities

Use the General Model Utilities panel to exclude any utility, but certain combinations of exclusions are prohibited. These prohibited combinations and other usage rules for combinations are as follows:

- You cannot exclude *both* of the following pairs of utilities:
  - CA Fast Unload (FUNLD) and Unload Table (UNLOAD)
  - CA Fast Unload - Recovery (FUNLD_R) and Unload Table - Recovery (UNLOAD_R)
  - CA Fast Load (FLOAD) and Load Table (LOAD)
  - CA Fast Load - Recovery (FLOAD_R) and Load Table - Recovery (LOAD_R)

  For example, if you exclude CA Fast Unload and then try to exclude Unload Table, CA Fast Unload is made active again, and vice versa.
- If you use CA Fast Load (FLOAD), you must use CA Fast Unload (FUNLD).
- If neither Reorg (REORG) nor CA Rapid Reorg (RREORG) is excluded, CA Rapid Reorg will be used in the analysis.
- If neither Image Copy (COPY) nor CA Quick Copy (QCOPY) is excluded, CA Quick Copy will be used in the analysis.

For any of the paired utilities listed above (for example, Image Copy and CA Quick Copy), if both are excluded, a message is issued that no utilities will be generated, but the analysis will complete successfully. Also, the message appears if one of the non-paired utilities is excluded (for example, RUNSTATS).

When an index reorganization is generated, the sequence for determining which unexcluded utility to use is as follows:

- CA Rapid Reorg of an Index (RREORGI)
- Reorg Indexes (REORGI)
- CA Rapid Reorg (RREORG)
- Reorg (REORG)

When a tablespace reorganization is generated, the sequence for determining which unexcluded utility to use is as follows:

- CA Rapid Reorg (RREORG)
- Reorg (REORG)

If all four utilities are excluded, a message is issued that no utilities will be generated, but the analysis will complete successfully.

Example Model Utility

The following example displays all of model utility Load as it appears in the Model Symbols screen. This is the utility referenced by @DEFAULT to load tables. The display includes user-defined symbols, automatic symbols and the model utility text.

```
[1]  UTILITY   OBJ DESCRIPTIONC              SIZE
     LOAD      T   LOAD TABLE                RPI

[2]  CMD SYMBOL    ERR  REPLACEMENT VALUE
         DATACLAS
         DISCDSN        %UNLDDSN..D
[3]      DISCSP1        %CALC(%ROWLEN+6*%NROWS/%BYTETRK*20/100+1)
         DISCSP2        %CALC(%ROWLEN+6*%NROWS/%BYTETRK*30/100+1)
         DISCSP3        TRACKS
         DISCUNIT       %DISK
         DISK           SYSDA
         ERRDSN         %UNLDDSN..E
         ERRSP1         %CALC(%NROWS*80/4096*30/100+1)
         ERRSP2         %CALC(%NROWS*80/4096*40/100+1)
         ERRSP3         BLK(4096)
         ERRUNIT        %DISK
         LOADDSN        %UNLDDSN
         LOG            NO
         MAPDSN         '%UNLDDSN..M'
         MAPSP1         %CALC(%NROWS*16/4096*30/100+1)
         MAPSP2         %CALC(%NROWS*16/4096*40/100+1)
         MAPSP3         BLK(4096)
         MAPUNIT        %DISK
         MGMTCLAS
         OUTSP1         %CALC(%ROSORT*050/100+1)
         OUTSP2         %CALC(%ROSORT*030/100+1)
         OUTSP3         CYL
         SORTDEVT       %DISK
         SORTNUM        0
         STORCLAS
         UT1SP1         %CALC(%ROSYSUT1*050/100+1)
         UT1SP2         %CALC(%ROSYSUT1*030/100+1)
         UT1SP3         CYL
```

```
   ------------------------------------------------------------------------
[4]     BYTETRK        *** AUTOMATIC VARIABLE ***
        COLUMNS        *** AUTOMATIC VARIABLE ***
        CREATOR        *** AUTOMATIC VARIABLE ***
        ENFORCE        *** AUTOMATIC VARIABLE ***
        LOADCALL       *** AUTOMATIC VARIABLE ***
        NROWS          *** AUTOMATIC VARIABLE ***
        OBJECT         *** AUTOMATIC VARIABLE ***
        ROSORT         *** AUTOMATIC VARIABLE ***
        ROSYSUT1       *** AUTOMATIC VARIABLE ***
        ROWLEN         *** AUTOMATIC VARIABLE ***
        TOSSID         *** AUTOMATIC VARIABLE ***
        UNLDDSN        *** AUTOMATIC VARIABLE ***
[5]  #CM  ********************************************************************
     #CM  UNLDDSN  = UNLOAD DATASET NAME
     #CM  UNLDBKSZ = UNLOAD DATASET BLOCKSIZE
     #CM  ENFORCE  = TO DISABLE RI CHECKING, IF NEEDED
     #CM  COLUMNS  = LIST OF COLUMNS TO LOAD
     #CM  ********************************************************************
     #CM  RC/Update PRESERVES THE UNLOAD DSN AND BLKSIZE TO
     #CM  COORDINATE THE INDDN, SYSDISC AND SYSERR ALLOCATIONS.
     #CM  ********************************************************************
[6] .CALL UTIL LOAD PARM(%TOSSID)
    .ALLOC FI(PTILOAD)+
       DA('%LOADDSN') SHR
    .ALLOC FI(SYSDISC)+
       DA('%DISCDSN')+
       MOD CATALOG UNIT(%DISCUNIT)+
       STORCLAS(%STORCLAS)+
       DATACLAS(%DATACLAS)+
       MGMTCLAS(%MGMTCLAS)+
       LIKE('%UNLDDSN')+
       SPACE(%DISCSP1,%DISCSP2) %DISCSP3 RLSE
     .ALLOC FI(SYSERR)+
       DA('%ERRDSN')+
       MOD CATALOG UNIT(%ERRUNIT)+
       STORCLAS(%STORCLAS)+
       DATACLAS(%DATACLAS)+
       MGMTCLAS(%MGMTCLAS)+
       SPACE(%ERRSP1,%ERRSP2) %ERRSP3 RLSE
```

```
.ALLOC FI(SYSMAP)+
   DA(%MAPDSN)+
   MOD CATALOG UNIT(%MAPUNIT)+
   STORCLAS(%STORCLAS)+
   DATACLAS(%DATACLAS)+
   MGMTCLAS(%MGMTCLAS)+
   SPACE(%MAPSP1,%MAPSP2) %MAPSP3 RLSE
.ALLOC FI(PTIUT1) UNIT(%DISK)+
   STORCLAS(%STORCLAS)+
   DATACLAS(%DATACLAS)+
   MGMTCLAS(%MGMTCLAS)+
   SPACE(%UT1SP1,%UT1SP2) %UT1SP3
.ALLOC FI(PTIOUT) UNIT(%DISK)+
   STORCLAS(%STORCLAS)+
   DATACLAS(%DATACLAS)+
   MGMTCLAS(%MGMTCLAS)+
   SPACE(%OUTSP1,%OUTSP2) %OUTSP3
.DATA
   LOAD DATA INDDN PTILOAD RESUME YES LOG %LOG %ENFORCE
    DISCARDDN SYSDISC
    ERRDDN   SYSERR
    MAPDDN   SYSMAP
    WORKDDN  (PTIUT1,PTIOUT)
    SORTDEVT %SORTDEVT
    #IF(%SORTNUM,¬=,0)
    SORTNUM %SORTNUM
    #ENDIF
    #IF(%LOADCALL)
    CONTINUEIF (72) = X'FF'
    #ENDIF
    INTO TABLE %CREATOR..%OBJECT
    %COLUMNS
.ENDDATA
.FREE FI(PTILOAD)
.FREE FI(SYSDISC)
.FREE FI(SYSERR)
.FREE FI(SYSMAP)
.FREE FI(PTIUT1)
.FREE FI(PTIOUT)

  ERROR LISTING
  ----- -------
```

More information:

Model Symbols Screen (see page 87)

Utility Information

The utility name (Load), object type (Table), description (Load Table), and size source (RPI) are displayed on line [1].

Column Headings

The following columns are displayed on the General Model Symbols screen on line [2]:

CMD

Command field. From the Model Symbols screen, you can insert, repeat, and delete lines.

SYMBOL

Symbolic variable names, in alphabetical order.

ERR

Error. This field displays a value if an error is flagged for the symbolic. Error flag values are described in Model Symbols Screen.

REPLACEMENT VALUE

You can specify the value of the symbolic variable. For example, DISCDSN is replaced with %UNLDDSN..D.

For automatic symbolics, Automatic Variable appears.

Symbols Area

The symbols areas of the General Model Symbols screen are described as follows. Annotation numbers are used to reference the example in Example Model Utility.

[3] %CALC

%CALC tells the Batch Processor that an arithmetic formula follows. The value of DISCSP1 is calculated using an equation which incorporates the value of the automatic variables %ROWLEN, %NROWS, and %BYTETRK. (The definitions of all automatic variables are in the tables that follow.)

[4] Automatic Variables

Automatic symbolic variables are displayed after the user-defined symbols.

Note: Automatic symbolics are *not* displayed unless you use the AUTOSYMS command.

[5] #CM

#CM is a comment in the model utility that will not be put into the generated JCL.

[6] Batch Processor Statements

These are actual Batch Processor statements. The first line instructs the Batch Processor to call the load utility using the value of %TOSSID as the parameter. This is the same as issuing the command:

```
LOAD(value of TOSSID)
```

Conditional Logic

Model utilities can contain conditional logic (IF, ELSEIF, ELSE, ENDIF). This feature lets you use a single model utility to provide different utilities based on different conditions.

Conditions can be based on the size variables MEDIUM and LARGE, which are specified on the General Model Utilities panel. These variables measure size in pages.

For example, the COPY model contains logic that sends image copy data sets to DASD if they are smaller than a specified size. The size is specified as the %LARGE variable.

```
.CALL UTIL COPY PARM(%TOSSID)
.ALLOC FI(SYSCOPY) DA('%COPYDSN')+
  #IF(%REORGP,<,%LARGE)                          <=== Conditional logic based
    NEW CATALOG UNIT(%COPYDUNT)+                    on the size needed for a
    VOLUME(%COPYDVOL)+                              REORG
    SPACE(%COPYSP1,%COPYSP2) %COPYSP3 RLSE
  #ELSE
    NEW CATALOG UNIT(%COPYTUNT)+
    VOLUME(%COPYTVOL)+
    POS(%INCR) LABEL(%LABEL)
  #ENDIF
.DATA
    COPY TABLESPACE %CREATOR..%OBJECT DEVT %DEVT
.ENDDATA
.FREE FI(SYSCOPY)
```

If the size required for reorganization (%REORGP) is less than %LARGE, the image copy is sent to DASD. If it is greater than or equal to %LARGE, it is sent to tape.

The MEDIUM variable works the same as LARGE. You can use one or both variables in the same model utility.

Syntax Rules

Note the following syntax rules:

- A pound sign (#) must precede each keyword (IF, THEN, ELSE, and ELSEIF), for example, #IF.
- Symbolic tests and conditional logic must be contained within parentheses.
- No blanks can be placed between keywords and conditional tests.

  #IF(%SQLPOSA) is valid.
  #IF (%SQLPOSA) is invalid.
- All automatic and conditional symbolics must be prefixed with a % (percent sign).
- An operator must be separated from its operands by commas (operand,operator,operand).

The following table describes conditional compare symbols that can be used:

Symbol	Meaning
= or EQ	Equal
< or LT	Less Than
> or GT	Greater Than
¬= or NE	Not Equal

For example, the following statements are valid:

```
#IF(%SQLPOSA) *

#IF(%REORGP,<,%LARGE)

#IF(%REORGP,<,%LARGE)
  NEW CATALOG UNIT(%COPYDUNT)+
  VOLUME(%COPYDVOL)+
  SPACE(%COPYSP1,%COPYSP2) %COPYSP3 RLSE
#ELSE
  NEW CATALOG UNIT(%COPYTUNT)+
  VOLUME(%COPYTVOL)+
  POS(%INCR) LABEL(%LABEL)
#ENDIF

#IF(%OBJTYPE,=,TS)
 LOG NO
#ENDIF
```

Note: Within the Model Services conditional logic, True has a value of one and False has a value of anything other than one.

***** In this example, the block is generated only when %SQLPOSA has a value of one.

Symbolic Parameters

Symbolic variables, which are general values that can be substituted for specific values, are included with the models supported using Model Services. These variables let you add conditional testing, change data set names, use different calculations than the defaults, and so on. There are two types of symbolic parameters: automatic and user-defined.

- Automatic symbolics are replaced automatically when the utility is executed. For example, %TSNAME will be replaced by the appropriate tablespace name.
- User-defined symbolics require you to enter a value. The value you specify will replace the symbolic when the utility is executed.

  Note: If you change a symbolic, such as DSN, its value is changed everywhere the symbolic is used. These values are stored in a DB2 table. Only one row can exist per symbolic.

If you want a period to separate a symbolic parameter from the next field in the name, then you must use two periods. For example, %USERID..%OBJECT..DATA would expand to PTIPH.EMPLOYEE.DATA if the user ID is PTIPH and the table name is EMPLOYEE.

Automatic Symbolics Available for Model Utilities

The following automatic symbolics are available for model utilities:

- Automatic symbolics for all objects (see page 81)
- Conditional automatic symbolics for all objects (see page 81)
- Conditional automatic symbolics per object (see page 82)
- Symbolics for partitioned objects (see page 83)
- Conditional automatic symbolics for space calculation
- Special-use symbolics (see page 84)

For All Objects

The following table lists the automatic symbolics available for all objects:

Symbolic	Description
STRATEGY	Strategy name.
USERID	TSO ID that submitted the analysis.
DATE	Analysis date formatted YYMMDD.
TIME	Analysis time formatted HHMMSSH.
FROMSSID	From subsystem name.
TOSSID	To subsystem name.
PREFIX	The TSO prefix for the User ID at the time that the analysis was run. The prefix is set by the TSO profile.
CCREATOR	The true creator of the object. This is similar to the symbolic variable CREATOR except for tablespaces. For tablespaces, CREATOR is the database name, and CCREATOR is the true creator of the tablespace.

Conditional Automatic Symbols For All Objects

The following table lists the conditional automatic symbolics for all objects:

Symbolic	Description
LASTIMAG	The data set name of the last image copy data set generated by Model Services.
LASTUNLD	The data set name of the last unload data set generated by Model Services.

Symbolic	Description
RECOVERY	Recovery. 1 True 0 False
SQLPOSB	Position for optional utility is before all SQL to create objects and after the unload utility. 1 True 0 False
SQLPOSO	Position for optional utility is with the SQL to create an object. 1 True 0 False
SQLPOSA	Position for optional utility is after all SQL and standard utilities. 1 True 0 False
TINCR	The file sequence number to be used in the LABEL parameter of the .ALLOC command for the tape data set.

Note: RECOVERY, SQLPOSB, SQLPOSO, and SQLPOSA are all conditional symbols. For an example of how to use them, see Conditional Automatic Symbols.

Per Object

The following tables list the conditional automatic symbolics per object:

Symbolic	Description
OBJTYPE	Type of object. SG—Storage Group DB—Database TS—Tablespace T—Table I—Index V—View S—Synonym A—Alias
CREATOR	Object creator
OBJECT	Object name
DBNAME	Object's parent database name
TSNAME	Object's parent tablespace name
INCR	General increment, reset by utility

For Partitioned Objects

The following table describes values that are automatically substituted according to the type of utility named in the model:

Symbolic	Description		
Partitioned Objects		**Non-Partitioned Objects**	
Image Copy Utilities (CA Quick Copy or QCOPY)		Reorganization Utilities (CA Rapid Reorg or RREORG)	
PART	001	partition number	(blank)
PARTLBL	'PART'	'PART'	(blank)
PARTNO	001	partition number	000

Conditional Automatic Symbolics for Space Calculation

The following table lists the conditional automatic symbolics for space calculation:

Symbolic	Description
BYTETRK	Specifies the number of bytes per track, based on the device type used (DEVTYPE).
CALC	Specifies an expression that you define for performing automatic space calculations. CALC is also used by the utilities for generating primary and secondary space allocations.
NROWS	Specifies the number of rows in the table, based on the SIZE option.
REORGP	Specifies the estimated pages for a tablespace or indexspace. The default is 1800 if no statistics are found. **Note:** This value has three possible sources based on the SIZE option specified for your utilities (the default is RPI): R = RUNSTATS, P = PDASTATS, I = IDCAMS. If R is specified first, the values come from: ■ Tables—NPAGES from SYSIBM.SYSTABLES ■ Tablespaces—NACTIVE from SYSIBM.SYSTABLESPACE ■ Indexes—NLEAF from SYSIBM.SYSINDEXES If P is specified first, the values come from: ■ Tables—RATB_PAGES_USED from RATB_STATS_# ■ Tablespaces—RATS_PAGES_REORG from RATS_STATS_# ■ Indexes—RAIX_REORG_PAGES from RAIX_STATS_# If I is specified first, the values come from an IDCAMS LISTCAT: HIGH USED RBA / 4056 (bytes per page)

Symbolic	Description
ROSORT	Specifies the cylinders required for PTIOUT (SORTOUT). PTIOUT holds sorted keys on SORT output for CA Fast Check or CHECK DATA, CA Fast Load or LOAD, and CA Rapid Reorg or REORG processing. The default is 10 cylinders.
ROSYSREC	Specifies the cylinders required for SYSREC. The SYSREC data set holds unloaded records for REORG.
ROSYSUT1	Specifies the cylinders required for SYSUT1. SYSUT1 is the temporary data set that holds sorted keys on SORT output for CA Fast Check or CHECK DATA, CA Fast Load or LOAD, CA Rapid Reorg or REORG, and CA Fast Recover or REBUILD utilities.
ROSORTWK	Specifies the cylinders required for SORTWKnn. SORTWKnn is the work data set for sorting indexes. It is used with the CA Fast Check or CHECK DATA, CA Fast Load or LOAD, CA Rapid Reorg or REORG, and CA Fast Recover or REBUILD utilities.
ROWLEN	Specifies the maximum row length in the table.

When applicable for the utility, you can use the previously listed variables for space calculation with the following utilities: CHECK, CHECK_R, FLOAD, FLOAD_R, LOAD, LOAD_R, REORG, REORGI, RREORG, and RREORGI.

More information:

Space Calculation Syntax Rules (see page 87)
Model Utility Data Set Calculations (see page 94)

Special Use Symbolics

The following table lists special-use symbolics:

Symbolic	Description
COLUMNS	Unload/Load list of columns with options.
ENFORCE	Load disable referential integrity checking, if necessary.
INDEX	Statistics utility index option for tablespace.
INFORM	The CA Fast Load format of the unload data set. Values are LOAD or SEQ.
LARGE	Threshold variable that can be used to determine whether data should go to disk or tape.
LIBRARY	BIND list of unique PDS names from DBRMs.
LIMIT	Unload maximum number of rows.

Symbolic	Description
LOADCALL	Used with the load utility to identify whether data was unloaded with CA Fast Unload or with the Batch Processor UNLOAD facility. ■ 0—Unloaded with CA Fast Unload. ■ 1—Unloaded with .CALL UNLOAD.
MEMBER	BIND list of DBRMs in plan.
OBID1	Values used by CA Fast Load for object ID translation. Values are spaces or OBID.
OBID2	Values used by CA Fast Load for object ID translation. If OBID1 is set to spaces, OBID2 is also set to spaces. Otherwise, OBID2 is the OBID of the data to be loaded.
OPTIONS	BIND list of non-default options.
OUTCNTL	The CA Fast Load OUTPUT-CONTROL parameter. Values are BUILD or ALL.
OUTFORM	The format of the unloaded data set. Corresponds to the CA Fast Unload keyword OUTPUT-FORMAT.
PAGESIZE	Unload utility page size of tablespace.
RBNDNAME	The Plan or Package being rebound.
REPLACE	Load utility allows replace for tablespace RESET.
RESUME	LOAD model utility parameter used with migration strategies only. If a tablespace is newly created, the value of %RESUME to NO is automatically set before loading the first table. For subsequent table loads in that tablespace, the value is automatically reset to YES.
SELECT	Unload utility list of SQL columns.
SORTNUM	Number of temporary data sets to be dynamically allocated for sorting.
TABLE	Statistics utility table option for tablespace.
UNLDBKSZ	Load data set blocksize from unload utility.
UNLDDSN	Load data set name from unload utility.

Note: These symbols should never be modified in the model utilities. UNLDDSN and UNLDBKSZ are useful in user-defined utilities where you need to know the old name of an unloaded object, or the size of an unloaded data set.

Conditional Automatic Symbols

The automatic symbols RECOVERY, SQLPOSB, SQLPOSO and SQLPOSA are conditional symbols whose values are set to either true or false in the course of generating the analysis file. Their values indicate the stage of the analysis. You can use them to control when and how utilities are generated. Model COPY_PC (used by @DEFAULT for Pre-SQL Image Copy) illustrates the use of conditional automatic symbols.

```
 #IF(%SQLPOSB)
   .CALL UTIL COPY PARM(%FROMSSID)
   #IF(%RECOVERY)
     .ALLOC FI(SYSCOPY)+
            DA('%RCOPYDSN')+
   #ELSE
     .ALLOC FI(SYSCOPY)+
            DA('%COPYDSN')+
   #ENDIF
     #IF(%REORGP,<,%LARGE)
       NEW CATALOG UNIT(%COPYDUNT)+
       MGMTCLAS(%MGMTCLAS)+
       STORCLAS(%STORCLAS)+
       DATACLAS(%DATACLAS)+
       VOLUME(%COPYDVOL)+
    SPACE(%COPYSP1,%COPYSP2) %COPYSP3 RLSE
  #ELSE
    NEW CATALOG UNIT(%COPYTUNT)+
    VOLUME(%COPYTVOL)+
    POS(%INCR) LABEL(%LABEL)
  #ENDIF
.DATA
  #IF(%REORGP,<,%LARGE)
    COPY TABLESPACE %CREATOR..%OBJECT DEVT %DEVT
      COPYDDN  SYSCOPY
  #ELSE
    COPY TABLESPACE %CREATOR..%OBJECT DEVT %TAPE
      COPYDDN  SYSCOPY
#ENDIF
```

%SQLPOSB is set to true after the UNLOAD and before the SQL. Therefore, this whole model is generated only after the UNLOAD and before the SQL to create the objects. If the conditional symbol SQLPOSO had been used, it would have been generated with the SQL to create the objects, after the UNLOAD. If SQLPOSA had been used, the utility would have been generated after the SQL and the unload utility.

If this model is part of a recovery routine, the data set name to which the file will be allocated is the value of the user-defined symbol %RCOPYDSN. Otherwise, it will be allocated to the value of the user-defined symbol %COPYDSN.

To see the values of the user symbols, you can look at utility COPY_PC in the @DEFAULT model ID.

Space Calculation Syntax Rules

Math is performed from left to right, and operations within parentheses are performed first. For example,

```
(10+5*10) = 150
(10+(5*10)) = 60
```

Available operators are:

- (—Left delimiter
- +—Add
- -—Subtract
- *—Multiply
- /—Divide
-)—Right delimiter

The following rules apply:

- All equations must begin with a left delimiter and end with a right delimiter.
- All equations must have a matching number of left and right delimiters.

  ```
  (10+10(5*4)) is valid.
  (10+10(5*4) is invalid.
  ```

- All numbers must be integers from 0 to 999999. Leading zeros are allowed.
- No blanks are allowed.

  ```
  (10+10) is valid.
  (10 + 10) is invalid.
  ```

- In division, all remainders are dropped.

Model Symbols Screen

If you use the S line command on the General Model Utilities panel, the General Model Symbols screen appears. This screen lets you specify values for the symbolic variables that appear in the model utility.

If you accessed the General Model Symbols screen by using the SYMBOLS command, symbolics for all utilities are listed.

You can use the QPRINT command to print the list of symbolics.

The model ID name, creator, description, and SSID are displayed. This information is not updatable.

The utility name, object type, and description are displayed. This information is not updatable.

Note: For information about the fields, press F1 (Help).

Use the F7 and F8 keys to scroll the symbolics list and analysis model text.

Press Enter on the General Model Utilities panel with no line commands. This will perform a model substitution, and the changes you have made will be saved.

Once you have specified a value for all user-defined symbolics, press F3 (End) to return to the General Model Utilities panel.

Automatic Symbolics

Automatic symbolics are not displayed on the General Model Symbols screen.

To display a list of automatic symbolics, enter the AUTOSYMS (AS) command in the command line. A list of automatic symbolics is placed after the list of user symbolics.

To toggle off the automatic symbolics display, use the AUTOSYMS command.

Edit the Model Utility Text

From the General Model Utilities panel, you can display a model utility in an edit session.

Follow these steps:

1. Enter the **E** line command for the model utility you want to edit.

   An edit session is initiated.

2. Make any necessary changes, and then press F3 (End) to save the changes.

   You are returned to the General Model Utilities panel.

Note: You can use the CANCEL command to return to the General Model Utilities panel without saving changes.

%SUBSTR Function

In editing a model utility from the General Model Symbols screen, you can derive substrings of symbolics by using the %SUBSTR function. %SUBSTR has three operands: start, length, and value, for example, SUBSTR(3,4,%TIME). The following conditions apply when using substrings:

- The operands may contain symbolics, calculations, or literals.
- If the start operand has a value of 0, it will be changed to 1.
- The %SUBSTR( ) function is terminated when the closing parenthesis or a blank is found.
- The value after all other substitutions has a maximum length of 80 bytes. The resulting value will have its trailing blanks removed.

For example, you can generate a data set that has the time as part of its name, for example, T%SUBSTR(3,4,%TIME) generates the hour and minutes as part of the data set name. The name will be in the format THHMM (where T is a constant, HH is hours, and MM is minutes), rather than including the seconds (formatted as THHMMSS).

Utilities Referenced by @DEFAULT

The following table lists utilities referenced by the @DEFAULT model ID:

Utility	Definition
BIND	BIND control statements
CHECK	IBM Check Data utility control statements
CHECK_R	IBM Check Data utility control statements with recovery
COPY	IBM Image Copy utility control statements
COPY_PC	IBM Image Copy utility control statements generated pre-SQL (example of optional user utility)
COPY_R	IBM Image Copy utility control statements with recovery
FCHECK	CA Fast Check Data Utility control statements
FLOAD	CA Fast Load control statements
FLOAD_R	CA Fast Load control statements with recovery
FRECIXAL	CA Fast Recovery Index All
FRECOVIX	CA Fast Recovery Index
FUNLD	CA Fast Unload control statements
FUNLD_R	CA Fast Unload control statements with recovery

Utility	Definition
BIND	BIND control statements
LOAD	IBM Load utility control statements
LOAD_R	IBM Load utility control statements with recovery
PDASTATS	CA Database Analyzer statistics collection
QCOPY	CA Quick Copy utility control statements
QCOPY_R	CA Quick Copy utility control statements with recovery
REBIND	REBIND control statements
RECIXALL	Rebuild Index All
RECOVIX	IBM Rebuild Index utility control statements
REORG	IBM Reorg Utility control statements for tablespaces
REORG_R	IBM Reorg Utility control statements for tablespaces with recovery
REORGI	IBM Reorg utility control statements for indexes
REORGI_R	IBM Reorg Utility control statements for indexes with recovery
RREORG	CA Rapid Reorg control statements for tablespaces
RREORG_R	CA Rapid Reorg control statements for tablespaces with recovery
RREORGI	CA Rapid Reorg control statements for indexes
RREORGI_R	CA Rapid Reorg control statements for indexes with recovery
RUNSTATS	IBM RUNSTATS utility control statements
UNLOAD	Batch Processor unload control statements
UNLOAD_R	Batch Processor unload with recovery
XUNLD	Excluded optional CA internal unload model **Note:** This utility is used automatically by unload processing when CA Fast Unload has been selected, but it does not support a data type conversion being requested in the unload. A Batch Processor unload is done, and the output is written in a format that CA Fast Load can use to load the table.
XUNLD_R	Excluded optional CA internal unload model with recovery **Note:** This utility is used automatically by unload processing when CA Fast Unload has been selected, but it does not support a data type conversion being requested in the unload. A Batch Processor unload is done, and the output is written in a format that CA Fast Load can use to load the table.

Model Utility Control Cards

The Utility Control Cards are basic default utility models. They can be customized based on your shop's standards.

Note: For details about the IBM Utility Syntax, see the IBM *DB2 Command and Utility Reference*. For details about the CA utility syntax, see the reference guide for that utility.

An example of a change that you might need to make is the SORTWK allocation for the load processing. There are two methods of allocating SORTWKs; one is to use dynamic allocations (default), and the other is to use SORTWK DDs in the JCL. These two methods are outlined in Dynamic Allocation Processing and SORTWK DD JCL Allocations.

Dynamic Allocation Processing

Currently, the LOAD, CHECK, and REORG models are defined using SORTDEVT, which causes the load utility to dynamically allocate SORTWKs. The amount of SORTWKs is determined by the SORTNUM parameter, which defaults to 0. The value of 0 tells the SORT package to use the SORT default allocations. In almost all shops these allocations are small, and cannot handle load processing with large numbers of records. This can cause the load to fail with ABEND B37, or insufficient core storage abends. These abends can be alleviated by increasing the SORTNUM value in the LOAD control cards. The SORTNUM value is a symbolic variable that can be updated through model services.

To update the SORTNUM symbolic variable through model services:

1. Access the Model Services screen using the Profile option.
2. You can edit utilities only in a model that you have the authority to update. If you have been using the @DEFAULT model, you must make a template of it before you can add a new model ID.
   - Enter **T** (for template) beside @DEFAULT.
   - Provide a new model ID for the templated model.
3. Edit the symbolics in the LOAD utility. Enter **S** in the CMD field next to the LOAD utility. The General Model Symbols screen appears, displaying the symbolics for the LOAD utility.
4. Change the replacement value for SORTNUM from 0 to a value from 0-9. This tells the number of temporary data sets to be dynamically allocated.
5. At analysis time, specify **Y** for Update Options, and specify the newly templated model to be used for analysis processing. Your analysis output will now have LOAD Utility Control Cards that include the SORTNUM parameter.

SORTWK DD JCL Allocations

The second method of allocating SORTWKs is to update the model, delete the SORTDEVT value, and define SORTWK DDs in the model JCL. This causes the SORT package to use the specified SORTWK DDs defined in the execution JCL, and not to use dynamic allocations.

To define SORTWK DDs in the model JCL:

1. Edit the model JCL. The member to edit is called MJBPMDL in *high-level*.CDBAMDL. Add applicable SORTWK DDs according to your shop standards.
2. Access the Model Services screen using the Profile option.
3. You can edit utility models only in a model ID that you have the authority to update. If you have been using the @DEFAULT model, you must make a template of it before you can add a new model ID.
   - Enter **T** (for template) beside @DEFAULT.
   - Provide a new model ID for the templated model.
4. Edit the utility model and delete the %SORTDEVT reference.
5. At analysis time, enter **Y** for Update Options, and specify the newly templated model to be used for analysis processing. Your analysis output will now have LOAD Utility control cards without the SORTDEVT variable. When you generate the JCL to run the output, the appropriate SORTWK DDs will be added because of your edits to the model JCL.

Many other changes similar to these can be made in any of the utility models for IBM utilities or CA utilities. Remember:

- You must template and create your own model when changing symbolics. Only the owner of @DEFAULT can update this model ID. This model ID also gets reinstalled with each new release, and any changes will be lost.
- At analysis time, specify using the model ID that you created.

The current DB2 SSID is now displayed on all screens where appropriate. This is for informational purposes only, so that the user knows which subsystem is being accessed.

Model Utilities Work Data Sets

All CA and IBM utilities require special data sets for the various phases of their processing. The data sets are the same as those required for a stand-alone utility execution, but there are two differences. One is that, when being executed through the CA Batch Processor, dynamic allocations are used (that is, .ALLOC FI(DDNAME) instead of //DDNAME DD....). The other is that there are optional keywords that allow the default data set names to be different. Following is a list of the DDs and how they correspond to each of the utilities.

IBM Utility Data Sets

This following table shows how the DDs correspond to each of the utilities:

DDNAME	Used By	Utility Name
PTIOUT=SORTOUT	IBM Check Data	CHECK CHECK_R
	IBM Load	LOAD LOAD_R
	IBM Reorg	REORG REORGI
	CA Fast Load	FLOAD FLOAD_R
	CA Rapid Reorg	RREORG RREORGI
PTIUT1=SYSUT1	IBM Check Data	CHEC CHECK_R
	IBM Load	LOAD LOAD_R
	IBM Reorg	REORG REORGI
	IBM Recover Index	RECOVIX
	CA Fast Load	FLOAD* FLOAD_R*
	CA Rapid Reorg	RREORG RREORGI
SYSERR=SYSERR	IBM Check Data	CHECK CHECK_R
	IBM Load	LOAD LOAD_R
	CA Fast Load	FLOAD FLOAD_R
	CA Rapid Reorg	RREORG RREORGI
SYSDISC=SYSDISC	IBM Load	LOAD LOAD_R
	CA Fast Load	FLOAD* FLOAD_R*

DDNAME	Used By	Utility Name
SYSMAP=SYSMAP	IBM Load	LOAD LOAD_R
	CA Fast Load	FLOAD* FLOAD_R*
SYSCOPY=SYSCOPY	IBM Copy	COPY COPY_R COPY_PC
	CA Quick Copy	QCOPY COPY_R
PTILOAD=SYSREC	IBM Load	LOAD LOAD_R
ROUNLDDN=SYSREC	IBM Reorg	REORG REORGI
PTIREC=SYSREC	CA Fast Load	FLOAD FLOAD_R1
	CA Fast Load	FLOAD* FLOAD_R*
RRUNLD=SYSREC	CA Rapid Reorg	RREORG RREORGI
SYSREC01=SYSREC	CA Fast Unload	FUNLD FUNLD_R

*These DD statements are not allocated for BUILD-type Fast Loads.

Data set names for CA utilities and their correlation to the IBM utility data set names are further documented in each utility's user guide.

Model Utility Data Set Calculations

The space needed for utility data sets is calculated using the %CALC Space Calculation Symbolic. To perform the space calculations, these symbolics are used: %REORGP, %ROSORT, %ROSYSREC, %ROSYSUT1, %ROSORTWK, %NROWS, %ROWLEN, and %BYTETRK. For the specific calculation used for a data set, use Model Services to review the symbolic replacement value for the symbolic.

The following table lists DDs used for the allocation type, primary allocation, and secondary allocations for the utility data sets:

DD	Use
SYSERR	ERRSP1 = Primary Allocation ERRSP2 = Secondary Allocation ERRSP3 = BLK(4096) allocations for IBM utilities and hard coded value of TRACKS for CA utilities.
PTIOUT	OUTSP1 = Primary Allocation OUTSP2 = Secondary Allocation OUTSP3 = CYL
PTIUTI	UT1SP1 = Primary Allocation UT1SP2 = Secondary Allocation UT1SP3 = CYL
PTIREC	RECSP1 = Primary Allocation RECSP2 = Secondary Allocation
SYSDISC	DISCSP1 = Primary Allocation DISCSP2 = Secondary Allocation DISCSP3 = Tracks
SYSMAP	MAPSP1 = Primary Allocation MAPSP2 = Secondary Allocation MAPSP3 = BLK(4096) for IBM Utilities and hard-coded value of TRACKS for CA Utilities
COPYSP1	COPYSP1 = Primary Allocation COPYSP2 = Secondary Allocation COPYSP3 = BLK(4096)
RRUNLD RRUNLDDN	ROSP1 = Primary Allocation ROSP2 = Secondary Allocation ROSP3 = CYL
SYSREC01	ALLOC1 = Primary Allocation ALLOC2 = Secondary Allocation

Model Services Frequently Asked Questions

Frequently asked questions regarding Model Services involve the following subjects:

Changing Utility Models by Exclusion and Inclusion

Question:

How can I instruct model services to use a different utility? For example, how can I use CA Rapid Reorg instead of IBM REORG?

Answer:

To use CA Rapid Reorg instead of IBM REORG, perform the following steps:

1. Use the Profile option to access the Model Services screen.
2. Enter **U** (update) next to the model ID to be changed. The General Model Utilities panel appears.
3. Exclude the REORG utility by entering **X** in the CMD field beside REORG.
4. If the RREORG utility (CA Rapid Reorg) is excluded, unexclude it by entering **X** in the CMD field beside RREORG.

   Note: If the SYM column is set to X, the utility is excluded.
5. Press Enter to process your input.

Once you have completed these steps, Model Services will invoke CA Rapid Reorg to perform reorganizations whenever reorganization is needed.

Adding a DCLGEN Utility

Question:

I want to add a DCLGEN utility to my model ID. What would the member look like?

Answer:

To add the DCLGEN utility, perform the following steps:

1. Access the Model Services screen, using the Profile option.
2. Enter **U** (update) next to the model ID name you want to change.

   The General Model Utilities panel appears. This screen lists all the current utilities for this model ID.
3. Enter **I** (insert) in the CMD field of the line directly above where you want to insert the DCLGEN name.

   A blank line appears.

   Note: If you enter **E** to edit the DCLGEN, an ISPF session appears immediately. See step 6 for the utility commands to enter.
4. In the blank line, enter the following:
   - **DCLGEN** in the Utility field
   - **T** in the OB (Object) field
   - **Generated Declarations** in the Description field
5. Press Enter. An ISPF session appears.
6. Enter the commands for the utility. For example, enter the following:

```
#IF(%SQLPOSA)
.CALL DSN PARM(%TOSSID)
.DATA
  DCLGEN TABLE(%OBJECT) -
  LIBRARY('%DCLLIB')    -
  ACTION(ADD)           -
  LANGUAGE(COB2)        -
  STRUCTURE(%OBJECT)
.ENDDATA
#ENDIF
```

7. When finished, press PF3 (or enter the END command) to end the edit session. The General Model Utilities panel reappears.

   The DCLGEN utility now has an I (for Invalid) in the SYM field, to indicate that the utility contains a user-defined symbolic that has no value (in this case, %DCLLIB).

   Note: You can call the General Model Symbols screen to specify values for the symbolic variables that appear in the commands you entered.

8. Enter **S** in the CMD field beside the new utility name, and press Enter. The General Model Symbols screen appears, listing the user-defined symbolic %DCLLIB and the utility commands. This screen lets you enter the replacement values.

9. Enter **%USERID..DCLGEN.LIB** in the Replacement Value field beside the symbol DCLLIB.

10. Press PF3 (End) to save the replacement value and return to the General Model Utilities panel. The Last Update field will reflect the last action, which was entering the replacement value for the DCLLIB symbol.

11. Press PF3 to return to the General Model Services screen. The new utility DCLGEN is saved as a part of the model ID you updated. In the future, when you choose that model ID, the DCLGEN utility will be used.

You can add other utilities to this or another model ID using the same steps.

Logging Based on Unload Size

Question:

I want to use conditional logic in a model utility to determine whether logging should be performed. How can I do this?

Answer:

You can include the following statements in the LOAD model utility. Logging will be performed only if the size is smaller than %VALUE.

```
#IF(%REORGP,<,%VALUE)
   LOG=YES
#ELSE
   LOG=NO
#ENDIF
```

Enter a replacement value for the %VALUE symbol on the General Model Symbols screen. (Enter **S** in the CMD line next to the load utility.)

Specifying an Expiration Date or Retention Period

Question:

I want to specify an expiration date on a tape. How can I do this?

Answer:

You can include an expiration date or a retention period on a tape by adding the EXPDT or RETPD keyword to the utility that writes data to tape. Both copy and unload utilities write data to tape.

To add RETPD to the copy utility, perform the following steps:

1. Access the General Model Services screen, using the Profile option.
2. You can only edit utilities in a model that you have update authority on. If you have been using the @DEFAULT model, you must make a template of it before you can add a new utility.

   a. Enter **T** (for template) beside @DEFAULT.

   b. Provide a new model ID for the templated model. You can change the description too.
3. Indicate that you want to update your model ID by entering **U** in the CMD field next to the model ID name.
4. Edit the COPY utility. Enter **E** in the CMD field next to the COPY utility. The COPY utility will be displayed.

5. Add the RETPD symbolic to the POS statement. The following example illustrates the modified COPY utility. See line 20:

```
 #IF(%SQLPOSB)
   .CALL UTIL COPY PARM(%FROMSSID)
   #IF(%RECOVERY)
     .ALLOC FI(SYSCOPY)+
            DA('%RCOPYDSN')+
   #ELSE
     .ALLOC FI(SYSCOPY)+
            DA('%COPYDSN')+
   #ENDIF
     #IF(%REORGP,<,%LARGE)
       NEW CATALOG UNIT(%COPYDUNT)+
       MGMTCLAS(%MGMTCLAS)+
     STORCLAS(%STORCLAS)+
     DATACLAS(%DATACLAS)+
     VOLUME(%COPYDVOL)+
     SPACE(%COPYSP1,%COPYSP2) %COPYSP3 RLSE
   #ELSE
     NEW CATALOG UNIT(%COPYTUNT)+
     VOLUME(%COPYTVOL)+
     POS(%INCR) LABEL(%LABEL) RETPD(%RETPD)
   #ENDIF
  .DATA
   #IF(%REORGP,<,%LARGE)
     COPY TABLESPACE %CREATOR..%OBJECT DEVT %DEVT
       COPYDDN  SYSCOPY
   #ELSE
     COPY TABLESPACE %CREATOR..%OBJECT DEVT %TAPE
         COPYDDN  SYSCOPY
     #ENDIF
   .ENDDATA
   .FREE FI(SYSCOPY)
 #ENDIF
```

6. Press PF3 (End) to end the edit session. The model utility will be marked as invalid, because there is no value for the new symbolic.

7. Update the symbolics in the COPY utility. Enter **S** in the CMD field next to the COPY utility on the General Model Utilities panel, and press Enter. The General Model Symbols screen appears.

8. The new variable, %RETPD, is included in the list of variables. Enter the value in days.

   Note: Enter the value for %EXPDT in Julian date format (yyddd).

9. Press F3 (End) to save the symbolic value.

10. Press F3 (End) again. The changed model ID will be saved.

Sending the Unload Data Set to Tape

Question:

I want to send the unload data set to tape. How can I do this?

Answer:

Model Services has logic in the Unload (UNLOAD) and CA Fast Unload (FUNLD) models to perform the unload to tape based on the size of the table. To force the unloads to always go to tape, set the LARGE symbol to 0.

Chapter 6: Drop Recovery

This section contains the following topics:

Drop Recovery Component

Drop Recovery is a facility that allows recovery of information contained in dropped DB2 objects. With Drop Recovery, you can use the following to drop objects:

- Standard drop (S)
- Standard drop with options (SO)
- Recoverable drop (R)
- Recoverable drop with options (RO)

Note: Before an object can be recovered, it must have been dropped using one of the recoverable drop options.

Drop Impact Analysis allows impact assessment before dropping an object by previewing a list of dependent objects that will also be dropped, as well as related aliases that will not be dropped.

Dropping an Object

Recoverable DDL is written according to the type of drop used. There are two types of drops: standard drops and recoverable drops.

- The standard drop is similar to the DB2 DROP statement in that it permanently drops the selected DB2 objects and dependents from the DB2 catalog.
- The recoverable drop allows users to drop the DB2 object and its dependents from the DB2 catalog, but the dropped object's recovery information is stored in a CA recovery table. This means that a recoverable drop permits recovery of the dropped object and its dependent objects. (The term recovery information is used to mean all the information saved at drop time. This can include the create and grant statements, control statements necessary to execute utilities after the data has been recovered, and the object's unloaded data.)

Standard and recoverable drops are important, because they allow the choice of whether the object can be recovered later, and provide control over how the object is dropped.

The opportunity is also provided to evaluate whether to drop an object by viewing a list of all dependent objects that would also be dropped. This list is generated when the Drop Impact Analysis option is selected for a standard drop or a recoverable drop. Based on this impact list, you can proceed with the drop or cancel it.

Recovering an Object

You can recover dropped DB2 objects by using the Drop Recovery feature. Drop Recovery references the CA recovery table to recover the dropped object and its dependents. The following list describes some of the other features Drop Recovery offers:

- Broad Selection Listing. The Object field can be left blank so that an across-the-board listing of all recoverable objects appears. Otherwise, a specific object type can be entered in the Object field for a more tailored listing.
- Recover Select. This option lets you select the objects to recover. The saved recovery information from the dropped object is retrieved from the CA recovery table and written to a user-specified data set.
- Recover Delete. This option lets you delete stored recovery information from the CA recovery table. Once the object's recovery information is deleted, it is no longer recoverable.
- Explode. Use the EXPLODE command to view the dependent objects to be affected by the recovery or deletion of the dropped object.

Standard and Recoverable Drops

When you drop an object (see page 105), you can do either of the following:

- Drop an object using a standard drop (S) (see page 105)
- Drop an object using a recoverable drop (R) (see page 105)

Drop a DB2 Object

You can drop a DB2 object if needed.

To drop a DB2 object, complete the following fields on the CA RC/Update main menu:

- Type **D** (for Drop) in the Option field.
- Specify the object type in the Object field.
- (Optional) Specify values for the Item Name and Creator fields.

Press Enter.

The Drop Selection panel appears. An object can be dropped using a standard or recoverable drop from the Drop Selection panel.

Note: The process used to drop referential integrity rules is slightly different. For more information, see the “Referential Integrity” chapter.

Drop an Object Using Standard Drop (S)

A standard drop is like a DB2 drop: the object (and any data or dependent objects) is permanently dropped.

To drop an object using a standard drop enter **S** next to each object to be dropped. These objects are not recoverable once dropped.

Drop an Object Using Recoverable Drop (R)

A recoverable drop drops the object from the DB2 catalog, but saves the object's definition. This enables you to recover the object (and any dependent objects or data). To drop an object using a recoverable drop, enter **R** next to each object to be dropped (and recovered).

Recover the object by using the Drop Recovery feature.

More information:

Recovering an Object (see page 104)

Dropping Indexes

When you select an index to be dropped, CA RC/Update determines whether the index contains a unique constraint. If the index does contain a unique constraint, the CA RC/Update Default Drop Analysis screen appears. Use this screen to analyze the drop and set DDL options.

ISPF Profile

There are also ISPF Profile values that govern how drops are made. For standard and recoverable drops, these options determine whether user-defined VSAM data sets are deleted and whether Drop Impact Analysis is invoked. For recoverable drops, the profile determines if data, the object's security, and other information associated with the drop and recovery of the selected object should be saved.

With either type of drop, the default ISPF profile values can be used, or a request can be made to view (and possibly change) the values. To use default information, select objects using the S and R options. To view and change the drop profile values, select objects using SO (standard/options) and RO (recoverable/options) instead of S and R. Any changes made to the Drop Recovery parameters are used for the current object drop. Options can also be saved to a user's ISPF profile.

More information:

Standard Drop Options (see page 108)

Drop Processing Flow

When users select a combination of the drop options from the Drop Selection panel, the standard drops are processed first and recoverable drops second.

Standard Drops

All standard drops (S and SO) are processed first in the order the objects are displayed on the Drop Selection panel. For Standard Drops (S), the ISPF profile values are used to determine whether user-defined VSAM data sets are deleted.

A Drop Options screen displays for each object selected with SO. This screen permits the review and change of the Delete VSAM Dataset value and Display Drop Impact value for this object. The ISPF profile value can be changed at this time.

If Drop Impact Analysis is selected, the Drop Recovery Analysis Wait screen appears while the impact of the drop is analyzed, and then the Drop Impact List screen appears.

One Drop Confirmation screen appears for all standard (S and SO) drops. At the Drop Confirmation screen, the drops can be accepted, edited, or canceled. Press Enter to process the object's drop, enter **EDIT** to edit the object's DDL, or press F3 (End) to cancel the drop.

Recoverable Drops

CA RC/Update returns to the top of the Drop Selection panel list and processes all recoverable drops (R and RO) according to the order the objects are listed on the Drop Selection panel.

For each object selected with R, the Drop Recovery Analysis Wait screen appears while the effect of the drop is analyzed. (The ISPF profile values are automatically used.)

Note: The CA General Model Services screen appears first if you need to specify a model ID to use.

If Drop Impact Analysis is invoked, the Drop Impact List screen appears. A Drop Confirmation screen appears for each object selected with RO.

For each object selected with RO, the Drop Recovery Options screen appears, allowing retention of or changes to the Drop Recovery values. Press F3 (End) once the Drop Recovery values have been confirmed. The Drop Recovery Analysis Wait screen appears while the effect of the drop is analyzed. If Drop Impact Analysis is invoked, the Drop Impact List screen appears. A Drop Confirmation screen appears for each object selected with R.

At the Drop Confirmation screen, the Drop/Recovery Parameter values it will use for that object drop are displayed. Press Enter to drop the object, enter **EDIT** in the command line to edit the DDL, or press F3 (End) to cancel the drop.

Drop Processing Summary

To summarize, the S or SO drops are processed, and then the R or RO drops are processed.

Standard Drops

Standard drops involve the following:

- Selecting a standard drop (see page 108)
- Standard drop options (see page 108)
- Drop Recovery Analysis Wait screen (see page 109)
- Drop Impact List screen (see page 109)
- Standard Drop Confirmation screen (see page 109)

Selecting a Standard Drop

To drop an object, enter **D** in the Option field and enter the object type in the Object field. The Drop Selection panel appears, from which objects can be selected for the drop.

The columns that appear on the Drop Selection panel vary according to the Object value.

Standard Drop (S)

Enter **S** to permanently drop the selected objects. The Delete VSAM Datasets value assigned in your ISPF profile is automatically used for processing. To change the value of the defaults, use the SO option.

Standard Option Drop (SO)

Press Enter to display a Drop Options screen for each selected object before the selected objects are permanently dropped. The object's Delete VSAM Datasets value or the Display Drop Impact value can be changed at the Drop Options screen. If the values are saved, the user ISPF profile will be updated.

Standard Drop Options

The Drop Options screen appears when an SO drop is processed. This screen displays the Profile's current values for Delete VSAM Datasets and Display Drop Impact, where these values can be changed or retained. Replace the ISPF profile's current Drop Options values with the changed values by entering SAVE in the command line.

Note: For information about the fields, press F1 (Help).

Press F3 (End) to process the drop options. If multiple SO drops have been selected, the next SO option screen appears. If there are no more SO drops waiting to be processed, the Drop Confirmation screen appears.

To cancel the processing from the Drop Options screen, enter **CANCEL** in the command line. If there are selected objects remaining to be processed, a message appears asking if processing should continue for the remaining queued selections.

- Enter **Y** to cancel processing all the selections and to return to the Drop Selection panel.
- Enter **N** to cancel only the selected object's processing and to continue processing the remaining selected objects.

Drop Recovery Analysis Wait Screen

If Drop Impact Analysis has been invoked (the option Display Drop Impact set to Y), the Drop Recovery Analysis Wait screen appears.

Note: The Drop Recovery Analysis Wait screen appears while Drop Impact Analysis is being processed, even if a Standard Drop is being processed.

A Standard Drop with Drop Impact Analysis is still nonrecoverable.

Drop Impact List Screen

If Drop Impact Analysis has been invoked, the Drop Impact List allows users to preview all objects to be dropped because they are dependent upon the object being dropped.

Note: For information about the fields, press F1 (Help).

Standard Drop Confirmation Screen

The standard Drop Confirmation screen appears after all S and SO drops process. The Drop Confirmation screen collectively lists the drop DDL statements for all the S and SO drops.

Note: For information about the fields, press F1 (Help).

Standard drops (objects selected with S and SO) are processed first.

Press Enter to accept the DDL and permanently drop the selected objects. Enter EDIT in the command line to edit the DDL.

Press F3 (End) or enter CANCEL in the command line to cancel the drop. After the S and SO drops process, any selected R and RO drops are processed.

Perform a Recoverable Drop of an Object

You can perform a recoverable drop. For this kind of drop, an object is dropped from the DB2 catalogs, but you can recover the object later.

Follow these steps:

1. Complete fields as follows on the CA RC/Update Main Menu:
   a. Type **D** in the Option field.
   b. Specify the object type in the Object field.

Press Enter.

The drop selection panel appears.

2. Customize drop options for objects by typing **RO** next to each object.

   The Drop Recovery Options panel appears, where you can customize the options for conducting the recoverable drop. For example, you can invoke Drop Impact Analysis to determine which objects the product will drop because they are dependent on the object being dropped.

   Changing the drop recovery options does not permanently change the ISPF profile values unless you enter SAVE in the command line.

   Note: The Drop Recovery Options panel is not available if Operation Mode is set to A (RC/Alter) on a CA RC/Update panel. To skip option customization and initiate a recoverable drop directly, enter **R** next to the object on the drop selection panel.

3. Change values as needed on the Drop Recovery Options panel (or retain the specified values), and then press the END key.

   A Drop Recovery Analysis Wait panel appears for each drop that the product processes. At any time, you can cancel all processing or cancel only the selected object processing. After processing occurs, a drop confirmation panel appears, showing recovery information (such as DDL and drop/recovery parameter values). You can scroll through the recovery information as needed.

4. Perform one of the following actions:

   - Press Enter to accept the information and drop the selected object.
   - Press the END key to cancel the drop.
   - Enter **IMPACT** in the command line to return to the Drop Impact List panel (if you requested Drop Impact Analysis).
   - Enter **EDIT** in the command line (to access the SQL Editor panel and edit the drop SQL).

     Note: You can submit the drop from the SQL Editor panel.
   - Enter LISTR in the command line to view information that the product uses to recover an object.

   If you accept processing, the product performs the drop. If necessary, you can recover the object.

Recover a Dropped Object

Objects dropped with a recoverable (R or RO) drop can be recovered with the Drop Recovery option. Drop Recovery references the CA recovery table to recover the dropped object and its dependents. This section demonstrates how to recover an object.

Follow these steps:

1. Enter **R** for Drop Recovery in the Option field.
2. Leave the Object field blank or enter an asterisk (*) to list all recoverable objects. This feature is unique to Drop Recovery. Otherwise, enter an object type to limit the recoverable objects listed.
3. Enter selection criteria or the specific object name in the Item Name field to limit the recoverable objects listed.
4. Enter selection criteria or a specific creator in the Creator field. However, if the object is a tablespace, enter the database name as the creator.
5. Enter **Y**, **S**, or **N** in the Where field. Access the Extended Query Facility (EQF) by entering Y or S. Enter **N** to refrain from accessing EQF.

   Note: For more information about EQF, see the *CA Database Management Solutions for DB2 for z/OS General Facilities Reference Guide*.
6. Press Enter. The Object Recovery List appears with the recoverable objects that match the header information. From here, you can execute Recover Select, Recover Delete, and Explode.

Processing Your Selection

A combination of the recover, explode, and delete options can be entered. Press Enter after the selections have been made. Processing begins.

Dependent Object Query

To list only the objects that have a specific dependent object, the Where clause can be extended with the Extended Query Facility (EQF) to retrieve the object listing. This is useful to see which objects contain a dependent object you want to recover.

To query for the dependent objects, enter **R** for Drop Recovery in the Option field and enter **Y** in the Where field. Leave the other header fields empty for a comprehensive search. Press Enter. The SQL Selection Panel appears.

For **Where Clause**, enter additional conditions to be applied to the SQL select statement (optional). The conditions are added to the existing WHERE clause. You can enter up to 45 lines of WHERE conditions. To view all lines, use the up (F7) or down (F8) scroll commands.

When entering conditions, the existing WHERE clause is being extended. Therefore, do not enter WHERE as part of the text because it is automatically added.

The table and column names list the CA recovery table's corresponding columns. DATA is the column containing the dependent object data.

The AND DATA LIKE '%PARTED_IX2%' statement indicates to search for all the dependent objects whose names contain PARTED_IX2. When writing a SQL select statement, use the following format:

```
AND DATA LIKE '%dependent object name%'
```

After entering Where clauses, press F3 (End). The Object Recovery List screen appears with the objects that contain the specified dependent objects.

Drop Recovery Processing Flow

If a combination of recover options is selected, the objects selected with the Recover Select (S) option are processed first in the order that they appear on the Object Recovery List. The Recovery DDL screen appears for each object selected for recovery.

After all objects selected with Recovery Select (S) have been processed, CA RC/Update searches for objects selected with a Recover Delete (D, DD, D#) option. The recovery information of all selected Recover Delete objects is processed collectively on one Recovery Delete Confirmation screen.

The Recovery Delete Confirmation screen allows confirmation or cancellation of the deletion.

- Confirm—Press Enter to process the deletions. The selected objects are not recoverable after confirming the deletion. A message appears stating that the delete request is being processed. When the deletion is complete, the Object Recovery List reappears.
- Cancel—There are three ways to cancel the deletion of an object. To cancel a specific object's deletion, space out the D from the selection line. To cancel all the deletions listed on the confirmation screen, press F3 (End) or enter CANCEL in the command line.

Objects selected with Explode (E) are processed after all selected Recover Select and Recover Delete objects are processed. Exit from any Drop Recovery screen by entering the CANCEL command in the command line or pressing F3 (End).

Note: For multiple selections of Recover Select (S), the CANCEL command terminates processing for all selected objects and returns you to the Object Recovery List screen. The F3 (End) key cancels processing for only the selected object.

More information:

Recover Select (see page 113)

Recover Select

The Recover Select (S) option allows recovery of an object dropped by a recoverable drop. A recoverable drop saves the object's recovery information in the CA recovery table. When an object is recovered, the recovery table is accessed, the object's saved recovery information is retrieved, and recovery information is written to a user-specified data set. The data set must be submitted to the Batch Processor for execution. Objects available for recovery are listed on the Object Recovery List screen. One, several, or all recoverable objects can be recovered. When selecting multiple objects, the objects process in the order they are listed on the Object Recovery List screen.

Recovery DDL Screen

From the Object Recovery List, select the objects available for recovery by entering S and pressing Enter. The Recovery DDL screen appears. This screen displays the DDL to be used to recover the object and its dependent objects.

Press Enter to allocate the data set and write the recoverable DDL to a data set. The Recovery DDL Dataset Allocation screen appears.

To cancel, press F3 (End). The next object's Recovery DDL screen appears if multiple objects have been selected. If there are no more objects left to process, the Object Recovery List appears.

Recovery DDL Dataset Allocation Screen

If Enter is pressed at the Recovery DDL screen, the Recovery DDL Dataset Allocation screen appears. At this screen, specify the data set to which the recovery information will be written. One data set must be specified for each object to be recovered.

Note: For information about the fields, press F1 (Help).

From the Recovery DDL Dataset Allocation screen, enter the information and press Enter, or press F3 (End) to cancel. After recovering an object, it remains on the Object Recovery List so it can be recovered again and saved to more than one data set. To remove the object from the Object Recovery List, use Recover Delete (D, DD, D#).

The data set with the recovery information must be submitted to the Batch Processor for execution to complete the recovery.

Note: For information about submitting the data set to the Batch Processor, see the *CA Database Management Solutions for DB2 for z/OS Batch Processor Reference Guide*.

Enter

Pressing Enter results in the recovery information being written to the specified data set.

- If multiple objects have been selected for recovery, the next object's Recovery DDL screen appears.
- If there are no more objects for recovery, any recoverable objects selected for deletion are processed. When there are no more selected objects, the Object Recovery List screen reappears.

F3 (End)

If F3 (End) is pressed to cancel, the canceled object's Recovery DDL screen reappears. Press Enter or press F3 (End).

- Press Enter to continue processing the object.
- Or, press F3 (End). If multiple objects have been selected for recovery, the next object's Recovery DDL screen appears.
- If there are no more objects for recovery, any recoverable objects selected for deletion are processed. When there are no more selected objects, the Object Recovery List screen reappears.

EXPLODE and SHRINK

The Drop Recovery EXPLODE command displays the recoverable object's dependent objects. This lets users determine which dependent objects will be affected if a recoverable object is recovered or deleted. The SHRINK command turns off the EXPLODE command.

Note: The EXPLODE function works only if, at Drop Recovery time, the Drop Impact option was set to Y. If this option was set to N, nothing appears for the Explode line command.

Use EXPLODE and SHRINK from the Object Recovery List screen. View the dependent objects for all recoverable objects listed at the Object Recovery List by using the EXPLODE primary command or view a specific recoverable object's dependents by using the EXPLODE line command. The SHRINK primary command turns off EXPLODE for all listed objects.

Note: For information about the fields, press F1 (Help).

Drop Recovery Conclusion

The Drop Recovery feature allows recovery of DB2 objects dropped by the recoverable drop.

A recoverable drop accesses the user's Profile's Drop Recovery Control Parameters to gather information about how the object and its dependents should be dropped and later recovered. A recoverable drop writes the object's recovery information to a CA recovery table.

Drop Recovery then accesses the recovery information through the CA recovery table. The recovery information from the recovery table can be either recovered or deleted when using the Drop Recovery feature. Drop Recovery has two options: Recover Select and Recover Delete.

- Recover Select writes the recovery information to a user-specified data set so the object and its dependents can be recovered.
- Recover Delete deletes the recovery information from the CA recovery table. Once the recovery information is deleted, the object is no longer recoverable.

Another feature of Drop Recovery is the EXPLODE command. EXPLODE lets users view the object's dependent objects so they can see what objects will be affected by the recovery or deletion of the recovery information. Drop Recovery gives users a second chance at recovering an object that would otherwise be forever deleted from the DB2 catalog.

Chapter 7: RC/Edit

This section contains the following topics:

RC/Edit Component

RC/Edit is a complete ISPF-style editor for manipulating information that is stored in DB2 tables. The data can be viewed in two different ways, column mode or form mode. All standard ISPF editor commands such as CHANGE, RCHANGE, FIND, and RFIND are supported. The extended view update support feature lets you edit most joined views. There is also an EXPLODE feature available for entering data into long character columns.

Used with the DB2 object create, drop, and alter capabilities, a complete environment for maintaining DB2 objects and data is provided, including the following.

- Access to DB2 data in Tabular (column) or Form mode. The data can be accessed in browse only mode or with update capability.
- Support for most DB2 data types includiing LONG VARCHAR, TIME, DATE, TIMESTAMP, TIMESTAMP with TIMEZONE, floating-point, graphical, and DBCS. All data entered is verified.
- Ability for multiple users to edit the same table at the same time using the Searched method of editing.
- An EXPLODE option for editing data greater than 256 characters. The EXPLODE option places the user into an ISPF full screen editor at the column level.

- Full support for Null values.
- Support for embedded blanks at the beginning or within values for the following items: column names, table names, and creators.
- Selection capability on columns and rows. You can specify an SQL statement to select only specific columns and rows, and save it for repeated use.
- A SORT feature for sorting the rows while they are being edited. This lets you constantly change the order of the data without re-executing the SQL statement.
- A FREEZE option for keeping columns fixed during scrolling. This facilitates editing columns dependent on other columns.
- Full ISPF line commands including insert, delete, undelete, and replicate. Block mode is supported.
- A K command to identify the primary (Pn) and foreign (FK) columns in a table automatically.
- FIND and RFIND commands to find specific data during the edit session.
- CHANGE, RCHANGE, CHANGE ALL, and SET commands to facilitate updates.
- A STATS command to view statistics any time during an edit session.
- PROTECT, UNPROTECT, XCOL, and SCOL commands for greater flexibility and control over the RC/Edit display.
- A SORT? command displays the Data Query Edit screen, where users can sort on multiple columns during the editing session.
- All ISPF scrolling modes (PAGE, HALF, Number, and CSR are supported).
- Support for the PPRINT, QPRINT, PFILE, and QFILE print commands.
- Support for 3278 model 5 terminals for viewing 133 columns (column mode only).
- Extended view update support feature to edit joined views on the EDIT screen using the Searched Column Mode. This is accomplished by mapping the columns in the views to the underlying base tables.
- Ability to edit a table that has the following types of columns:
  - Identity columns

    If the column is GENERATED ALWAYS, DB2 always assigns a value whenever a new row is inserted. If the column is GENERATED BY DEFAULT, you can specify a value to override the generated DB2 value.
  - ROWID columns

    You cannot specify an overriding value. Inserted rows always have ROWID values generated by DB2.

- LOB columns

  RC/Edit automatically omits the LOB columns of the edited table. You do not need to manually omit them. You can use RC/Edit to view and change the non-LOB columns of the table.

- XML columns

  RC/Edit automatically omits the XML columns of the edited table. You do not need to manually omit them. You can use RC/Edit to view and change the non-XML columns of the table.

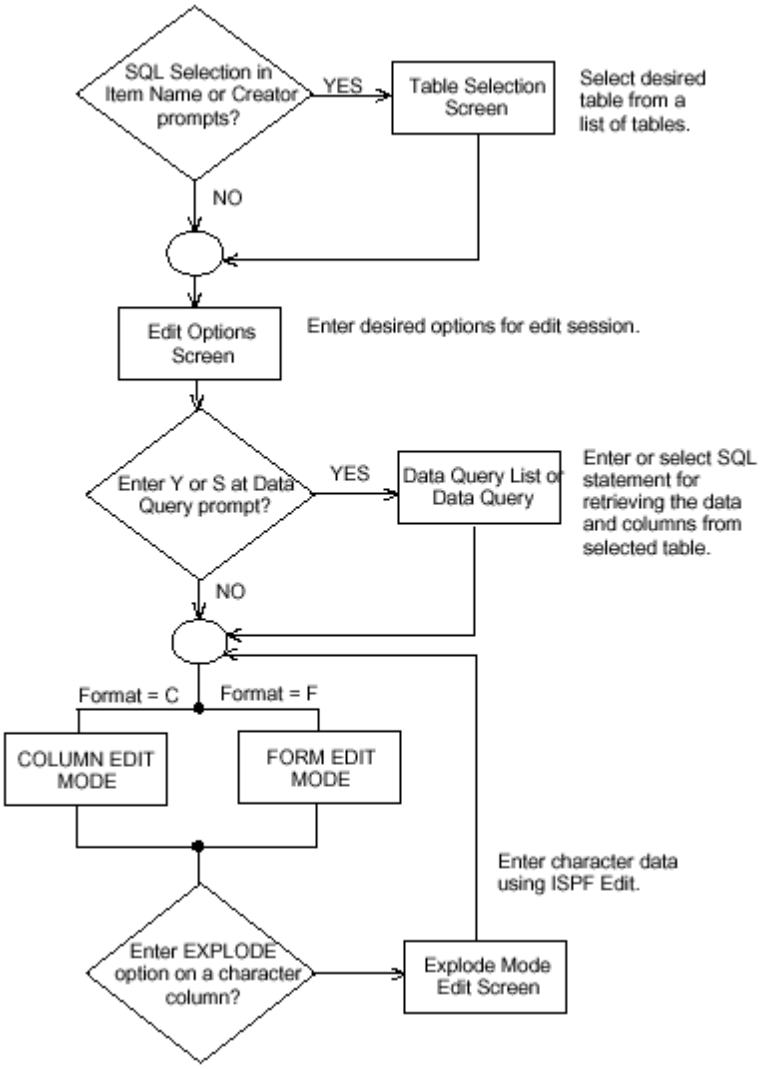
Screen Flow for RC/Edit

RC/Edit is composed of the following screens.

- Table Selection—Displayed when SQL selection criteria are input for the Item and Creator prompts. All tables matching the selection criteria and associated authorizations are displayed.
- Edit Options—Controls data to be edited in a table. Displayed for the E option. Can also be used with the fast edit (FE) option.
- Statistics Summary—Displays editing statistics during an edit session when the STATS command is entered. Pending statistics (the number of updates, deletes, and inserts that will be done if the user saves the data) are included on this screen.
- Data Query List, Data Query Edit—The Data Query List screen permits users to select a previously saved SQL statement for execution, edit, or deletion. The Data Query Edit screen is used to create or edit an existing SQL statement. All column names are listed and any WHERE clause can be input. Replaceable parameters are also supported.
- Edit Screens—Column Mode and Form Mode displays. The Column Mode displays the data in tabular form (columns appear horizontally across the top of the screen) and the Form Mode displays one row per screen (columns appear vertically down the left side of the screen).
- Explode Mode—Permits character columns to be edited using the ISPF full screen editor. Placing the cursor on a character column and pressing the EXPLODE PF key will explode the column's data into a full screen edit session, allowing easy editing of LONG VARCHAR data and data copying into character columns from other sources.

Processing Flow

The following diagram depicts a possible processing flow:



Locking Considerations

Locking is not a concern in browse mode.

Positional Update Method

The Positional Update method updates only one row. The row that is updated is determined by the current position of a cursor. The rows are refetched in their original order to get to the row that needs to be updated, as in the following statements.

```
UPDATE DSN8320.EMP SET WORKDEPT = :HOST1
    WHERE CURRENT OF CURSOR1
    HOST1 = 'B01'
    CURSOR1 is defined as follows:
    DECLARE CURSOR1 CURSOR FOR
        SELECT EMPNO, WORKDEPT
            FOR UPDATE OF WORKDEPT
```

Searched Update Method

When editing a table using the Searched Update method, it cannot be locked. The table name is enqueued in its own internal queue. No DB2 locks are placed on the table, but RC/Edit will not permit another user to edit the table using the Positional Update method.

Multiple users can edit the same table as long as they all use the Searched Update method. If another user attempts to edit the table using the Positional Update Method, a message appears to both users informing them of the contention. (In a data sharing environment where the other user is on a different z/OS system, you will see the message, but the other user will not.)

Data Type Verification

When editing data, RC/Edit performs type verification and full range checking based on the column's DB2 data type. Specific data verification rules follow:

- General Numeric Rules—All numeric values must be within the defined numeric range. For example, a SMALLINT of 99,999 cannot be entered. Type over any existing commas or decimal points. They will automatically be reinserted. Commas are optional.
- Character Data—The following points apply to character data.
  - Maximum Length. Data cannot be entered that exceeds the field's maximum length.
  - CHANGE. If data exceeds the field's maximum during a CHANGE, processing stops at the field that would cause the truncation error.
  - CHANGE ALL. The fields that could be changed (no truncation errors) are highlighted.
  - EXPLODE. The data is truncated upon return from the ISPF session.

- VARCHAR. Two options are available for processing trailing blanks when editing VARCHAR and LONG VARCHAR columns with an EDIT screen.

  First, when updating columns with RC/Edit, any trailing blank spaces are automatically deleted for selected VARCHAR columns to save storage space. The trailing spaces are deleted if any columns in the row are updated, regardless of whether the VARCHAR column is updated. For rows with no column updates, the VARCHAR columns are not changed.

  Second, you can preserve trailing blanks using delimiter characters set by the user on the RC/Edit Parameters screen. The default opening and closing delimiter characters are the less than (<) and greater than (>) characters, respectively.

  Data strings can contain embedded delimiter characters. To begin and end a data string with delimiter characters (for example, '<abc >'), insert the opening and closing delimiters as usual (for example, '<<abc >>').

  When fetching rows, RC/Edit automatically inserts the delimiter characters if a column's data contains leading and/or trailing blanks, or if the first and/or last data character is one of the delimiter characters. However, when entering data through RC/Edit, you must enter the delimiter characters yourself to preserve trailing blanks and data that begins and ends with delimiter characters.

- CAPS ON/OFF. Use the CAPS ON/OFF command to control the case. Whole columns can be converted to upper or lower case by entering the LCASE or UCASE commands.

  The display length of character fields does not affect the FIND, RFIND, CHANGE, RCHANGE, and CHANGE ALL command. The complete column's length is searched regardless of the display length. To view the whole column, use the EXPLODE command.

- BIGINT—Value must be from –9223372036854775808 to 9223372036854775807.
- SMALLINT—Value must be from -32,768 to 32,767.
- INTEGER—Value must be from -2,147,483,648 to 2,147,483,647.
- DECIMAL—The number of digits are automatically truncated to the right of the decimal point based on the column's definition. Value must be from +/- 9,999,999,999,999,999,999,999,999,999,999.
- FLOAT, REAL, DOUBLE PRECISION—Data must follow exponential notation. If the number is entered without exponential notation, it will automatically be converted. Type over the decimal point during entry and it will automatically be renormalized with the corresponding change made to the exponent. Value must be approximately from +/- 7.2E+75 to +/- 5.4E-79.

- DATE—The entry must follow standard date rules. The cursor will be positioned on the invalid entry within the date field, such as an invalid month. The date format is DB2 installation dependent. Value must be from 0001-01-01 to 9999-12-31.
- TIME—Entry must follow standard time format. The cursor will be positioned on the invalid field. The time format is DB2 installation dependent. Value must be from 00.00.00 to 24.00.00
- TIMESTAMP—Same rules as above for DATE and TIME. Value must be within above ranges.

More information:

RC/Edit Command Reference (see page 148)

Inserting Rows

When you insert a row, all fields are given an initial value based on the field type.

- Character Data (CHAR and VCHAR)—Spaces.
- Numeric (all types)—Zeros.
- Date—Current date.
- Time—Current time.
- Timestamp—Current date and time.
- Identity column—If the column is generated always, a value is always supplied by DB2. If the column is generated by default, you can specify a value that RC/Edit uses to override the DB2 generated value.

Entering Data

This section describes the guidelines for you to follow when entering data:

- To enter a new value for a field, enter the number and press EOF to clear the remaining data.
- To zero out a field, press EOF and press Enter. A zero value will automatically be inserted.
- To make a value negative, enter a minus sign anywhere within the blank part of the field.
- To update a number, cursor over and type over the existing values. Type over commas. RC/Edit automatically reinserts commas and decimal points when Enter is pressed.

- Insert character data by using the insert key. All fields have implied NULLs at the end.
- In Column Mode, all numbers are right justified. The cursor is positioned at the left. To append digits to the front of a number, enter the digits and press Enter. The digits automatically move to the front of the number.
- In Form Mode, all values are left justified, so all data is aligned. Press the INS key to add digits to the front of a number.
- To enter data values that are longer than the display area, specify the EXPAND primary command, place your cursor in the area to be expanded, and press Enter.

Null Values

If a field is defined as NULL, then the RC/Edit Null Indicator (see Column Mode Edit or Form Mode Edit) will be Y for these fields. Default values are placed into NULL fields to provide templates and default values. Remember, if the Null Indicator is Y, the field is considered null. To enter a not null value, the NULL indicator must be set to N. For fields defined as NOT NULL or NULL WITH DEFAULT, the above default values will always be applied.

If specific table columns are selected for editing, the remaining, non-selected columns can be defined with the NOT NULL attribute. In this case, inserts or replicates cannot be performed during the edit session. This restriction is applied because any rows inserted to DB2 would fail on a NULL error.

GENERATED Values

If an identity column is GENERATED BY DEFAULT, a G indicator appears before the column. The column will have a value of Y or N for an inserted row:

- Y—The field will contain dashes. When the row is inserted into the database, DB2 will assign a value to the field.
- N—The field will contain a valid integer value. When the row is inserted into the database, DB2 will not assign a value; it will use the value specified by the user.

If the row was fetched from the database, the column will have a value of N for the G indicator. If you change N to Y, the change is ignored and the value is set back to N. (The G indicator will only show Y if the row has not yet been inserted into the database; DB2 generates a value for the field when the row is inserted into the database.)

An inserted row shows I in the Opt field at the left of the line and a fetched row shows some other value other than I in this field.

HEX Mode Support

RC/Browse accepts and processes columns with hexadecimal data. If data is retrieved that contains nondisplayable (hex) characters, the hex characters are displayed as periods and the message HIGHLIGHTED CHAR TYPE COLUMNS CONTAIN INVALID (NONDISPLAY) CHARACTERS appears.

To browse the hex characters, use the RC/Browse column mode screens or the RC/Browse Explode Facility screen. ISPF Browse will be invoked with HEX on if there are nondisplayable characters. Toggle the ISPF command off and on by entering HEX in the command line.

More information:

RC/Browse Command Reference (see page 200)

Updating the Actual Data

While in the edit session, no changes are actually made to DB2 until the data is saved. Changes are committed to DB2 by using primary edit commands. These commands can be entered on the command line or assigned to function keys.

Update Order

Table updates are applied differently in the two editing methods, Positional Update and Searched Update.

Positional Method

When RC/Edit applies updates using the Positional Update method, they are applied in fetch order, NOT in the current display order. The order of updates is inconsequential except for the case of unique indexes on columns. For example, a user can delete a row that has a primary key value of 5, and then update another row, assigning the same primary key of 5. If the delete does not occur before the update, a duplicate error will occur.

As a result, there may be need to make some updates involving unique indexes by using multiple edit sessions or issuing the SAVE command to commit the index changes during the edit session.

Searched Method

The Searched Update method updates the rows in the order they are displayed on the screen. The Searched method uses only half as many fetches as the Positional method.

Table Selection

The Table Selection facility permits table selection from a list. You can control the list of tables displayed by entering selection criteria for the Item Name or Creator prompts that appear in the header. An extended query can also be entered to further refine the list of displayed tables. (If a specific table name is entered, this screen is not displayed.)

Note: Embedded blanks are supported in table and creator names. When specifying a value for Item Name or Creator, you can also embed blanks at the beginning or elsewhere in the value.

Column Updates

When columns are updated, the processing updates *all* columns in the DB2 tables rows. This can cause DB2 triggers that should only be executed if specific columns are updated to execute anyway. To prevent this from occurring, add the following to the CREATE TRIGGER DDL:

- The SQL clause REFERENCING OLD AS OROW
- A WHERE condition that checks if the trigger action column actually changed

Double-Byte Character Data

The EXPLODE command might need to be used (along with a graphics display terminal) to browse DBCS (Double-Byte Character Set) data or graphics data.

Note: For more information, see the “Double-Byte Character Set Support” appendix.

Searched and Positional Methods

RC/Edit offers two editing methods: the Searched Update method and the Positional Update method. The Searched Update method uses a different algorithm to fetch and update the data than the Positional Update method. The following lists situations when the Searched Update method could be used for editing data.

- Two or more people want to edit the same table simultaneously. The Searched Update method allows multiple users to edit the same tables. An explicit lock is never issued when using the Searched Update method.
- To make global changes to a column. To make global changes to all rows or to selected rows, the SET command in the Searched Update method is quicker than CHANGE ALL in the Positioned Update.

- When data does not contain NULL values. When there are no NULL values in the data, Searched editing uses fewer resources than Positional editing, regardless of the types of changes being made.
- To refrain from changing a row that has been changed by another process. If the row was changed by another user since the row was fetched, the update will not be allowed.

  Note: When using the Searched Update method, data can be changed throughout the whole set of retrieved data. Be sure to know what data will be affected before using the Searched Update method.

The following are situations when the Positional Update method could be used for editing data.

- The data contains NULL values. Positional editing uses fewer resources than Searched editing when making changes one row at a time to data that contains NULL values.
- To update, delete, or insert rows in a table containing a LONG VARCHAR type column.
- To lock the table for some reason. When using the Searched Update method, the table cannot be locked for the edit.

Identical Rows in Retrieved Data

When data is changed in a row with the Searched Update method of editing, the data in all identical rows will be changed. Many tables do not allow identical rows, but this possibility does exist for some users. The Positional Update method only affects the actual row edited.

Editing, Locking, and Table Contention

The Positional method permits table locking; the Searched method does not. If you want to edit a table using the Searched Update method and another user is editing the data using the Positional Update method, you will be locked out of the table. A message will appear indicating the table is in use by user X. That user will receive a message indicating that your user ID is waiting to edit the table (unless both are in a data sharing environment, and the other user is in another MVS).

Two or more people using the Searched method of editing on the same data can encounter table contention if they both try to save their changes at the same time. Common SQL errors are +100 ROW NOT FOUND, or -911 ROLLBACK DUE TO DEADLOCK OR TIMEOUT.

+100 indicates that another user has changed the data after you fetched. The data in the editor will have to be refreshed and the changes made again.

-911 while using the SAVE or SET command indicates someone else is committing data at the same time. Wait a moment and try again. The deadlock should be over.

More information:

Error Processing (see page 147)

NULL Value Considerations

If the table being edited contains a number of columns that have NULL values, the Searched Update method of editing should only be used for global changes using the SET command or to permit simultaneous editors. The NULL values cause regular row updates to be processed slower than the Positional Update method.

Positional Method Update Order

When RC/Edit applies updates using the Positional Update method, they are applied in fetch order, not in the current display order. The order of updates is inconsequential except for the case of unique indexes on columns. For example, a user can delete a row that has a primary key value of 5, then update another row, assigning the same primary key of 5. If the delete does not occur before the update, a duplicate error occurs.

Consequently, some updates involving unique indexes might need to be made by using multiple edit sessions or issuing the SAVE command to commit the index changes during the edit session.

Searched Method Update Order

The Searched Update method updates the rows in the order they are displayed on the screen. The Searched method uses only half as many fetches as the Positional method.

Performing Sort During Edit and Browse Processing

When using CA RC/Update to edit a table, you can specify an option (D) for the Searched Update method. This option invokes a processing sequence that can help improve performance. First, this method initiates the sorting process in DB2. Then, if you specified a row limit, this method applies a data retrieval limit to the sorted data.

This processing sequence is helpful when you do not specify extended data queries and you apply the row limit. Having the sort processing occur before the limit processing helps ensure that the correct data is retrieved.

Note: This processing sequence is the default behavior when browsing tables.

Example: Generate a DB2 Sort and Fetch the First 20 Table Rows

This example demonstrates invoking an editing method that generates a DB2 sort (ORDER BY) first and then retrieves a limited number of rows (LIMIT). Using this method can help improve performance through more efficient background processing.

1. Choose a table to edit (from the CA RC/Update Main Menu), and then complete fields as follows on the RC/Edit Options panel:

   a. Type **Y** in the Where field (to create a data query).

   b. Type **20** in the Row Limit field.

   c. Type **D** (for Searched Update) in the Update Method field.

   Press Enter.

   The Data Query Edit panel appears.

2. Type **1D** next to a column (to retrieve the 20 rows in descending order), and press the END key.

   Retrieval results appear, showing 20 rows arranged as instructed.

Fetching in Searched Update Method

RC/Edit has two methods for fetching table rows: asynchronous and synchronous. The asynchronous method permits users to start editing large tables faster. The synchronous method fetches all rows of a table before editing begins.

Asynchronous background fetching can be enabled or disabled on the RC/Edit Parameters screen or in the RCEDIT parmlib member. By default, asynchronous fetching is enabled.

Asynchronous Background Fetching

When a large table is selected for editing, asynchronous fetching fetches and displays an initial set of rows, and editing can begin immediately. The rest of the table is fetched in a background task-continuously or as needed-while editing is taking place. The asynchronous background fetching method operates in one of two modes: continuous mode or pause mode.

Continuous Mode

When asynchronous fetching is operating in continuous mode, RC/Edit fetches the first 500 rows of the table, and then displays the first of those rows. Notice in the following sample screen that the Row number field indicates 1 of 500. The 1 indicates the number of the row at the top of the row display, and the 500 indicates the current number of rows that can be displayed. The status line reads FETCH STATUS: IN PROGRESS, indicating that RC/Edit is in the process of fetching more rows.

```
RUEDITC        ------- RC/Edit: Searched Column Mode   ------
COMMAND ===>                                                 SCROLL ===> PAGE

For Table ==> TSSMM.TABLEABC                  Row number==> 1 OF 500
Edit Mode ==> C                               Max Char  ==> 070
SSID: D81B ----------FETCH STATUS: IN PROGRESS---------------------- TDAMY
OPT S CORE_TRANS_ID    SOURCE_SYSTEM_CODE PART_IND TYPE_CODE
___   ABYAABYAAAAAAAAA AAA                AA       A
___   AAAAAAAAAAAAAAAB AAB                AB       B
___   AAAAAYYZDAAAAAAC AAC                AC       C
___   AAAAAAYYZAAAAYYZ YYZ                AD       D
___   AAAAAAYYZAAAAAAE AAE                AE       E
___   AAAAAAYYZAAAAAAF YYY                AF       F
___   AAAAAAYYZAAAAAAG AAG                AG       G
```

The next sample screen shows what happens when you press Enter, a scrolling key, or any interrupt key. In this example, the F8 (Down) key caused the first number in the Row number field to be 20. In the meantime, RC/Edit has fetched more rows in the background, so the second number of the Row number field shows 3611, the total number of rows currently fetched and available to be displayed and edited.

The following message indicates that RC/Edit is still fetching lines (above 3611):

```
FETCH STATUS: IN PROGRESS
```

```
RUEDITC        ------- RC/Edit: Searched Column Mode   ------
COMMAND ===>                                                SCROLL ===> PAGE

For Table ==> TSSMM.TABLEABC                  Row number==> 20 OF 3611
Edit Mode ==> C                               Max Char  ==> 070
SSID: D81B ----------FETCH STATUS: IN PROGRESS---------------------- TDAMY
OPT S CORE_TRANS_ID    SOURCE_SYSTEM_CODE PART_IND TYPE_CODE
___   AAAAAAAAAAAAAAAR AAR                AR       R
___   AAAAAAAAAAAAAAAS AAS                AS       S
___   AAAAAAAAAAAAAAAT AAT                AT       T
___   AAAAAAAAAAAAAAAU AAU                AU       U
___   AAAAAAAAAAAAAAAV AAV                AV       V
___   AAAAAAAAAAAAAAAW AAW                AW       W
___   AAAAAAAAAAAAAAAY AAY                AY       Y
___   AAAAAAAAAAAAAAAY AAY                AY       Y
___   AAAAAAAAAAAAAAAZ AAZ                AZ       Z
```

The Row number field displays the current row number and total number of displayable rows in the format *n* OF *nn*, where *n* is the current row and *nn* is the current number of displayable rows. The current row is at the top of the screen. The number of displayable rows equals the current total of rows fetched, plus the number of rows inserted, minus the number of rows deleted.

The number of displayable rows is updated when the screen is updated by pressing Enter, a scrolling key, or any interrupt key. The message FETCH STATUS: COMPLETE appearing in the status line of the screen indicates that all rows have been fetched.

Actions Requiring All Rows

If an action is performed that requires all rows before the entire table is fetched, a window appears.

When the window appears, the background fetch is still in progress. Return to the edit session at any time by pressing the Attention key. If the Attention key is pressed, the window disappears, and the requested action continues with the rows that have been fetched. However, the action is not performed on any rows that were not fetched before pressing the Attention key.

Note: If a system error occurs and RC/Edit is unable to continue fetching rows, it attempts to recover and continue the edit session with the rows that were already fetched.

Actions that require all rows include entering a FIND or CHANGE command that includes the ALL keyword; entering the MAX command using the PF8 (Down) key; and entering a SORT command.

Pause Mode

When asynchronous mode is operating in pause mode, RC/Edit fetches rows in increments of 500 and pauses between each fetch. RC/Edit fetches the first 500 rows of the table, and then displays the first of those rows. The following sample screen shows that the Row number field indicates 1 of 500. The status line reads FETCH STATUS: PAUSED, indicating that there are additional rows to be read, but the fetch has been paused.

```
RUEDITC        ------- RC/Edit: Searched Column Mode   ------
COMMAND ===>                                                 SCROLL ===> PAGE

For Table ==> TSSMM.TABLEABC                    Row number==> 1 OF 500
Edit Mode ==> C                                 Max Char  ==> 070
SSID: D81B ----------FETCH STATUS: PAUSED--------------------------- TDAMY
OPT S CORE_TRANS_ID    SOURCE_SYSTEM_CODE PART_IND TYPE_CODE
___   ABYAABYAAAAAAAAA AAA                AA       A
___   AAAAAAAAAAAAAAAB AAB                AB       B
___   AAAAAYYZDAAAAAAC AAC                AC       C
___   AAAAAAYYZAAAAYYZ YYZ                AD       D
___   AAAAAAYYZAAAAAAE AAE                AE       E
___   AAAAAAYYZAAAAAAF YYY                AF       F
___
```

The next set of 500 rows will be fetched when an attempt is made to scroll past the last row that has already been read. In the following example, the second number in the Row number field shows 1000, the total number of rows currently fetched and available to be displayed and edited:

```
RUEDITC        ------- RC/Edit: Searched Column Mode   ------
COMMAND ===>                                                 SCROLL ===> PAGE

For Table ==> TSSMM.TABLEABC                    Row number==> 579 OF 1000
Edit Mode ==> C                                 Max Char  ==> 070
SSID: D81B ----------FETCH STATUS: PAUSED--------------------------- TDAMY
OPT S CORE_TRANS_ID    SOURCE_SYSTEM_CODE PART_IND TYPE_CODE
___   AAAAAAAAAAAAAAQC AQC                QC       C
___   AAAAAAAAAAAAAAQD AQD                QD       D
___   AAAAAAAAAAAAAAQE AQE                QE       E
___   AAAAAAAAAAAAAAQF AQF                QF       F
___   AAAAAAAAAAAAAAQG AQG                QG       G
___   AAAAAAAAAAAAAAQH AQH                QH       H
___
```

For a description of the actions that require all rows, see the section, Continuous Mode.

Synchronous Fetching

Using synchronous fetching, RC/Edit fetches all the rows of a table before displaying any rows. Although RC/Edit will attempt to use the asynchronous method whenever possible, there are some cases when the synchronous method will be used. The synchronous method is used in the following circumstances.

- Asynchronous fetching is disabled on the RC/Edit Parameters screen or globally in the RCEDIT parmlib member.
- An ORDER BY is specified on a data query.
- Editing an updatable joined view.
- A REFRESH command is issued and ordering information is specified, in an initial data query or by a sort command issued during the edit session.
- Editing in positional mode.

Extended View Updatability

The Extended View Updatability feature lets you edit most joined views on the EDIT screen using Searched Column Mode. This is accomplished by mapping the columns in the views to underlying base tables.

Set the View Updatability Options field to E (Extended Support) on the RC/Edit Parameters screen to activate the joined view edit capability. Extended support is active only when a joined view is edited using the Searched Update method. It is not available for the Positional Update method.

After setting the View Updatability Options field to E, there is no need to issue a command or select an option to activate the feature. It is activated automatically for an edit session when a view is specified that cannot be updated by DB2 because the view contains a join between two or more tables.

Extended View Updatability Support Screen

After selecting the view to be edited and entering choices for the edit session on the Options screen, the Extended View Updatability Support screen appears.

This screen indicates that a joined view has been selected for editing, and it identifies the underlying tables. At this point, the edit session can be canceled with F3 (End) or can be continued by pressing Enter to continue to display the RC/Edit screen.

Searched Column Mode Screen

The edit screen also indicates that View Updatability Support is active by displaying an informational message under the command line.

```
RUEDITC        -------- RC/Edit: Searched Column Mode   -------
COMMAND ===>                                            SCROLL ===> PAGE
RU583I: RC/Edit extended view update support is active
For Table ==> PDDAM.EMPVIEW                    Row number==> 1 OF 7
Edit Mode ==> C                                Max Char  ==> 070
SSID: D81B ------------------------------------------------------------ USER3
OPT S EMP_NO EMP_NAME        EMP_TITLE       UNIT_NO UNIT_DESCRIPTION
___   0001   JACK TUCKER     DEVELOPER       01      RC/Update
___   0002   JIM BROADHURST  DEVELOPER       01      RC/Update
___   0004   DAVE MARSHALL   QUALITY ANALYST 01      RC/Update
___   0003   RICH LEBLANC    DEVELOPER       01      RC/Update
___   0006   ADAM RIRIE      DEVELOPER       02      RC/Migrator
___   0005   WAYNE DRISCOLL  DEVELOPER       02      RC/Migrator
___   0007   DICK TURGEON    DEVELOPER       03      INFO/SESSION
******************************* BOTTOM OF DATA ********************************
```

All columns are editable. To edit joined views, use the EDIT screen in the same manner as for a table or an unjoined view. In the previous joined view sample, the two underlying tables contain the following columns.

- The EMPLOYEE table includes the columns EMP_NO, EMP_NAME, EMP_TITLE, and UNIT_NO.
- THE UNIT table includes the columns UNIT_NO and UNIT_DESCRIPTION.

In the joined view EMPVIEW, the two tables are joined on column UNIT_NO, so UNIT_NO appears only once in the displayed view.

Affecting Other Rows

Changing one row can affect others in the view. Changing the first line of UNIT_DESCRIPTION from RC/Update to RC/SECURE and pressing Enter causes all the UNIT_DESCRIPTION data to be changed in all rows where UNIT_NO is 01. The following screen shows that this change affects rows for the first four employees in the view. The U in the S column indicates that at least one column for that row has been affected:

```
RUEDITC        -------- RC/Edit: Searched Column Mode   -------
 COMMAND ===>                                              SCROLL ===> PAGE

 For Table ==> PDDAM.EMPVIEW                    Row number==> 1 OF 7
 Edit Mode ==> C                                Max Char  ==> 070
 SSID: D81B ------------------------------------------------------------ USER3
 OPT S EMP_NO EMP_NAME        EMP_TITLE       UNIT_NO UNIT_DESCRIPTION
 ___ U 0001   JACK TUCKER     DEVELOPER       01      RC/SECURE
 ___ U 0002   JIM BROADHURST  DEVELOPER       01      RC/SECURE
 ___ U 0003   RICH LEBLANC    DEVELOPER       01      RC/SECURE
 ___ U 0004   DAVE MARSHALL   QUALITY ANALYST 01      RC/SECURE
 ___   0006   ADAM RIRIE      DEVELOPER       02      RC/Migrator
 ___   0005   WAYNE DRISCOLL  DEVELOPER       02      RC/Migrator
 ___   0007   DICK TURGEON    DEVELOPER       03      INFO/SESSION
 ****************************** BOTTOM OF DATA ********************************
```

Editing Multiple Tables

Use the extended view update support to edit more than one table at a time. In the following screen example, the UNIT_NO for EMP_NAME Dave Marshall was changed to 04, a value that did not previously exist. Placeholder characters or default values are used to indicate where data is needed. In the following example, the placeholder character (@) is inserted in the appropriate row of UNIT_DESCRIPTION:

```
RUPEDITC       -------- RC/Edit: Searched Column Mode   -------
COMMAND ===>                                              SCROLL ===> PAGE

For Table ==> PDDAM.EMPVIEW                    Row number==> 1 OF 7
Edit Mode ==> C                                Max Char  ==> 070
SSID: D81B ------------------------------------------------------------ USER3
OPT S EMP_NO EMP_NAME        EMP_TITLE       UNIT_NO UNIT_DESCRIPTION
___ U 0001   JACK TUCKER     DEVELOPER       01      RC/SECURE
___ U 0002   JIM BROADHURST  DEVELOPER       01      RC/SECURE
___ U 0003   RICH LEBLANC    DEVELOPER       01      RC/SECURE
___ U 0004   DAVE MARSHALL   QUALITY ANALYST 04      @
___   0006   ADAM RIRIE      DEVELOPER       02      RC/Migrator
___   0005   WAYNE DRISCOLL  DEVELOPER       02      RC/Migrator
___   0007   DICK TURGEON    DEVELOPER       03      INFO/SESSION
****************************** BOTTOM OF DATA ********************************
```

Typing over the placeholder character assigns a real value (CORPORATE) for the UNIT_DESCRIPTION of the new UNIT_NO value 04, as shown in the following example:

```
RUPEDITC       -------- RC/Edit: Searched Column Mode   -------
COMMAND ===>                                              SCROLL ===> PAGE

For Table ==> PDDAM.EMPVIEW                    Row number==> 1 OF 7
Edit Mode ==> C                                Max Char  ==> 070
SSID: D81B ------------------------------------------------------------ USER3
OPT S EMP_NO EMP_NAME        EMP_TITLE       UNIT_NO UNIT_DESCRIPTION
___ U 0001   JACK TUCKER     DEVELOPER       01      RC/SECURE
___ U 0002   JIM BROADHURST  DEVELOPER       01      RC/SECURE
___ U 0003   RICH LEBLANC    DEVELOPER       01      RC/SECURE
___ U 0004   DAVE MARSHALL   QUALITY ANALYST 04      CORPORATE
___   0006   ADAM RIRIE      DEVELOPER       02      RC/Migrator
___   0005   WAYNE DRISCOLL  DEVELOPER       02      RC/Migrator
___   0007   DICK TURGEON    DEVELOPER       03      INFO/SESSION
****************************** BOTTOM OF DATA ********************************
```

As a result, two tables are changed in the following ways.

- In the EMPLOYEE table, the column UNIT_NO has a new value, 04, in the row for EMP_NAME Dave Marshall.
- In the UNIT table there is a new row. In this new row the column UNIT_NO has the value 04, and the column UNIT_DESCRIPTION has the value CORPORATE.

More information:

Form Mode Edit (see page 144)

Restrictions on View Updatability

Although extended view update support feature allows users to edit most joined views, the following are cases where editing is not permitted.

- A joined view is selected for which the join condition is incomplete or too complex.
- The view has a join that includes a remote table.
- The view is inherently not updatable, for example, the SELECT clause contains the keywords GROUP BY or HAVING.

Note: The extended view update support feature does not currently support updating an Outer Join.

In cases where the extended view update support feature is not applicable, the feature is not activated. Instead, the Edit screen appears after the Options screen, to allow the user to browse the view, and message RU584I appears.

Note: The CREATE VIEW statement has been extended to support the optional CASCADED and LOCAL keywords as part of the CHECK OPTION clause. RC/Edit does not allow table editing through views if any of the views are defined with the CASCADED or LOCAL check options.

Editing Algorithms

The editing algorithms of the Searched Update method and the Positional Update method are shown in examples which use the following sample table.

```
BROWSE -- USER5.DB2.CDBASAMP(EDITMETH) - 01.00 ------ LINE 00000000 COL 001 080
COMMAND ===>                                                  SCROLL ===> CSR
******************************** TOP OF DATA *********************************
 SAMPLE TABLE  DSN8810.EMP
 OPT S EMPNO  FIRSTNME     MIDINIT LASTNAME        N:WORKDEPT N:PHONENO
 ___   000010 CHRISTINE    I       HAAS            N A00      N 3978
 ___   000020 MICHAEL      L       THOMPSON        N B01      N 3476
 ___   000030 SALLY        A       KWAN            N C01      N 4738
 ___   000050 JOHN         B       GEYER           N E01      N 6789
 ___   000060 IRVING       F       STERN           N D11      N 6423
 ___   000070 EVA          D       PULASKI         N D21      N 7831
 ___   000090 EILEEN       W       HENDERSON       N E11      N 5498

****************************** BOTTOM OF DATA ********************************
```

The goal for both methods illustrated in the following is to change the Work Department (WORKDEPT) from E01 to B01 for Employee Number (EMPNO) 000050.

Searched Update Method

When editing a table using the Searched Update method, it cannot be locked. The table name is enqueued in its own internal queue. No DB2 locks are placed on the table, but RC/Edit will not permit another user to edit the table using the Positional Update method.

Multiple users can edit the same table as long as they all use the Searched Update method. If another user attempts to edit the table using the Positional Update Method, a message appears to both users informing them of the contention. (In a data sharing environment where the other user is on a different z/OS system, you will see the message, but the other user will not.)

Positional Update Method

When editing a table using the Positional Update method, you have the following lock options:

- Locking the Table—If the table is locked, an explicit DB2 lock is placed on that table to help ensure that no other users will be able to edit the table using another product, such as SPUFI, while you are in an edit session. The lock is freed when the edit session is exited.
- Not Locking the Table—If the table is not locked, update activities outside of the product can occur during an edit session, with unpredictable results. RC/Edit will still enqueue the table name in its own internal queue. No DB2 locks will be placed on the table, but RC/Edit will not permit another user to edit the table using the Positional Update method or the Searched Update method.

Note: If another user attempts to edit the table, a message indicating that the other User ID is waiting to edit the table appears. Likewise, the other user receives a message that your User ID is editing the table. (In a data-sharing environment where the other user is on a different z/OS system, you will see the message, but the other user will not.)

The table should be locked whenever there is a concern about preserving data integrity. You can omit locking the table if you are editing a test table and are certain that there will not be any update activity outside CA RC/Update.

Positional Update Warning Screen

The following informational screen displays when Positional Update is selected without specifying locked tables.

Note: This screen never appears if you set the Default Update Method (UMETHOD) in the DEFAULTS parmlib member to P to force locking with the Positional Update method. An error message is generated if this method is requested, but locking cannot be used. This occurs when the table is a remote table.

To suppress display of this panel in the future when editing tables in Positional Update mode when locked tables are not specified, enter **Y** (yes) in the Suppress display of this panel in the future field at the bottom of the screen. This screen will not display again. Use this option with caution. From this point on when editing a table in Positional Update mode without specifying locked tables, this panel will not display and no warning will display. If N (no) is specified, the next time a table is edited in Positional Update mode without specifying locked tables, this warning panel will be displayed.

RC/Edit Options

The RC/Edit Option screen's main function is to control the amount of data that is retrieved for an edit session. A row limit and an SQL statement can be specified to control the number of rows returned. A test count can also be requested before retrieving the rows to determine the impact of a selection.

This screen can also be used to control the initial display format (Column or Form) and the maximum display character size. All these attributes can be saved as the default for the selected table to save time in future sessions.

The Edit Options screen is bypassed if using the FE (fast edit) option. If FE is entered from the Edit Options screen, all options on the screen are ignored. The default options last saved for the table are used. If there are no default options, then the RC/Edit default options are used.

RC/Edit Options Screen Layout

The Edit Option's header is unprotected and can be changed to edit another table or perform another option. There is no need to return to the Main Menu to perform a different function. Selection criteria can also be input for the Item Name and Creator prompts to retrieve the Table Selection Screen.

Note: The Data Selection fields are used to restrict the data retrieved from the selected table. For information about the fields, press F1 (Help).

Processing for RC/Edit Options

You can specify the following processing options:

- To begin the edit session, press Enter.
- To return to the previous screen, press F3 (End). If the previous screen was the RC/Edit Table Selection screen, the user will be returned to that screen with the previous selection criteria still active. This permits easy selection of another table for editing. To return directly to the Main Menu, enter the MAIN command. If the previous screen was not the Table Selection screen, the Main Menu appears.

- To change to a different table, change the Table Name or Creator prompt in the header. A new Edit Options screen appears for the new table. This permits viewing of the default edit options for that table and the ability to make any changes before beginning the edit session.

  Edit options can be entered on the current Edit Options screen at the same time the new table name is entered. Because changes have been made to the edit options, RC/Edit assumes that the options currently displayed on the panel should be used and immediately takes advances to edit mode. The new Edit Options panel will not appear.

- Specify the FE option to exit a Fast Edit session. If the FE option is used, the Edit Options screen (versus the Main Menu) appears when the edit session is exited. This is intentional in case fast-mode was entered and the user really did want to enter options. Because the header on the option screens is active, the option screen can serve the same purpose as the Main Menu.

Column Mode Edit

Column Mode is selected by entering **C** for the Edit Mode field in the RC/Edit header or Edit Options screen. The Column Mode screen appears.

Column mode is used to edit multiple rows on the screen at one time. A row always takes one line and columns are scrolled horizontally with the F10 (Left) and F11 (Right). Rows are scrolled with the F7 (Up) and F8 (Down) keys.

You can control the display length of character data by taking either of the following actions:

- Adjust the value in the Max Char field in the header area. For example, enter a small value to view more columns on one screen.
- Issue the SETWIDTH command to request the preferred length.

Column Mode Header

The screen header provides information and acts as a control center for the edit session. The control information can be changed at any time.

Note: For information about the fields, press F1 (Help).

More columns can be viewed on the screen by shrinking the character width down to a smaller number. Use the EXPLODE option (see EXPLODE Facility) to view the whole column in ISPF Edit.

Processing for Column Mode Edit

The Column Mode Edit screen is designed to maximize row operations. Edit operations can be performed on as many rows as needed. Full ISPF scrolling is available and extensive type checking is performed to help ensure valid data.

Enter a line editing command in the OPT (option) column. The available ISPF-like line editing commands follow:

A

Specifies a point after which to move or copy lines.

B

Specifies a point before which to move or copy lines.

C, C*nn*, or CC

Copies a line, a number of lines, or block of lines, respectively.

D, D*nn*, DD

Deletes a line, a number of lines, or block of lines, respectively.

F or F*nn*

Shows the first line of a block of excluded lines, or first *nn* number of lines from an excluded block.

I, I*nn*

Inserts a line or a number of lines.

K

Inserts an information line in the table display, indicating the primary (P*n*) and foreign key (FK) columns of the table.

L or L*nn*

Shows the last line of a block of excluded lines, or last *nn* number of lines from an excluded block.

M, M*nn*, or MM

Moves a line, a number of lines, or block of lines, respectively.

R, R*nn*, RR

Repeats a line, a number of lines, or a block of lines, respectively.

S, Snn, or SS

Shows an excluded line, the first *nn* number of excluded lines, or block of excluded lines, respectively.

U, Unn, or UU

Undeletes a line, a number of lines, or a block of lines, respectively.

X, Xnn, or XX

Excludes a line, a number of lines, or a block of lines, respectively.

As in ISPF, single line commands operate on individual rows. For example, entering D deletes that row.

Copy, Delete, Undelete, Exclude, Repeat, and Move support block mode. Enter CC, DD, UU, XX, RR, or MM at the start and stop rows of the block perform the action on all rows within that block, inclusive of the rows containing the block commands.

Each command can be followed by a number to indicate multiple occurrences. For example, I20 inserts 20 new rows. X13 excludes 13 rows. For the MOVE command, a destination must be specified.

If line commands have been entered in error, enter the RESET command to remove the invalid entries.

Note: Although RC/Edit allows users to move rows, DB2 does not move the rows, because by definition a relational database is non-positional. The MOVE command allows row movement for readability while in RC/Edit only.

EXCLUDE Command Notes

If a FIND command is issued that searches for text in an excluded line, the text will be found and the line unexcluded. If a CHANGE command is issued that affects text in an excluded line, the text will be changed, and the lines unexcluded.

There are also line commands that affect excluded lines.

F*n*

Shows the first *n* excluded lines, where n is a number. For example, F6 shows the first six excluded lines.

L*n*

Shows the last *n* excluded lines, where n is a number. For example, L14 shows the last 14 excluded lines.

SS

Shows excluded blocks of text. Enter **SS** on the first and last lines of the block of text to show. This command is useful for showing more than one excluded block of text.

The following describes the columns that follow the OPT field.

(S)

Indicates the current status of the row. A blank status denotes a row in its initial fetched format.

D

Indicates that the row has been deleted. The SDELETE option must be turned on to view deleted rows.

I

Indicates that the row has been inserted. If the R line command is used, the inserted rows will have a status of I.

If a user deletes a row that was inserted, then undeletes the row, the status returns to inserted.

U

Indicates that the row has been updated.

X

Indicates that the row has been excluded. If a block of rows was excluded, one status line for the block appears. A message displaying the number of excluded lines is also displayed.

Nulls

If a field accepts null values, then an extra prompt (N) will appear next to the name. This prompt is the Null Indicator.

It denotes if the field currently is set to Null (Y). Set a field to Null by entering Y in the Null Indicator.

If a field is set to null, the previous value remains in the event the user wants to change it. However, the value for that field will be NULL if the indicator is Y. For a description of Null value implications, see Null Values.

Field Names

The DB2 column name.

Abbreviated Field Names

If the TYPE command is on, an abbreviated name appears under the DB2 name. (If a second line does not appear under the field Names, the TYPE toggle does not need to be active.)

The abbreviated names begin with A1. The abbreviated names make it much easier to enter column names for other primary commands such as XCOL and CHANGE.

DB2 Data Type

The DB2 data type, displayed along with the abbreviated name. This line is toggled on and off with the TYPE command.

There are many primary edit commands that can be entered on the command line or assigned to function keys for controlling the column display. Some of these commands include CHANGE, RCHANGE, CHANGE ALL, SORT, and TYPE.

More Information:

RC/Edit Command Reference (see page 148)

Form Mode Edit

Form Mode is selected by placing an **F** for the Edit Mode prompt in the RC/Edit header or Edit Options screen. Form mode is used to view one row per screen. The columns are displayed vertically down the left side, much like a form. This approach displays more columns on the screen than Column Mode.

Character data's display length is controlled by the Max Char prompt in the header. Form Mode supports a maximum display length of 256 characters. If the entry exceeds 50 characters, an arrow indicates that additional text is present but not shown. You can expand the area by using the EXPAND command.

Form Mode Header

Form Mode's header provides the same functions and information as Column Mode, but with two additional fields: a Status field for viewing the row's current status and an Option field for entering edit actions.

Note: For information about the fields, press F1 (Help).

Processing for Form Mode Edit

The Form Mode Edit screen is designed to maximize the columns displayed on the screen. Full ISPF scrolling is available during the edit session. Also, extensive type checking is performed to help ensure that valid data is entered.

(ABBREVIATED FIELD NAMES)

In the front of each field appears an abbreviated field name. The abbreviated names begin with A1. The abbreviated names make it much easier to enter column names for other primary commands such as FIND, RFIND, and FREEZE.

COLUMN NAME / COLUMN TYPE

The DB2 column name or data type. The TYPE command determines which information appears.

NULL

The Null Indicator appears after each column name. If a column is defined as NULL (nulls allowed), users can cursor to this field. Otherwise, the field is protected.

The Null Indicator denotes the column's current state as well as permits changes to the null attribute. If the field is currently null, a Y appears in this field. To set a column to null, enter **Y** in this field.

If a field is set to null, the previous value remains in the event the user wants to change it. However, the value for that field will be null if the indicator is Y. For a description of null value implications, see Null Values.

DATA FOR ROW #*n*

Data values for the row number indicated. Control the display width of character data with the Max Char prompt in the header.

There are many primary edit commands that can be entered on the command line or assigned to PF keys for controlling the display. Some of these commands include CHANGE, RCHANGE, CHANGE ALL, SORT, and TYPE.

Expand and Edit Data for Long Character Columns

You can edit long character column data by using the EXPLODE facility within RC/Edit. This facility invokes the standard ISPF editor and displays full screen mode. In the ISPF edit session, you can perform any ISPF command, and you can copy data from other data sets.

You can use the editor to enter formatted character data with indentation and blank lines. If using QMF or [set the PRF value for your book] to view the column's data, you can enter an edit code to wrap the data at 50 characters. This allows column character data to be treated as a formatted data set.

Contiguous text can be entered with no indentation or padding with spaces. If a data set exists into which you want to copy the character's column, refill the text in the original data set to fit within 50 columns.

Note: By default, the ISPF edit session displays the starting position of the data on the associated line within the expanded column. During the edit process, if you move, insert, or delete lines, the line numbers will not update dynamically to reflect new positions. To refresh the column position, you must exit the EXPLODE facility (by pressing PF3) and then re-enter.

Follow these steps:

1. Complete fields as follows on the CA RC/Update Main Menu:
   - Type **E** in the Option field.
   - Type **T** in the Object field.
   - Specify a table object name in the Item field.

     Note: You can also enter selection criteria to display a selection list.
   - (Optional) Complete the remaining fields as needed to refine your request.

   Press Enter.

   The RC/Edit Options panel appears.
2. Complete the fields on the panel as needed to determine options for your edit session, then press Enter.

   A column or form mode panel appears, listing rows of column data.
3. Type **EXPLODE** in the command line, move the cursor to the column whose data you want to expand (explode), and press Enter.

   Note: The EXPLODE command (along with a graphics display terminal) might be needed to edit DBCS (Double-Byte Character Set) data or graphics data.

   An ISPF edit session begins, showing the detailed column data.
4. Perform any edits (issuing ISPF commands as needed), then press the F3 (End) key.

   Note: You can edit hexadecimal data by using the HEX and HEX OFF primary commands, which work as a toggle. Additionally, if you need to refresh the sequential line numbering, you can issue the NUM OFF primary command.

   The column or form mode panel reappears after you press the End key.

More information:

RC/Edit Command Reference (see page 148)

Trace Facility

You can view the RC/Edit row processing during a SAVE, SET, or END command using the trace facility. Turn the trace facility on with the TRACE ON command or by entering **Y** at the Trace DB2 Save I/O prompt on the RC/Edit Options screen.

During the trace, RC/Edit displays all DB2 calls issued such as FETCH, INSERT, UPDATE, and DELETE. Also, real-time statistics are displayed showing the total number of fetches, updates, deletes, and inserts performed. At the end of the trace, a summary of the statistics appears. Press Enter to continue.

The trace facility can be used to monitor progress during a large update session or to view internal processing of RC/Edit.

Important! The trace facility updates the screen each time a row statistic (such as the number of fetches) is updated. The high number of screen updates uses significant resources in the teleprocessing environment. Use this facility with caution.

Error Processing

When changes are saved (END, SET or SAVE commands), those changes are then committed to DB2. However, there is the possibility of an error, even though RC/Edit performs security and type checking during entry.

Some possible errors include:

- Duplicate key for a unique index on an insert or update
- A security violation, occurring when someone removes user authority while the user is editing the data
- Table contention in the Searched Update Method, occurring when issuing the SET or SAVE command at the same time as another user is committing data. Wait a minute and then try again, and the deadlock should be over.

When an error occurs, the Error screen appears. The row in error appears along with a corresponding error message. The user has four options: Ignore the row, Retry the request, Cancel the edit session, or View the full message text for the error. Update the row by moving the cursor to the appropriate fields. Only the primary edit commands TYPE or EXPLODE can be entered.

Note: For information about the fields, press F1 (Help).

RC/Edit Command Reference

RC/Edit supports a full range of primary commands for controlling the display and performing edit related functions. These commands can be assigned to Function keys or entered on the command line.

Matching Data

Some of the primary commands search for data within columns to perform their functions. Some examples include FIND, RFIND (F5), CHANGE, RCHANGE (F6), and SORT. The following sections describe how RC/Edit performs searches as a result of issuing these commands:

General

When RC/Edit searches for a string as a result of a FIND or CHANGE command, the search starts at the first column of the first visible row on the screen unless a column name is specified. If a FIND or CHANGE command is issued with a specified column name, the search starts at the top of the table. If the search is successful, the cursor moves to the column containing the search string. If the next search command is RFIND or RCHANGE (without a column name), the search starts after the current cursor position. If the next search command is RFIND or RCHANGE with a specified column name, the search starts at the top of the table.

RC/Edit searches all columns automatically unless a column name or column abbreviation is specified to limit the search. The column being searched does not have to appear on the screen. When there is a successful match that is not already on the screen, RC/Edit automatically displays that column on the left side of the screen (for Column Mode) or the top of the screen (for Form Mode).

Keywords for Searches

Following are keywords used to search data.

ALL

After a FIND command, ALL searches for all occurrences of the search string and displays the count of occurrences instead of the actual occurrences.

After a CHANGE command, ALL searches for all occurrences of the search string, changes each, and displays a count of the changes. If the change string is longer than the search string in a particular occurrence, there might not be room to make the substitution. In this case, a count of changes that were not allowed is generated, along with the count of actual changes.

X

After a CHANGE command, the X keyword causes the search to look only for occurrences in rows or columns that have been excluded, with XCOL or the X line command.

NX

After a CHANGE command, the NX keyword causes the search to look only for occurrences in rows or columns that have not been excluded with XCOL or the X line command.

Y or NULL

After a CHANGE command, the keywords Y or NULL cause the search to look only for NULL values.

Examples:

```
FIND Y
FIND NULL
```

N or ¬NULL

After a CHANGE command, the keywords N or ¬NULL cause the search to look only for non-NULL values. If either of these keywords represent the replacement value in a CHANGE command, the value found is changed to 0 in a numeric column, or to a blank in a character column.

Examples:

```
CHANGE NULL  ¬NULL
CHANGE  Y  N
```

Note: To search for characters that are keywords, they must be enclosed in single quotes (for example, FIND 'Y').

Asterisk as Search String Parameter

Use an asterisk (*) as a search string parameter instead of typing in the previous one again. When a search string is entered for a CHANGE command, that value is saved as the current search string. If FIND AJAX is entered, AJAX is saved as the current search string. CHANGE * ACME can then be entered, and searches for the next occurrence of AJAX and change it to ACME. The asterisk (*) can be also found as the search string by enclosing it in single quotes (for example, find '*').

Character Data

When searching for character data, the string needs to be included in quotes if it contains blanks or if the string is embedded in apostrophes.

Note: To search for a null value in a nullable column, use ' ' as the find string.

If column length has been controlled with the SETWIDTH command, RC/Edit searches the whole column for character data, regardless of the display length. If only 10 characters are displayed in a 1000 character column, RC/Edit still searches all 1000 characters.

When character data is being changed (CHANGE family of commands), the data surrounding the change is expanded or contracted accordingly. If the expansion of data would cause loss of significant data, then the change is not processed.

Numeric Data

For numeric data, RC/Edit uses the whole number as the search value. It does NOT break a number into parts like character data. For example, if a search value of 1000 is entered, RC/Edit will not match 10000.

Notational Conventions for RC/Edit Commands

The following notational conventions are used for the command syntax.

Notation	Description
UPPERCASE characters	Must be entered as shown.
lowercase characters	User-specified variables.
<*column name*>	The full DB2 column name or the RC/Edit column abbreviation. **Note:** A name that contains embedded blanks must be enclosed in quotation marks.
( )	Must be entered where shown.
[ ]	Enclose optional parameters, choose 1.

Notation	Description
{ }	Enclose required parameters.
|	Or, as in Y | N.
Underlined	Abbreviations.

RC/Edit Commands

The following commands are associated with RC/Edit.

Note: For a thorough listing and command function description, see the "Command Summary" appendix.

- CANCEL (see page 152)
- CAPS (see page 152)
- CHANGE or RCHANGE (see page 152)
- COMMA (see page 153)
- END (see page 153)
- EXPLODE (see page 154)
- HEX (see page 154)
- LCASE and UCASE (see page 155)
- PROTECT and UNPROTECT (see page 155)
- REFRESH (see page 155)
- RESET (see page 156)
- SAVE (see page 156)
- SCOL (see page 156)
- SDELETE (see page 156)
- SET (see page 157)
- SORT (see page 157)
- SORT? (see page 157)
- SQL (see page 158)
- STATS (see page 158)
- TRACE (see page 158)
- TYPE (see page 158)
- XCOL (see page 158)

CANCEL

Cancels the current edit session and returns to the Edit Options screen. No changes are made to the DB2 table.

The command syntax is:

```
CANCEL
```

The mode is Edit.

CAPS

Controls the data capitalization within a column. This attribute is assigned at the column level for more control. When CAPS is ON, all text entered for the column is converted to capital letters. This only affects new or updated data. To change all occurrences, see the UCASE or LCASE commands.

The command syntax is:

```
CAPS ON <column name>
```

```
CAPS OFF <column name>
```

The mode is Edit.

CHANGE or RCHANGE

Changes the value of a character string or number within a DB2 column.

The command syntax is:

```
CHANGE old-value new-value [column name]
```

```
RCHANGE CHANGE old-value new-value [column name]
```

The mode is Edit.

CHANGE

Changes the occurrence to another value.

RCHANGE

Changes the next occurrence from a value to another value. You must issue a CHANGE command before RCHANGE.

ALL

Searches for all occurrences of the search string, changes each, and displays a count of the changes. If the change string is longer than the search string in a particular occurrence, there might not be room to make the substitution; counts of changes not allowed and of actual changes are generated.

X

Searches for occurrences in rows or columns that have been excluded with the XCOL or the X line command.

NX

Searches for occurrences in rows or columns that have *not* been excluded with XCOL or the X line command.

Y or NULL

Searches for NULL values.

N or ¬NULL

Searches for non-NULL values.

To search for characters that are keywords, enclose them in single quotes.

COMMA

Determines if numeric columns will be formatted with commas. If COMMA is OFF, numeric columns will not be formatted with commas. Turn commas off when editing numeric columns (account or social security numbers).

The command syntax is:

```
COMMA ON

COMMA OFF
```

The mode is Edit or Browse.

END

Ends the edit session and saves all changes. If TRACE is ON, the trace screen displays during processing. If errors occur, the RC/Edit error screen displays.

The command syntax is:

```
END
```

The mode is Edit or Browse.

EXPLODE

Lets users enter the EXPLODE Facility to edit the contents of a character column using an ISPF full screen editor. To explode a character column, place the cursor on the column and press the EXPLODE function key. This feature allows the user to edit extremely long character data (any size) and to import data into the column from other data sets.

The command syntax is:

```
EXPLODE
```

The mode is Edit or Browse.

HEX

Specifies whether data displays in hexadecimal format when displaying the RC/Edit Searched Column Mode, Positional Column Mode, or Explode Facility screens. HEX is a toggle command. If the HEX mode is off, enter HEX and press Enter to turn the hexadecimal display mode on, and vice versa.

The display varies according to column type. When HEX mode is on, all character, graphic, and numeric columns display in hexadecimal format, and all date, time, and domesticate columns display as blanks.

HEX is not supported on Form Mode screens or the RI Editor screen. There are three lines per record display when the HEX mode is on. The first line is the EBCDIC format of the data. The second and third lines are the hexadecimal equivalents of the first line. The two characters that make up the hex representation of each EBCDIC character are directly under that character in vertical format.

Edit hexadecimal data by overtyping the information. Valid characters are 0-9 or A-F. Invalid characters are ignored.

The command syntax is:

```
HEX
```

The mode is Edit or Browse.

LCASE and UCASE

Change the case of every character in a character column. All values in the column are converted to the specified case.

The command syntax is:

LCASE *column_name*

The mode is Edit.

PROTECT and UNPROTECT

Protects or unprotects a column.

The command syntax is:

PROTECT *column_name*

UNPROTECT {*column_name*}

The mode is Edit.

PROTECT

Protects a specified column.

UNPROTECT

Unprotects a column that had been protected by the PROTECT command.

Note: Columns that are protected when the edit session ended are not updated. If any data in that column was updated before the column was protected, the update is ignored.

Column specification for UNPROTECT is optional. If no column is specified, all columns that were protected using the PROTECT command are unprotected.

REFRESH

Refreshes the data from the DB2 table. Any pending uncommitted updates are lost.

The command syntax is:

REFRESH

The mode is Edit.

RESET Command

Clears the current column mapping so the process can be restarted. Use this command with automapping to experiment with different mapping techniques.

The command syntax is:

```
RESET
```

SAVE

Saves the changes.

The command syntax is:

```
SAVE
```

The mode is Edit.

SCOL

Show Column. Unexclude columns excluded by the XCOL command.

The command syntax is:

```
SCOL [column name|@ALL]
```

The column specification and @ALL are optional using SCOL. If no column name is specified, SCOL unexcludes all columns that have been excluded by XCOL.

The mode is Edit.

SDELETE

Toggles the display of deleted records. The default mode is OFF so that deleted records disappear as they are deleted. To view deleted records, enter the SDELETE command. Undelete deleted records using the U line edit command.

Note: The SDELETE command works in Column Display mode only.

The command syntax is:

```
SDELETE
```

The mode is Edit.

SET

Changes the value of the column named in the command.

Note: The SET command works only when you are using the Searched Update method.

The command syntax is:

```
SET <column_name> = new_value [WHERE ...]
```

Note: If you do not specify a where clause, the value is changed for every row in the retrieved data. The SET command always uses any data query you entered on the RC/Edit Options screen to limit the rows affected by the command.

The SET command automatically commits all update changes made to the data, makes the change requested in the command and commits those changes to the database. The effect of the command is shown only if the Auto Refresh option is ON, or the REFRESH command is issued.

The mode is Edit.

SORT

Sorts the rows in ascending (default) or descending order according to the values in the designated column.

The command syntax is:

```
SORT column_name {A | D}
```

SORT with no parameters re-sorts the rows to the initial fetch order.

The mode is Edit or Browse.

SORT?

Displays the Data Query Edit screen. Use the Data Query Edit screen to sort on more than one column during the edit session.

The command syntax is:

```
SORT?
```

The mode is Edit or Browse.

SQL

Displays the SQL statement used to retrieve the data for the current edit session.

The command syntax is:

```
SQL
```

The mode is Edit or Browse.

STATS

View editing statistics anytime during an edit session using the STATS command. The Statistics Summary screen appears when the STATS command is entered while in an edit session. Pending statistics (the number of updates, deletes, and inserts that will be done if the user saves the data) are included in this screen.

TRACE

Turns online tracing of all activity during SAVE or END processing on or off. If TRACE ON is set, a screen displays showing the progress of the activity as it occurs. You can use this to gauge progress through the rows.

The command syntax is:

```
TRACE [ON|OFF]
```

The mode is Edit.

TYPE

Turns the DB2 data type line on and off. The data type line displays the column's DB2 data type along with the column abbreviation.

The command syntax is:

```
TYPE
```

The mode is Edit or Browse.

XCOL

Excludes one column or all columns from the display area.

The command syntax is:

```
XCOL {column name | @ALL}
```

The mode is Edit.

Chapter 8: RI Editor

This section contains the following topics:

Overview of RI Editor

RI Editor makes it easier to edit tables that have referential integrity relationships. With RI Editor, you can perform any of the editing operations and commands available in RC/Edit, plus new features for working with multiple tables of a referential integrity (RI) set as follows:

- View complete RI structures, including all tables that are referentially tied together with either system-defined or user-defined RI, based on user-supplied selection criteria.
- Visualize all the relationships within an RI structure by viewing the structure online in an indented tree format.
- Select and work with all or a subset of tables in an RI set.
- Display multiple tables of an RI set simultaneously on the screen. The size of the display area can be adjusted for each table individually, and each table can be edited.
- Hide a table temporarily from the display without removing it from the RI set or subset you are working with.
- Use synchronized scrolling to automatically match a parent table's unique constraints with the foreign keys of its child tables.
- Use the Key Field (K) line command to identify the unique constraints and foreign unique constraints columns in the tables you are displaying.

Note: RI Editor supports long names. For more information about support of long names, see the "Using CA RC/Update" chapter.

Edit Table Selection Panel

To display the Edit Table Selection panel and access RI Editor, enter **RE** in the Option field of the Main Menu. The Edit Table Selection panel appears. Tables are listed in referential sets, when applicable, according to the selection criteria you entered for the Item Name, Creator, and Where fields. Dashed lines mark the beginning of a new referential set. Single table names surrounded by dashed lines indicate tables that are currently not in a referential integrity relationship with any other tables.

The following sample screen shows three referential sets (two referential sets of tables and a solo table).

```
               --- RC/Update Edit Table Selection ---------
COMMAND ===>                                                SCROLL ===> PAGE

Option    ===> RE                Object   ===> T         Mode  ===> O ONLINE
Item Name ===> *              >  Creator  ===> PTI8230 > Where ===> N
SSID: D81B ------------------------------------------------------- USER3

  Enter 'S' to select a Base Edit Table.

Sel S  XXX  TABLE NAME                          LVL  CREATOR  GROUP     RI
   ---------------------------------------------------------------------------
__   1. ACT                                    1   PTI8230
__   2.   PROJACT                              2   PTI8230            S
__   3.     EMPPROJACT                         3   PTI8230            S
__   4. DEPT                                   1   PTI8230
__   5.   DEPT (4+)                            2   PTI8230            S
__   6.   EMP                                  2   PTI8230            S
__   7.     DEPT (4+)                          3   PTI8230            S
__   8.     EMPPROJACT (3)                     3   PTI8230            S
__   9.     PROJ                               3   PTI8230            S
__  10.       PROJ (9+)                        4   PTI8230            S
__  11.       PROJACT (2+)                     4   PTI8230            S
__  12.   PROJ (9+)                            2   PTI8230            S
   ---------------------------------------------------------------------------
__   1. TABLE1                                 1   PTI8230
__   2.   TABLE2                               2   PTI8230            S
__   2.   TABLE2                               2   PTI8230            S
   ---------------------------------------------------------------------------
__   1. ADDRESS_TABLE                              PTI                ** NO RI
   ---------------------------------------------------------------------------
```

Notice that the header fields are carried over from the Main Menu.

Note: For more information about the header fields, see the “Using CA RC/Update” chapter.

Notice the following selection column fields for each table listed.

S (Selection)

Enter **S** next to the table to be edited.

XXXX

The sequential number of the table within the RI set. Each RI set of tables has its own set of numbers, beginning with 1 for the highest level parent table. In the previous example, the tables are numbered from 1 through 12, although there are only 6 tables in the set.

Note: Each solo table (one not associated with others by referential integrity) has an XXXX value of 1.

TABLE NAME

The table name. The table names in an RI set are displayed in an indented hierarchical fashion, so the parent-child relationships can be visualized. Each child table is shown underneath its parent, indented by two spaces.

Because of space limitations on the screen, only nine levels can be shown in the indented format with the full 18-character name field allowed. For RI sets with more than nine levels of tables, see the referential level number in the LVL column.

RI relationships can create referential cycles; that is, a parent table can be a child of its child. In the previous example, the table DEPT is its own child table. This is shown both by the appearance of DEPT indented under itself, and by the (4+) after the child DEPT, indicating that the table first appeared in the RI relationship at sequential number 4.

CREATOR

The creator (user ID) of the table.

GROUP

The Group Name of a group of referentially related sets of tables, when applicable. The Group Name can reflect the application for which the table sets are grouped together. The Group Name is created by the RI Manager component of the Data Navigator product.

Note: Group names are valid only for user-defined RI sets.

RI

The Referential Integrity type.

S

Specifies system-defined RI.

U

Specifies user-defined RI.

**** NO RI**

Specifies no RI relationship.

Interpreting the Edit Table Selection Panel

Within a referential set, the tables are displayed in an indented tree format, beginning with the highest level parent table, proceeding to its child tables, and so forth. Each child table appears below its parent table, indented two spaces. The LVL (Level Number) field also shows the relationships. Each dependency level is indented two spaces and assigned a LVL that indicates its level of dependency. Due to screen limitations, only nine levels can be displayed in the indented format. Higher levels of dependency are indicated only by the value of the LVL field.

Sequential Line Numbers

The tables within the referential set are numbered sequentially, beginning with *1* for the highest level table within the set. This sequence number is shown in the XXXX field for the table.

Solo Tables

Solo tables (those without any referential relationships) appear as single table referential sets, after all sets with RI relationships. Solo tables have an XXXX value of 1, and no LVL number is shown. The RI/Status fields contain the value ** NO RI. Solo tables are displayed after all referential sets. In the previous example, ADDRESS_TABLE is a solo table.

Referential Cycles

RI relationships can create referential cycles: a table can be *self-referencing* (a child table to itself), or it can be its own descendent (a child table to its own child table). In the previous example, DEPT is both a parent table and a child table of its child table EMP, as well as a child table of itself. This is illustrated both by the indented child tables and the increased LVL numbers. On line 4, DEPT has a LVL number of 1; on line 5, DEPT has a LVL number of 2; and on line 7, DEPT has a LVL number of 3.

Tables Listed Multiple Times

One of the results of the way the Edit Table selection panel displays referential cycles is that a given table can be listed several times within a referential set, indicating its complex relationships to the other tables. In the first referential set shown previously, there are 12 table entries for only six unique tables: ACT, PROJACT, EMPPROJACT, DEPT, EMP, and PROJ.

Previous Line Number Listed

The Edit Table selection panel also uses line number references to identify referential cycles and multiple relationships. After the second and subsequent occurrences of a table, the line number of the first occurrence appears in parentheses. In the previous example, on line 3, EMPPROJACT is a child table to PROJACT. On line 8, EMPPROJACT is also a child table to EMP. To indicate that EMPPROJACT has been previously listed in the referential set, the previous line number, 3, is listed in parentheses after the table name on line 8.

Plus Sign Indicates Suppressed Child Tables

The reference to the original line number includes a plus sign (+) if subsequent child tables have been suppressed from the display (because the referential cycle could go on forever). In the previous example, DEPT is both a parent table and a child table of its child table EMP, as well as a child table of itself. Notice that the two references to its initial Line ## are (4+), indicating that child tables have been suppressed from the display. A self-referencing table is always listed with the plus sign to indicate suppressed child tables in its second and subsequent listings on the display. In the previous example, both DEPT and PROJ are self-referencing tables.

Multiple Referential Relationships

It is possible for a pair of tables to have multiple referential relationships. In this case, the child table is listed for each relationship. In the previous example, TABLE1 is a parent table at LVL 1. TABLE2 is shown at LVL 2 twice below TABLE1, indicating that there are two referential relationships between TABLE1 and TABLE2. The unique constraint for TABLE1 is linked to one unique constraint in TABLE2 for one relationship, and to a different unique constraint in TABLE2 for the second relationship.

Selecting a Table

The table selected at the Edit Table selection panel becomes the Base Table until another one is selected from the same screen. This table is the starting point for displaying multiple tables in the referential set to which the table belongs. When a table is selected and Enter is pressed, the RI/Edit Options screen appears to allow users to specify the options for the edit session.

RI/Edit Options Screen

The RI/Edit Options screen's main function is to control the amount of data retrieved from the table for an edit session. A row limit and an SQL statement can be specified to control the number of rows returned. A test count can be requested before retrieving the rows to determine the selection's impact. This screen can be used to control the maximum number of characters to display per column. All these attributes can be saved as the default for the selected table to save time in future sessions.

The RI/Edit Options screen's header is unprotected and can be changed to edit another table or perform another option. There is no need to return to the Main Menu to perform a different function. Selection criteria for the Item Name and Creator prompts can be changed to retrieve the Edit Table selection panel.

Note: The Data Selection fields are used to restrict the data retrieved from the selected table. For information about the fields, press F1 (Help).

To return to the previous screen, press F3 (End). If the previous screen was the Table selection panel, that screen reappears, with the previous selection criteria still active. This permits easy selection of another table for editing. To return directly to the Main Menu, enter the MAIN command. If the previous screen was *not* the Table selection panel, the Main Menu appears.

Change Tables

To change to a different table, go back to the Edit Table Selection panel and select another table. A new Edit Options screen appears for the new table. This lets you view the default edit options for that table and make any changes before beginning the edit session.

Processing for RI/Edit Options

To begin the edit session, press Enter. The Data Retrieval In-Process screen appears while the system is retrieving the data for the table.

When the data has been retrieved, the RI/Edit screen appears.

RI/Edit Screen

The RI/Edit screen is like the RC/Edit Column Mode screen, and has the same editing capabilities, with additional commands and capabilities for working with RI structures. Users can edit one table here, as in the Column Mode screen, with the added feature of displaying the tables related to the current table, and then editing all of them. The display of multiple tables can be adjusted using the SIZE command, which is specific to this screen.

This screen uses only the Searched Update method.

Note: For more information about the Searched Update method, see the “RC/Edit” chapter.

Only Column Mode is supported. Column mode is used to edit multiple rows on the screen at one time.

A column is not displayed unless the entire column can fit on the screen (it is *not* split between screens). When scrolling left or right, the number of displayed columns will vary based on their definition. If only one column appears on the screen, the next column is a large character column and cannot fit on the screen with the current column. Scroll right to view the other columns.

Control the display length of character data with the Max Char field in the header. By entering a small value, more columns can be viewed on one screen.

The screen header provides information about the table or tables being edited. The Max Char field can be changed, and another table can be assigned to the Current field with the CURRENT command.

Note: For information about the fields, press F1 (Help).

More columns can be viewed on the screen by shrinking the character width. The EXPLODE Facility can also be used to view the whole column in an ISPF Edit session.

Note: For more information, see the "RC/Edit" chapter.

More information:

Displaying Multiple RI-Related Tables (see page 171)

Processing for RI/Edit Screen

The RI/Edit screen is designed to maximize row operations. Edit operations can be performed on as many rows as needed. Full ISPF-type scrolling is available, and extensive type checking is performed to help ensure valid data.

The ISPF-like line-editing commands available on the RI/Edit screen are shown in the following. The RI Editor line commands are the same as in RC/Edit, with one additional command. E (for Exception) is used with the SYNC (synchronized scrolling) command; it displays dependent rows that do not match any displayable parent rows.

In the OPT (option), enter a line editing command:

A

Specifies a point after which to move or copy lines.

B

Specifies a point before which to move or copy lines.

C, C*nn*, or CC

Copies a line, a number of lines, or block of lines, respectively.

D, D*nn*, or DD

Deletes a line, a number of lines, or block of lines, respectively.

E

Inserts an informational line with the message EXCEPTION ROW into the Current Table display, and changes the status of that row to E. When the Exception Row appears in the parent table, the dependent tables listed display foreign key rows that do not match parent rows. Active only in synchronized scrolling (SYNC) mode.

F or F*nn*

Shows the first line of a block of excluded lines or shows the first *nn* number of lines from an excluded block.

I or I*nn*

Inserts a line or a number of lines.

K

Inserts an information line in the Active Table display, indicating the primary (Pn) and foreign key (FK) columns of the table.

L or L*nn*

Shows the last line of a block of excluded lines or shows the last *nn* number of lines from an excluded block.

M, M*nn*, or MM

Moves a line, a number of lines, or block of lines, respectively.

R, R*nn*, or RR

Repeats a line, a number of lines, or block of lines, respectively.

S, S*nn*, or SS

Shows an excluded line, a number of excluded lines, or block of excluded lines, respectively.

U, U*nn*, or UU

Undeletes a line, a number of lines, or block of lines, respectively.

X, X*nn*, or XX

Excludes a line, a number of lines, or block of lines, respectively.

As in ISPF, single line commands operate on individual rows (like, entering D deletes that row). Delete, Undelete, Exclude, Show, Copy, Repeat, and Move support block mode. Enter **DD**, **UU**, **XX**, **SS**, **CC**, **RR**, or **MM** at the start and stop rows of the block to cause the action to be performed on all rows within that block, inclusive of the rows containing the block commands. Each command can also be followed by a number to indicate multiple occurrences (for example, I20 inserts 20 new rows; X13 excludes 13 rows). For the MOVE command, the user must specify a text destination.

Use the **B** and **A** commands to move marked text before or after a specific line.

If line commands have been entered in error, the RESET command can be entered to remove the invalid entries.

Note: Although RI Editor lets you move rows, DB2 does not move the rows, because by definition a relational database is non-positional. The MOVE command allows row movement for readability only while in RI Editor.

EXCLUDE Command Notes

If a FIND command is issued that searches for text in an excluded line, the text will be found and the line unexcluded. If a CHANGE command is issued that affects text in an excluded line, the text will be changed and the line unexcluded. There are also line commands that affect excluded lines.

F*n*

Shows the first *n* excluded lines (for example, F6 shows the first six excluded lines).

L*n*

Shows the last *n* excluded lines (for example, L14 shows the last 14 excluded lines).

SS

Shows a block of text (when entered on the first and last lines of the block). This command is useful for showing more than one excluded block of text.

The following describes the S column that follows the OPT field:

S

Displays the current status of the row. A blank status denotes a row in its initial fetched format. Valid values are:

D

Indicates that the row has been deleted. The SHOW option must be turned on to view deleted rows.

E

Indicates that the row is an information row with the message EXCEPTION ROW, as a result of the E line command. When the Exception Row appears in the parent table, the dependent tables listed display foreign key rows that do not match parent rows. This is active only in synchronized scrolling (SYNC) mode.

I

Indicates that the row has been inserted. If the R line command is used, the inserted rows will have a status of I. If an inserted row is deleted and then undeleted, the status returns to inserted.

K

Indicates that the row contains the primary and foreign key information as a result of entering the K (Key Field) command.

U

Indicates that the row has been updated.

X

Indicates that the row has been excluded. If a block of rows was excluded, one status line for the block appears. A message indicating the number of excluded lines is also displayed.

Note: For information about the other fields, press F1 (Help).

There are many primary edit commands that can be entered on the command line or assigned to PF keys for controlling the column display, for example, FIND, RFIND, CHANGE, RCHANGE, CHANGE ALL, SORT, TYPE.

Displaying Multiple RI-Related Tables

Displaying more than one RI-related table on the screen takes two steps. First, from the RI/Edit screen, enter **REL** in the command line to display the RI/Edit: Related Table List screen. This lists all the tables that have a referential integrity relationship with the displayed table. Second, choose the tables to include in the display when returning to the RI/Edit screen. Due to the size of the display area of different monitors, the number of rows displayed initially depends on the number of selected tables.

```
RUEDITG        ---------- RI/Edit: Related Table List ---------
COMMAND ===>                                                SCROLL ===> PAGE

Base Table   : 4.USER5.GCHILD
Current table: 3.USER5.CHILD3
SSID: D81B ------------------------------------------------------------ USER3

S XXX  TABLE NAME         CREATOR   TYPE      RULE NAME  WHERE      ROWLIMIT
s   1. PARENT2            USER5     PARENT    SC1        N ________ ________
    3. CHILD3             USER5     CURRENT
    4. GCHILD             USER5     CHILD               N ________ ________
s   5. GCHILD2            USER5     CHILD     SC2       N ________ ________
 ******************************** BOTTOM OF DATA ********************************
```

Notice that there are no editable fields on the CHILD3 line of this sample screen, because CHILD3 has been set as the Current Table. This sample screen is ready to have tables selected for multiple table windows. Notice that two tables, PARENT2 and GCHILD2, have an S in their S (Selection) fields. Pressing Enter selects these tables to be added to any that are already displayed on the RI/Edit screen with the Current Table CHILD3 (shown in the next example). The header fields on this screen are for display only.

Note: For information about the fields, press F1 (Help).

When tables have been selected to display along with the Current Table, press Enter. The RI/Edit screen appears again, displaying the base table and the selected tables.

```
RUEDITH        ----------------- CA RI/Edit -------------------
COMMAND ===>                                                SCROLL ===> PAGE

Table   ==> 1.USER5.PARENT2                 Row number ==> 1 OF 4
Current ==> 3.USER5.CHILD3                  Max Char   ==> 070 Active ==> 2
SSID: D81B ------------------------------------------------------------ USER3
OPT S SP1    SP2    SP3
___   000001 000002 0000021110
___   000002 000011 0000021111
___   000004 000003 0000021211
-------------------------------------------------------------------------------
3.USER5.CHILD3          ==>  4   C  SC2                     1 OF 5
OPT S SC1    SC2    SC3
___   000010 000001 000002Y112
___   000001 000002 0000024101
___   000006 000004 0000024110
___   000004 000005 0000002000
___   000006 000004 0000002001
-------------------------------------------------------------------------------
4.USER5.GCHILD           BASE TABLE                         1 OF 8
OPT S SC1    SC2    SC3
___   000001 000001 0000000000
___   000002 000001 0000000000
___   000003 000001 0000000000
-------------------------------------------------------------------------------
5.USER5.GCHILD2         <==  3   R  SC2                     1 OF 3
OPT S SC1    SC2    SC3
___   000001 000001 0000000000
___   000006 000006
___   000002 000010
```

By making another table the current table, and then selecting additional tables from the Related Table List screen, more related tables can quickly be displayed and identified.

Note: For information about the fields, press F1 (Help).

RI Editor automatically adjusts the number of rows for each table's display area based on the number of chosen tables. The display can be adjusted further by using the RI/Edit: Display Control screen discussed in the next section.

Display Control Screen

The Display Control screen lets you change the display of tables on the RI/Edit screen by:

- Changing the number of rows displayed for each table (within the constraints of your monitor size)
- Temporarily reducing the number of tables to be displayed
- Rearranging the order of the displayed tables (using the M line command).

When the primary command SIZE is issued at the RI/Edit screen, the Display Control screen appears.

Note: For information about the fields, press F1 (Help).

When finished adjusting the numbers, press Enter to recalculate the totals. If there are no error messages, press F3 to return to the RI/Edit screen to use the new display in working with the tables.

Identifying Unmatched Foreign Key Rows

RI/Edit lets you insert informational lines, with the message EXCEPTION ROW, into a parent table display, which changes the status of that row to E. (One or more Exception Rows can be created.) When scrolling to the Exception Row of a parent table, the dependent tables automatically display the rows that do not match any row in the parent table. This command is active only in synchronized scrolling (SYNC) mode.

The following example shows a sample RI/Edit screen before the E line command is issued.

```
RUEDITH        ----------------- CA RI/Edit ------------------
COMMAND ===>                                                SCROLL ===> PAGE

Table   ==> 1.USER5.RI_PARENT                Row number ==> 1 OF 20
Current ==> 1.USER5.RI_PARENT                Max Char   ==> 070 Active ==> 1
SSID: D81B ------------------------------------------------------------ USER3
OPT S KEY1   KEY2 NONKEY1    NONKEY2
e__   AAAAAA AAA  FIRST      FIRST
-------------------------------------------------------------------------------
2.USER5.RI_CHILD  <==  1   N  USER5RI                        1 OF 1
OPT S N:DATA1               N:DATA2               N:FORNKEY1 N:FORNKEY2
___   N FIRST               N FIRST               N AAAAAA   N AAA
****************************** BOTTOM OF DATA ********************************

-------------------------------------------------------------------------------
3.USER5.RI_CHILD3  <==  1   N  FORNKEY1                      1 OF 1
OPT S N:DATA1               N:DATA2               N:FORNKEY1 FORNKEY2
___   N 1                   Y -------------------  N AAAAAA   AAA
****************************** BOTTOM OF DATA ********************************
```

The following example shows a sample RI/Edit screen after the E line command is issued. The exception row is absent because it was inserted after the current line of the parent table.

```
RUEDITH        ------------------ CA RI/Edit -----------------
COMMAND ===>                                                SCROLL ===> PAGE

Table   ==> 1.USER5.RI_PARENT                Row number ==> 1 OF 21
Current ==> 1.USER5.RI_PARENT                Max Char   ==> 070 Active ==> 2
SSID: D81B ------------------------------------------------------------ USER3
OPT S KEY1   KEY2 NONKEY1    NONKEY2
___   AAAAAA AAA  FIRST      FIRST
--------------------------------------------------------------------------------
2.USER5.RI_CHILD  <==  1   N  USER5RI                        1 OF 1
OPT S N:DATA1               N:DATA2               N:FORNKEY1 N:FORNKEY2
___   N FIRST               N FIRST               N AAAAAA   N AAA
****************************** BOTTOM OF DATA ********************************

--------------------------------------------------------------------------------
3.USER5.RI_CHILD3  <==  1   N  FORNKEY1                      1 OF 1
OPT S N:DATA1               N:DATA2               N:FORNKEY1 FORNKEY2
___   N 1                   Y -------------------  N AAAAAA   AAA
****************************** BOTTOM OF DATA ********************************
```

The following example shows the result of scrolling with F8 (Down) to the parent table's exception row. Now the first child table shows no data rows, which means that there are no exceptions. Every row in the child table has a parent row.

The second child table shows a row containing AAAAAD in the FORNKEY1 column and CCC in the FORNKEY2 column. This means that a row matching the foreign keys does not exist in the edit session.

```
RUEDITH        ----------------- CA RI/Edit -----------------
COMMAND ===>                                                 SCROLL ===> PAGE

Table   ==> 1.USER5.RI_PARENT                Row number ==> 2 OF 21
Current ==> 1.USER5.RI_PARENT                Max Char   ==> 070 Active ==> 1
SSID: D81B------------------------------------------------------------ USER3
OPT S KEY1   KEY2 NONKEY1    NONKEY2
___ E - - E X C E P T I O N   R O W - - - - - - - - - - - - - - - - - - - -
--------------------------------------------------------------------------------
2.USER5.RI_CHILD  <==  1   N  USER5RI                        0 OF 0
OPT S N:DATA1               N:DATA2               N:FORNKEY1 N:FORNKEY2
****************************** BOTTOM OF DATA ********************************

--------------------------------------------------------------------------------
3.USER5.RI_CHILD3  <==  1   N  FORNKEY1                      1 OF 1
OPT S N:DATA1               N:DATA2               N:FORNKEY1 FORNKEY2
___ U N 2                   Y -------------------  N AAAAAD   CCC
****************************** BOTTOM OF DATA ********************************
```

The E command saves time in identifying unmatched key columns that need editing. Use the same screen to insert a new row or edit a column in the parent or child table if necessary.

LINK Command

The LINK command is available from the RI/Edit or RI/Browse screen. The LINK command simulates a referential integrity relationship between the current table and another table, based on the unique constraints of the current table. Issuing the LINK command displays the RI/Edit Column Mapping panel used to create the pseudo-link.

You can use the LINK command dynamically to create a pseudo-referential relationship between the current table and another table based on the unique constraints of the current table.

Based on the parameters entered with the LINK command, users can display the Table selection panel; select the target table to display on the RI/Edit Column Mapping panel; or specify the table and go directly to the RI/Edit Column Mapping panel.

On the RI/Edit Column Mapping panel, the target (foreign key) columns to which the current table's unique constraints will be mapped are specified. Users can then browse, edit, synchronize, and perform RI checking exactly as they would for a real system- or user-defined RI rule.

The mode is Edit and Browse.

The command syntax is:

```
LINK [creator.tablename]
```

The following three methods can be used to display the RI/Edit Column Mapping panel from the RI/Edit screen.

- Enter LINK with no parameters. This displays the Table selection panel, where the target table for mapping columns is selected.
- Enter LINK with creator and table parameters indicating your selection criteria. This also displays the Table selection panel, with tables listed according to selection criteria. A mask using the creator or table name, or both can be used.
- If known, LINK can be entered with the specific target table creator and name. This displays the RI/Edit Column Mapping panel directly, bypassing the selection panel.

Linking Restrictions

There are restrictions that apply for linking tables. First, users can link to only one table. Second, users cannot link from a linked table. Finally, the LINK command cannot be used in a session that already contains more than one table.

The following sample of the RI/Edit screen illustrates how to enter the LINK command with a selection mask for the table name.

```
RUEDITH        ----------------- CA RI/Edit ------------------
COMMAND ===> link USER2.proj%                                SCROLL ===> PAGE

Table   ==> 3.USER2.EMPPROJACT                Row number==> 1 OF 3
Current ==> 3.USER2.EMPPROJACT                Max Char  ==> 070 Active ==> 3
SSID: D81B ------------------------------------------------------------ USER3
OPT S EMPNO  PROJNO ACTNO   N:EMPTIME N:EMSTDATE   N:EMENDATE
___   000010 MA2100     10  N    0.50 N 1982-01-01 N 1982-11-01
___   000010 MA2110     10  N    1.00 N 1982-01-01 N 1983-02-01
___   000010 AD3100     10  N    0.50 N 1982-01-01 N 1982-07-01
****************************** BOTTOM OF DATA ********************************
```

The following Table selection panel shows the results of the LINK command using the mask USER2.PROJ% and the line command for selecting table USER2.PROJACT. PROJACT is selected to be used as the target table for mapping with the source table USER2.EMPPROJACT shown in the previous RI/Edit screen example.

```
RURPLSTT            ----- CA, Table Selection ---------
CMD ==>                                                    SCROLL==> PAGE

Item Name => PROJ%            > Creator  => USER2   > Where => N
------------------------------------------------------ USER3       >

    Enter 'S' to select a Table.

Sel TABLE NAME         CREATOR  DATABASE  TSNAME    COLCOUNT  RECLEN
 __ PROJ               USER2    USER2D31  PROJ          8        70
 s_ PROJACT            USER2    USER2D31  PROJACT       5        29
 __ PROJ2              USER2    USER2D31  PROJ          8        70
 ***************************** BOTTOM OF DATA ****************************
```

When a selection is made from the Table selection panel, the RI/Edit Column Mapping panel appears. This screen is described in the next section.

RI/Edit Column Mapping Panel

The RI/Edit Column Mapping panel provides a convenient way to simulate a referential integrity relationship between the current table and another table. The RI/Edit Column Mapping panel appears when the LINK command is issued on the RI/Edit screen. It displays the column names of the two tables side by side, so the current table's unique constraints can be mapped to the target (foreign key) columns.

When the columns are mapped, users can browse, edit, and synchronize, exactly as if the tables were related by a system- or user-defined referential integrity rule. The pseudo relationships are maintained until the current RI Editor session ends.

The RI/Edit Column Mapping panel has two independent scrolling areas.

Note: For information about the fields, press F1 (Help).

RESET Command

The RESET primary command clears the current column mapping and lets you restart the process.

The command syntax is:

RESET

Results of Linking and Mapping

When F3 is pressed from the RI/Edit Column Mapping panel, the RI/Edit screen reappears, showing the results of the linking and mapping.

Tables EMPPROJACT and PROJACT and the mapped rows show the pseudo relationship that was created. The area where the delete rule (R, C, or N) usually appears contains the word JOIN instead, to show that the usual delete rules are not in effect because the RI relationship indicated is for display only in this edit session.

Note: The pseudo relationships are maintained until the RI Editor session ends. No referential integrity relationships are created or altered except for synchronizing the display within this session, and no delete rules apply.

Synchronized Scrolling: The SYNC Command

The synchronized scrolling feature, available from the RI/Edit screen, provides a convenient way to update matching data in referential sets. When a parent table and one or more child tables are displayed together, the child rows that have foreign keys matching the current row in the parent table can automatically be displayed. Scrolling from row to row in the parent table, the child table's rows are automatically scrolled so that the matching data appears. To change to synchronized scrolling (SYNC) mode, enter the primary command SYNC on the RI/Edit screen as shown in the following example, then press Enter.

When the SYNC command is issued, the message SYNCHRONIZED SCROLLING AND RI/CHECKING HAVE BEEN ACTIVATED appears. The screen displays only one row for each parent table, and each child table displays the dependent rows of its parent table row.

RI/Checking has been activated means that the CHECK ON command has been issued, and users will be prevented from making table updates that violate referential integrity.

Note: The CHECK ON feature only checks data that is retrieved by RI/Edit.

If users try to enter rows based on what is on screen, but data queries had been used to limit data retrieval, then actual changes may in fact *not* be made when DB2 tries to make the update.

To remove this constraint, enter CHECK OFF and press Enter. Keep in mind that CHECK OFF does not allow users to bypass restraints imposed by DB2. When RI/Edit tries to make a change that is not allowed by DB2, DB2 will catch the violation.

In the following example, the data in FORNKEY1 and FORNKEY2 of the two child tables matches KEY1 and KEY2 in the parent table. KEY1 is AAAAAA, and KEY2 is AAA.

```
RUEDITH        ----------------- CA RI/Edit ------------------
COMMAND ===>                                                 SCROLL ===> PAGE
RU548I: Synchronized scrolling and RI/Checking has been activated.
Table   ==> 1.USER5.RI_PARENT                Row number ==> 1 OF 20
Current ==> 1.USER5.RI_PARENT                Max Char   ==> 070 Active ==> 1
SSID: D81B ------------------------------------------------------------ USER3
OPT S KEY1   KEY2 NONKEY1    NONKEY2
___   AAAAAA AAA  FIRST      FIRST
--------------------------------------------------------------------------------
2.USER5.RI_CHILD  <==  1   N  USER5RI                        1 OF 1
OPT S N:DATA1                N:DATA2               N:FORNKEY1 N:FORNKEY2
___   N FIRST                N FIRST               N AAAAAA   N AAA
****************************** BOTTOM OF DATA ********************************

--------------------------------------------------------------------------------
3.USER5.RI_CHILD3  <==  1   N  FORNKEY1                      1 OF 1
OPT S N:DATA1                N:DATA2               N:FORNKEY1 FORNKEY2
___   N 1                    Y -------------------  N AAAAAA   AAA
****************************** BOTTOM OF DATA ********************************
```

The next example shows the result of scrolling with F8 (Down). The parent table displays the next row, and the child tables show no matches for the AAAAAA and BBB data in the KEY1 and KEY2 columns.

```
RUEDITH        ----------------- CA RI/Edit ------------------
COMMAND ===>                                                 SCROLL ===> PAGE

Table   ==> 1.USER5.RI_PARENT                Row number ==> 2 OF 20
Current ==> 1.USER5.RI_PARENT                Max Char   ==> 070 Active ==> 1
SSID: D81B ------------------------------------------------------------ USER3
OPT S KEY1   KEY2 NONKEY1    NONKEY2
___   AAAAAA BBB  SIXTH      SIXTH
--------------------------------------------------------------------------------
2.USER5.RI_CHILD  <==  1   N  USER5RI                        0 OF 0
OPT S N:DATA1                N:DATA2               N:FORNKEY1 N:FORNKEY2
****************************** BOTTOM OF DATA ********************************

--------------------------------------------------------------------------------
3.USER5.RI_CHILD3  <==  1   N  FORNKEY1                      0 OF 0
OPT S N:DATA1                N:DATA2               N:FORNKEY1 FORNKEY2
****************************** BOTTOM OF DATA ********************************
```

The next example shows the result of scrolling with F8 (Down) again. The parent table displays the next row. The first child table shows three matches for the AAAAAA and CCC data in the parent table's KEY1 and KEY2 columns, and the second child table shows one matched row.

```
RUEDITH         ----------------- CA RI/Edit -----------------
COMMAND ===>                                                  SCROLL ===> PAGE

Table   ==> 1.USER5.RI_PARENT                Row number ==> 3 OF 20
Current ==> 1.USER5.RI_PARENT                Max Char   ==> 070 Active ==> 1
SSID: D81B ------------------------------------------------------------ USER3
OPT S KEY1   KEY2 NONKEY1    NONKEY2
___   AAAAAA CCC  FIRST      THIRD
--------------------------------------------------------------------------------
2.USER5.RI_CHILD  <==  1   N  USER5RI                          1 OF 3
OPT S N:DATA1                N:DATA2                N:FORNKEY1 N:FORNKEY2
___   N FIRST                N UPDATE               N AAAAAA   N CCC
___   N FIRST                N UPDATE               N AAAAAA   N CCC
___   N FIRST                N UPDATE               N AAAAAA   N CCC
****************************** BOTTOM OF DATA ********************************

--------------------------------------------------------------------------------
3.USER5.RI_CHILD3  <==  1   N  FORNKEY1                        1 OF 1
OPT S N:DATA1                N:DATA2                N:FORNKEY1 FORNKEY2
___   N 2                    Y ------------------   N AAAAAA   CCC
****************************** BOTTOM OF DATA ********************************
```

By seeing the matches on the same screen, you can edit each table as needed. To find foreign keys that *do not* have a match in the parent table, you can use the E command described in the next section. To turn off the synchronized scrolling mode and return to the regular display for the RI/Edit screen, enter SYNC OFF and press Enter.

More information:

RI Editor Command Reference (see page 181)

RI Editor Command Reference

RI Editor supports a full range of primary commands for controlling the display and performing edit-related functions. These commands can be assigned to F keys or entered on the command line. Most RI Editor primary commands are the same as those used by RC/Edit, and work the same way. This section includes only the primary commands that are unique to the RI Editor facility.

Notational Conventions for RI Editor Commands

The following table lists notational conventions that are used in this document for command syntax:

Notation	Description
UPPER CASE characters	Must be entered as shown.
lower case characters	User-specified variables.
<*column name*>	The full DB2 column name or the RC/Edit column abbreviation.
( )	Must be entered where shown.
[ ]	Enclose optional parameters, choose 1.
{ }	Enclose required parameters.
|	Or, as in Y | N.
Underlined	Abbreviations.

Directed Command Parameters

Some of the RI Editor primary commands can be issued as directed commands on a screen displaying multiple tables. Users can direct the command to operate on a specific table by specifying its sequence number in the referential integrity set you are displaying. The syntax is:

command.sequence

(where *command* is the command to be executed and *sequence* is the sequence number shown to the left of the table name to be the target of the command)

In the following example, the current table is 1.USER5.RI_PARENT. If CURRENT.3 is entered, the current table changes to 3.USER5.RI_CHILD3.

```
RUEDITH        ----------------- CA RI/Edit ------------------
COMMAND ===>                                                SCROLL ===> PAGE

Table   ==> 1.USER5.RI_PARENT                Row number ==> 3 OF 20
Current ==> 1.USER5.RI_PARENT                Max Char   ==> 070 Active ==> 1
SSID: D81B ------------------------------------------------------------ USER3
OPT S KEY1   KEY2 NONKEY1    NONKEY2
___   AAAAAA CCC  FIRST      THIRD
--------------------------------------------------------------------------------
2.USER5.RI_CHILD  <==  1   N  USER5RI                        1 OF 3
OPT S N:DATA1                N:DATA2               N:FORNKEY1 N:FORNKEY2
___   N FIRST                N UPDATE              N AAAAAA   N CCC
___   N FIRST                N UPDATE              N AAAAAA   N CCC
___   N FIRST                N UPDATE              N AAAAAA   N CCC
****************************** BOTTOM OF DATA ********************************

--------------------------------------------------------------------------------
3.USER5.RI_CHILD3  <==  1   N  FORNKEY1                      1 OF 1
OPT S N:DATA1                N:DATA2               N:FORNKEY1 FORNKEY2
___   N 2                    Y -------------------  N AAAAAA   CCC
****************************** BOTTOM OF DATA ********************************
```

The directed commands, described in the following, include ACTIVE, CURRENT, MAX, RELATED, and REMOVE.

RI Editor Commands

This section describes the following primary commands, which are unique to RI Editor. With RI Editor, you can use any of the primary commands available with the RC/Edit facility.

Note: Some RC/Edit commands (for example, SORT), if used in RI/Editor while SYNC ON displays are active, affect only the current display set.

- ACT (ACTIVE) (see page 184)
- CHECK (see page 185)
- CUR (CURRENT) (see page 185)
- INSERT (see page 186)
- LINK (see page 186)
- MAX (see page 187)

- MIN (see page 187)
- REL (RELATED) (see page 187)
- REM (REMOVE) (see page 188)
- RSIZE (see page 188)
- SIZE (see page 188)
- SYNC (see page 189)
- TREE (see page 189)

ACT (ACTIVE)

Changes the active table (the scrollable table, on an RI/Edit screen that has more than one table displayed).

Issuing the ACT command by itself (without specifying a table sequence number) changes the active table to the next table displayed, progressing from the top of the screen.

The Active Table can be changed without using ACT by pressing Enter while the cursor is in the table window to be active.

The command syntax is:

```
ACTIVE [.sequence number]
```

The mode is Edit or Browse.

CHECK

Turns Referential Integrity (RI) checking on or off. When ON, a message appears when an attempt is made to make an update that violates referential integrity constraints among the selected related tables, as in the following example.

```
RU557I: THE DELETE ATTEMPT WOULD VIOLATE REFERENTIAL CONSTRAINT:

EMPNO PARENT: 6.AAAUSER5.EMP
CHILD: 8.AAAUSER5.EMPPROJACT.
```

Note: This type of message appears for an **insert** only when Synchronized Scrolling (SYNC) is on, **not** when SYNC is off and CHECK is on.

RI checking is turned on automatically when the SYNC command is issued for synchronized scrolling.

RI checking applies only to the set of tables selected on the RI/Edit: Related Table List screen.

The command syntax is:

```
CHECK [ON] CHECK OFF
```

The mode is Edit.

CUR (CURRENT)

Changes the Current Table on an RI/Edit screen that has more than one table displayed. (The Current Table is the one affected by the RELATED command described in the following).

Issuing the CUR command by itself (without specifying a table sequence number) changes the Current Table to the next table displayed, progressing from the top of the screen.

The command syntax is:

```
CURRENT[.sequence number]
```

The mode is Edit and Browse.

INSERT

The INSERT command is used during synchronized scrolling mode to insert a row into a dependant table. The command may ONLY be applied to a dependant table in which zero rows are being displayed. It MUST be qualified (for example, INSERT.3) to direct it to a particular table.

The foreign key columns of the inserted row contain the values from the parent key columns of the currently displayed parent table row.

The command syntax is:

`INSERT.`*`sequence number`*

The mode is EDIT.

LINK

You can use the LINK command to dynamically create a pseudo-referential relationship between the current table and another table, based on the parent key of the current table.

Based on the parameters you enter with the LINK command, you can display the Table selection panel, to select the target table to display on the RI/Edit Column Mapping panel, or you can specify the table and go directly to the RI/Edit Column Mapping panel.

On the RI/Edit Column Mapping panel you specify the target (foreign key) columns to which the current table's unique constraints will be mapped. Then, you will be able to browse, edit, synchronize, and perform RI checking exactly as you would for a real system- or user-defined RI rule.

The command syntax is:

`LINK [`*`creator.tablename`*`]`

The mode is Edit and Browse.

MAX

Changes the size of a table on the RI/Edit screen to fill the whole screen. You can use the MIN command to return the table display to its original size.

Issuing the MAX command by itself without specifying a table sequence number changes the Active Table display size to fill the whole screen.

The command syntax is:

```
MAX [.sequence number]
```

The mode is Edit and Browse.

More information:

MIN (see page 187)

MIN

Changes the RI/Edit screen's Active Table display back to the size it was before the MAX command was issued (see the MAX command).

The command syntax is:

```
MIN
```

The mode is Edit and Browse.

REL (RELATED)

Displays the RI/Edit: Related Table List screen. This screen lists all the tables that have a referential integrity relationship with the Current Table. Use the screen to select related tables for display on the RI/Edit screen. Available from the RI/Edit screen. Issuing the REL command with a sequence number also makes that table the Current Table.

The command syntax is:

```
RELATED [.sequence number]
```

The mode is Edit and Browse.

REM (REMOVE)

Removes a table from the RI/Edit session.

Issuing the REM command by itself (without specifying a table sequence number) removes the last table selected from the Related Table list screen.

The command syntax is:

```
REMOVE [.sequence number]
```

The mode is Edit and Browse.

RSIZE

Changes the number of rows displayed for the Active Table on the RI/Edit screen. If the cursor is outside the Active Table when Enter is pressed, the number of rows displayed for the Active Table becomes larger. If the cursor is inside the Active Table when Enter is pressed, the number of rows displayed becomes smaller. The command can be repeated to continue adjusting the size.

The command syntax is:

```
RSIZE
```

The mode is Edit and Browse.

SIZE

Displays the RI/Edit: Display Control screen. This screen allows the adjustment of display size and display order of tables currently shown on the RI/Edit screen. Tables can be temporarily removed from the display without removing them from the referential set selected for editing.

The command syntax is:

```
SIZE
```

The mode is Edit and Browse.

SYNC

Turns Synchronized Scrolling on or off. Available from the RI/Edit screen, SYNC mode provides a way to update matching data quickly in referential sets. When a parent table and one or more child tables are displayed together, the child table rows that have foreign key data matching the current row in the parent table can be displayed automatically. As you scroll in the parent table, each matching foreign key data row appears in the child table. The RI Checking feature is turned on automatically when SYNC mode is turned on (see CHECK).

The command syntax is:

```
SYNC [ON|OFF]
```

The mode is Edit and Browse.

TREE

Displays referential sets of tables in an indented tree hierarchy at the RI/Edit: Related Table List screen.

The command syntax is:

```
TREE [ON|OFF]
```

The mode is Edit and Browse.

Command Summary

This following table lists all the available primary commands.

Note: For more information about commands that are on this list but not included in this chapter, see the "RC/Edit" chapter.

Command	Syntax	Mode
ACTIVE	ACTIVE [.*sequence number*]	E,B
CANCEL	CANCEL	E
CAPS ON	CAPS ON	E
CAPS OFF	CAPS OFF	E
CHANGE	CHANGE *old_value new_value* [*column_name*] [ALL] [X | NX]	E
CHECK ON	CHECK [ON]	E
CHECK OFF	CHECK OFF	E

Command	Syntax	Mode
COMMA	COMMA ON, COMMA OFF	E,B
CURRENT	CURRENT [*.sequence number*]	E,B
END	END	E,B
EXPLODE	EXPLODE	E,B
FIND	FIND *value* [*column_name*] [ALL] [X | NX]	E,B
FREEZE	FREEZE *column_name*	E,B
INSERT	INSERT *.sequence number*	E
LCASE	LCASE *column_name*	E
LINK	LINK [*creator.tablename*]	E,B
MAX	MAX [*.sequence number*]	E and B
MELT	MELT	E,B
MIN	MIN	E and B
PROTECT	PROTECT <*column name*>	E
RCHANGE	RCHANGE	E
RESET	RESET	E
RELATED	RELATED [*.sequence number*]	E,B
REFRESH	REFRESH	E
REMOVE	REMOVE [*.sequence number*]	E,B
RFIND	RFIND	E,B
RSIZE	RSIZE	E,B
SAVE	SAVE	E
SCOL	SCOL [*column name*|@ALL]	E
SET	SET <*column_name*> = *new_value* [WHERE ...] The SET command works only when you are using the Searched Update method.	E
SHOW	SHOW **Note:** The SHOW command works in Column Display mode only.	E
SIZE	SIZE	E,B
SORT	SORT *column_name* [A | D]	E
SORT ?	SORT ?	E
SQL	SQL	E,B

Command	Syntax	Mode
STATS	STATS	E
SYNC ON	SYNC [ON] [.*sequence number*]	E,B
SYNC OFF	SYNC OFF	E,B
TRACE	TRACE [ON | OFF]	E
TREE ON	TREE [ON]	E,B
TREE OFF	TREE OFF	E,B
TYPE	TYPE	E,B
UCASE	UCASE *column_name*	E
UNFREEZE	UNFREEZE *column_name*	E,B
UNPROTECT	UNPROTECT [*column name*]	E
XCOL	XCOL {*column_name* | @ALL}	E

Chapter 9: RC/Browse

This section contains the following topics:

RC/Browse Component

RC/Browse is a restricted form of RC/Edit and lets users view or browse data in DB2 tables without editing. The data can be viewed in column or form mode. Standard ISPF editor commands such as FIND and RFIND are supported. The extended view browse support feature permits editing of most joined views. There is also an EXPLODE mode available for viewing data in long character columns.

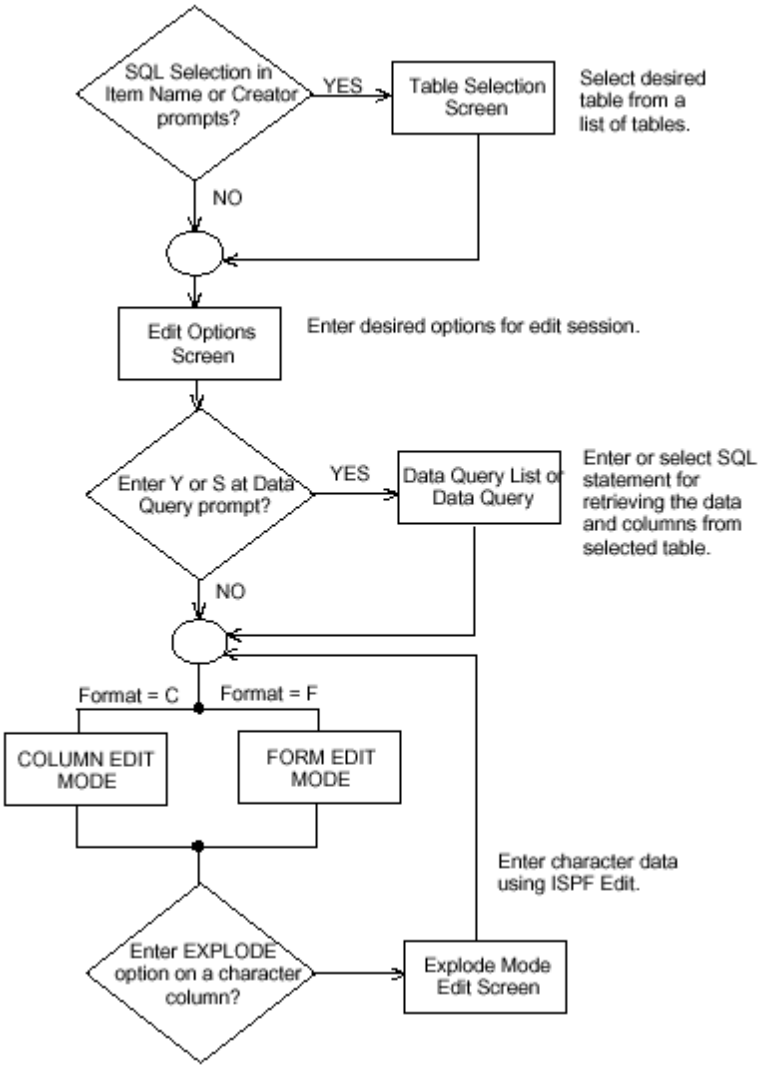
RC/Browse provides a significant ISPF-like data editor to browse information stored in DB2 tables. Some of the capabilities of the RC/Browse include:

- Access to DB2 data in tabular (column) or form mode. Browsing is beneficial for viewing DB2 catalog or production tables.
- Support for most DB2 data types including LONG VARCHAR, TIME, DATE, TIMESTAMP, TIMESTAMP with TIMEZONE, and floating-point.
- An EXPLODE option for browsing data greater than 256 characters. The EXPLODE option places the user into an ISPF full screen browse session at the column level.
- Full support for NULL values.
- Support for embedded blanks at the beginning or within values for the following items:
  - Column names
  - Table names
  - Creators
- Support for long names.

- Selection capability on columns and rows. An SQL statement can be specified to select only specific columns and rows. These SQL statements can be saved for repeated use.
- A SORT feature for sorting the rows while they are being browsed. This permits users to constantly change the order of the data without re-executing the SQL statement.
- A FREEZE option for keeping columns fixed during scrolling. This facilitates examining columns dependent on other columns.
- FIND and RFIND commands for finding specific data during the browse session.
- A K command to automatically identify the primary (Pn) and foreign (FK) columns in a table.
- The extended view support feature lets users browse most joined views on the BROWSE screen, using the Column Mode. This is accomplished by mapping the columns in the views to the underlying base tables.

Processing Flow

The following diagram depicts a possible processing flow:



Locking Considerations

Locking is not a concern in browse mode.

HEX Mode Support

RC/Browse accepts and processes columns with hexadecimal data. If data is retrieved that contains nondisplayable (hex) characters, the hex characters are displayed as periods and the message HIGHLIGHTED CHAR TYPE COLUMNS CONTAIN INVALID (NONDISPLAY) CHARACTERS appears.

To browse the hex characters, use the RC/Browse column mode screens or the RC/Browse Explode Facility screen. ISPF Browse will be invoked with HEX on if there are nondisplayable characters. Toggle the ISPF command off and on by entering HEX in the command line.

More information:

RC/Browse Command Reference (see page 200)

Table Selection

The Table Selection facility lets you select a table to browse from a list. You can control the list of tables displayed by entering selection criteria for the Item Name or Creator prompts that appear in the header. An extended query can also be entered to further refine the list of displayed tables. (If a specific table name is entered, this screen is not displayed.)

Note: Embedded blanks are supported in table and creator names. When specifying a value for Item Name or Creator, you can also embed blanks at the beginning or elsewhere in the value.

Double-Byte Character Data

The EXPLODE command might need to be used (along with a graphics display terminal) to browse DBCS (Double-Byte Character Set) data or graphics data.

Note: For more information, see the “Double-Byte Character Set Support” appendix.

RC/Browse Options

The RC/Browse Option screen's main function is to control the amount of data retrieved for a browse session. A row limit and an SQL statement can be specified to control the number of rows returned. A test count can be requested before retrieving the rows to determine the impact of the selection. This screen can be used to control the initial display format (Column or Form) and the maximum display character size. All attributes can be saved as the default for the selected table to save time in future sessions.

Using the FB (fast browse) option, the Browse Options screen is bypassed. If the Browse Options screen appears when the FB options are entered, all options on the screen are ignored. The default options last saved for the table are used. If there are no default options, then RC/Browse's default options are used.

RC/Browse Options Screen Layout

The Browse Option's header is unprotected and can be changed to browse another table or perform another option. There is no need to return to the Main Menu to perform a different function. Selection criteria for the Item Name and Creator prompts can be input to retrieve the Table Selection screen (see previous section).

Note: The Data Selection fields are used to restrict the data retrieved from the selected table. For information about the fields, press F1 (Help).

Browse Table Data

You can browse table data by specifying browse options.

Follow these steps:

1. Press Enter on the RC/Browse Options screen.

   A browse screen appears.

2. Perform one of the following actions:

   - Browse data as needed.
   - Enter **MAIN** in the command line to return to the CA RC/Update Main Menu.
   - Press F3 to return to the RC/Browse Options screen.

3. To change to a specific table, change the Item Name or Creator prompts in the header on the CA RC/Update Main Menu.

   A new Browse Options screen appears for the new table. This permits users to view the default browse options for that table and make any changes before beginning the browse session.

   Browse options can be entered on the current Browse Options screen at the same time the new table name is entered. Because changes have been made to the browse options, RC/Browse uses the options currently displayed on the screen and immediately enters browse mode. The new Browse Options screen will not appear.

Note: If the FB option is entered on the CA RC/Update Main Menu, the Browse Options screen is immediately displayed.

RC/Browse Header

The RC/Browse screen header provides information and acts as a control center for the browse session. The control information can be changed at any time. The header is the same for both Column Mode and Form Mode.

More columns can be viewed by shrinking the character width. Use the EXPLODE option (see RC/Browse Command Reference) to view the whole column in ISPF Browse.

Note: For information about the fields, press F1 (Help).

Column Mode Browse

Column Mode is selected by entering a **C** for the Browse Mode prompt in RC/Browse's header or Browse Options screen. Column mode is used to browse multiple rows on the screen at one time.

You can control the display length of character data by taking either of the following actions:

- Adjust the value in the Max Char field in the header area. For example, enter a small value to view more columns on one screen.
- Issue the SETWIDTH command to request the preferred length.

Column Mode's header provides the same functions and information as Form Mode's header. The Column Mode Browse screen is designed to maximize row operations.

The screen displays information about each column.

The following table describes each part of the column information displayed for the second column, DESCRIPTION.

N

Null Indicator. If a field accepts null values, then an extra prompt (N) will appear next to the name. It indicates whether the field currently is set to Null (Y).

DESCRIPTION

The DB2 column name.

A2

Abbreviated name. If the TYPE command is on (see RC/Browse Command Reference), an abbreviated name appears under the DB2 name. (If a second line does not appear under the Field Names, then the TYPE toggle is not active.)

The abbreviated names begin with A1. The abbreviated names make it much easier to enter column names for other primary commands.

DB2 Data Type

DB2 data type line, toggled on and off with the TYPE command. The DB2 data type appears, along with the abbreviated name. This is handy for viewing the definition of character or decimal columns.

There are many primary browse commands that can be entered on the command line or assigned to PF keys for controlling the column display (for example, SORT and TYPE).

Form Mode Browse

Form Mode is selected by placing an **F** for the Browse Mode prompt in RC/Browse's header or Browse Options screen. Form mode is used to view one row per screen. The columns are displayed vertically down the left side, much like a form. This approach displays more columns on the screen than Column Mode.

You can control the display length of character data by taking either of the following actions:

- Adjust the value in the Max Char field in the header area. For example, enter a small value to view more columns on one screen.
- Issue the SETWIDTH command to request the preferred length.

Form Mode supports a maximum display length of 256 characters. If the entry exceeds 50 characters, an arrow indicates that additional text is present but not shown. You can expand the area by using the EXPAND command. Form Mode's header provides the same function and information as Column Mode's header.

There are many primary browse commands that can be entered on the command line or assigned to PF keys for controlling the display (for example, SORT and TYPE).

More information:

RC/Browse Header (see page 198)

Expand and View Data for Long Character Columns

You can display data for long character table columns by using the EXPLODE facility within RC/Browse. RC/Browse displays full screen mode, allowing you to see longer sections of data in one area.

Follow these steps:

1. Complete fields as follows on the CA RC/Update Main Menu:
   - Type **B** in the Option field.
   - Type **T** in the Object field.
   - Specify a table object name in the Item field.

     Note: You can also enter selection criteria to display a selection list.
   - (Optional) Complete the remaining fields as needed to refine your request.

   Press Enter.

   The RC/Browse Options panel appears.
2. Complete the fields on the panel as needed to determine options for your browse session, and press Enter.

   A column or form mode panel appears, listing rows of column data.
3. Type **EXPLODE** in the command line, move the cursor to the column whose data you want to expand (explode), and press Enter.

   Note: The EXPLODE command (along with a graphics display terminal) might be needed to browse DBCS (Double-Byte Character Set) data or graphics data.

   An ISPF session begins, showing the detailed column data.
4. Review the data as needed, and press the F3 (End) key when finished.

   The column or form mode panel reappears.

More information:

RC/Browse Command Reference (see page 200)

RC/Browse Command Reference

RC/Browse supports a full range of primary commands for controlling the display. These commands can be assigned to function keys or entered on the command line.

Note: RC/Browse is not an editing tool. Disregard references to the CANCEL, CAPS ON/CAPS OFF, CHANGE, RCHANGE, LCASE, UCASE, PROTECT, UNPROTECT, REFRESH, EDIT, SCOL, SET, SHOW, STATS, TRACE, and XCOL commands.

For a list of the commands available for use with RC/Browse, see the “Command Summary” appendix. Only those commands with Browse listed in the Mode field are usable in RC/Browse.

Chapter 10: RC/Copy

This section contains the following topics:

RC/Copy Component

RC/Copy lets you copy data between any two DB2 tables, even if their definitions do not match. Users can select the data to be copied and request that the target table be automatically created during the copy process. Users can also map unlike source and target columns and RC/Copy automatically performs all necessary data conversions. Virtually any type of data conversion is supported.

Using RC/Copy with RC/Edit, data can be easily copied from an existing table and edited as needed. The RC/Edit and RC/Copy facilities provide a powerful tool for creating test environments

Overview of RC/Copy

RC/Copy permits data copying between any two DB2 tables, even with different definitions. Using the RC/Copy feature in conjunction with RC/Edit, data can easily be copied from an existing table and then edited as needed.

The RC/Edit and RC/Copy features provide a powerful method for creating test systems. Some of the capabilities of the copy feature include:

- Selection of source and target tables from selection lists.
- Specify data to be copied by entering a SELECT statement (EQF data query). You can also specify the order of the data (in batch mode only) and the columns to be copied.

- Delete existing rows in the target table before the copy is performed. This option lets you refresh data in a table very easily.
- Explicitly map columns between the source and target tables. The mapping feature lets you copy data between unlike tables. Data conversion is performed automatically.

  Note: Column mapping can also be used with GENERATED columns, to let DB2 generate the value instead of RC/Copy. This must be done when the target table has a GENERATED ALWAYS column; otherwise, DB2 will prevent RC/Copy from updating the target table. A GENERATED ALWAYS column is either an identity column or a ROWID column.
- Automatically create the target table during the copy process. The database and tablespace to be used for the newly created table can also be specified.

  Note: RC/Copy does not create indexes when it creates the target table. In some cases, DB2 requires an index on a column. (In particular, an identity column or ROWID column which is GENERATED BY DEFAULT must have a UNIQUE index.) If the target table requires an index, you cannot use RC/Copy to create the target table.
- Include utility execution as part of the copy process. Utilities are to be run with each copy can also be specified.
- Copy data between two DB2 tables in online or batch mode.

  Note: RC/Copy cannot copy in batch mode, if the target table contains an identity column. It can perform such a copy in online mode.
- Copy data between DB2 tables on two different subsystems in batch mode.
- Reorganize the tablespace of the target copy by customizing the batch copy job stream to generate a batch Reorg utility.

Screen Flow for RC/Copy

RC/Copy is composed of the following screens:

Table Selection

Appears for selecting the source and target tables. EQF extended query screens can also be displayed.

Data Query

Appears for specifying the exact source data to copy (versus all data, all columns) to the target table.

Column Mapping

Appears if a request is made to map columns between the source and target tables. Users can explicitly define which source columns are copied to which target columns.

Utility Options

Appears if you elect to update Batch Utility Options. This screen allows specification of which utilities should be performed after the data has been copied.

Edit Mask Specification

Appears when a character column is mapped to a time or date column, at which point an edit mask must be entered that defines the data format within the character column.

Batch Copy Confirmation

Lets you accept or reject the generated batch input.

Tablespace Selection

Appears if a target database is selected without selecting a target tablespace on the Copy Options screen. The resulting tablespace selection list is based on the entered database name.

Database Selection

Appears if a target tablespace is selected without selecting a target database from the Copy Options screen. The resulting database selection list is based on the selected database name. This screen is also accessed if an asterisk is entered in both the Target Tablespace and Target Database fields on the Copy Options screen.

Copy Options

After entering the CO option, the RC/Copy Options screen appears so the user can define the copy options to enable for the copy operation.

The standard header, present on many of the screens, determines the source table for the copy operation. When the CO option is entered, the Item Name and Creator fields determine the source table for the copy operation. If selection criteria are entered in the Item Name or Creator fields, the table object Selection panel appears before the RC/Copy Options screen appears.

Note: For information about the fields, press F1 (Help).

Processing Considerations-Online Copies

Note the following processing considerations for online copies:

- To begin the copy process, press Enter.
- To cancel the process, enter the END command or press F3 (End). If the source table was selected from a selection list, the selection list appears (for selecting another table).
- To interrupt a copy in progress, press the ATTN key (or equivalent). The operation will be stopped and the option of continuing with the copy operation or canceling is provided.
- To return to the main menu, enter the MAIN command or press the appropriate function key.
- To copy another table, enter the changes and press Enter. If only the source table name is changed, the RC/Copy Options screen appears with the standard defaults. If information is changed on the RC/Copy Option screen and Enter is pressed, it is assumed the requested copy operation is to be performed using the entered options.

  Note: To copy data from a table in one subsystem to a table in another subsystem, batch mode must be used.
- To access a different option, enter the new option in the header. For example, E.
- When using EQF to specify source data selection criteria, ROW LIMIT and ORDER BY specifications will not be used. Batch mode must be used if ROW LIMIT or ORDER BY is required.

Column Mapping

There might be the need to map columns from the source table to columns in a target table that differ in name, type, or length. The RC/Copy Column Mapping feature permits mapping of source columns to target columns. All data conversions are automatically performed. For example, a column defined as CHAR can be copied to VCHAR.

There also may be a need to omit a column from the copy operation, so DB2 can generate a value for the target column. This is necessary if the column is GENERATED ALWAYS (and appears as A or I in the DF column). It is optional if the column is GENERATED BY DEFAULT (and appears as D or J in the DF column). A GENERATED column is either a ROWID column (indicated by values A and D in the DF column) or an identity column (indicated by values I and J in the DF column). You omit a column by setting its MAP# to blank, instead of specifying a numeric value.

The RC/Copy Column Mapping screen has two independent scrolling areas.

Note: For information about the fields, press F1 (Help).

When you leave the MAP# blank, DB2 supplies the values.

Data Conversion

The following chart summarizes the data type conversions allowed within the online facility. The vertical axis represents input. The horizontal axis represents output. Shaded areas indicate the conversion is online supported.

TO FROM	SML INT.	INT.	DEC.	FLT.	CHAR	V.C.	L.V.C.	GRAPH	V.G.	L.V.G.	DATE	TIME	T.S.	ROWID
SML INT.	x	x	x	x										
INT.	x	x	x	x										
DEC	x	x	x	x										
FLT.	x	x	x	x										
CHAR					x	x	x							
V.C.					x	x	x							
GRAPH.								x	x	x				
V.G.								x	x	x				
DATE EXTNL											x			
TIME EXTNL												x		
T.S. EXTNL											x	x	x	
ROWID														x

The following table describes the abbreviations used in the previous table:

Abbreviation	Name
SML. INT	Small Integer
INT.	Integer
DEC.	Decimal
FLT.	Float
V.C.	Varchar
L.V.C.	Long Varchar
GRAPH	Graphic
V.G.	Vargraphic
L.V.G.	Long Vargraphic
DATE EXTNL	Date External
TIME EXTNL	Time External
T.S.	Timestamp
T.S. EXTNL	Timestamp External
ROWID	Row ID

The DATE EXTNL, TIME EXTNL, and TIMESTAMP EXTERNAL types listed previously are assumed to be CHAR columns in date or time format. Also, for an online character conversion, the source column character length cannot be larger than the target column character length.

The following chart summarizes the data type conversions allowed within the Batch Copy facility. It accepts more data type conversions than the online facility. The vertical access represents input. The horizontal axis represents output. A shaded area indicates that the conversion is batch supported.

TO FROM	SML INT.	INT.	DEC.	FLT.	CHAR	V.C.	L.V.C.	GRAPH	V.G.	L.V.G.	DATE	TIME	T.S.	ROWID
SML INT.	x	x	x	x	x	x	x							
INT.	x	x	x	x	x	x	x			x	x			
DEC	x	x	x	x	x	x	x				x	x	x	

TO FROM	SML INT.	INT.	DEC.	FLT.	CHAR	V.C.	L.V.C.	GRAPH	V.G.	L.V.G.	DATE	TIME	T.S.	ROWID
FLT.	x	x	x	x	x	x	x							
CHAR	x	x	x	x	x	x	x				x	x	x	
V.C.	x	x	x	x	x	x	x							
LVC.	x	x	x	x	x	x	x							
GRAPH.								x	x	x				
V.G.								x	x	x				
L.V.G.								x	x	x				
DATE EXTNL			x	x	x	x	x				x		x	
TIME EXTNL			x	x	x	x	x					x	x	
T.S. EXTNL					x	x	x				x	x	x	
ROWID														x

The following table describes the abbreviations used in the previous table:

Abbreviation	Name
SML. INT	Small Integer
INT.	Integer
DEC.	Decimal
FLT.	Float
V.C.	Varchar
L.V.C.	Long Varchar
GRAPH	Graphic
V.G.	Vargraphic
L.V.G.	Long Vargraphic
DATE EXTNL	Date External
TIME EXTNL	Time External

Abbreviation	Name
T.S.	Timestamp
T.S. EXTNL	Timestamp External
ROWID	Row ID

Mapping Commands

The AUTOMAP and RESET commands are provided to help map source columns to target columns. These commands should be typed in command line.

AUTOMAP Command

Enter the AUTOMAP command to have RC/Copy perform default column mapping. This can save time when there is the need to change only a few column mappings. RC/Copy makes the default mappings based on name, type, and length (in that order). To specify a different order of precedence, use the N, T, and L parameters.

N

Specifies to AUTOMAP only if the column names are equal.

T

Specifies to AUTOMAP only if the column types are equal.

L

Specifies to AUTOMAP only if the column lengths are equal.

The default is NTL. Specify any combination of these codes to control automapping.

The command syntax is:

```
AUTOMAP [N|T|L]
```

RESET Command

Clears the current column mapping so the process can be restarted. Use this command with automapping to experiment with different mapping techniques.

The command syntax is:

```
RESET
```

Processing Considerations for Column Mapping

Note the following processing considerations:

- To map columns, press F3 (End) to begin processing.
- To cancel processing, enter the CANCEL command or press the appropriate PF key.
- To scroll source columns, press the F7 (Up) and or F8 (Down) keys.
- To scroll target columns, press the F10 (Left) and F11 (Right) keys.

Note: If the decision is made to automatically create the target table during the copy process, the table creation in-progress screen will appear before the column mapping screen because the target table must be created before columns can be mapped.

Batch Copy Facility

The Batch Copy facility lets you copy data between any two DB2 tables in batch mode and automates how the data is copied.

For example, when using Batch Copy, the statements to unload the data from the source table to the target table are generated, the load utility is executed, and the target table can be created automatically. The control statements for the copy are stored in a user-specified batch data set. Once this batch data set is closed and unallocated, it can be submitted to the CA Batch Processor for execution.

Use of the Batch Copy facility provides the following benefits:

- Familiarity—Because the Batch Copy facility is an extension of RC/Copy, users that are familiar with RC/Copy will be able to quickly use the Batch Copy facility.
- Increased Resources—If data needs to be copied between enormously large tables, resources can be consumed at an exhaustive rate during peak hours. With the Batch Copy facility, you can write the statements needed to copy the data to a batch data set. This data set can be submitted during off-peak hours to increase resource availability.
- Row Limitations—You can specify an exact number or rows to be copied, which is useful when only a few rows are needed from a large table.
- Multiple Subsystem Support—Table selection is expanded because you can copy data from multiple DB2 subsystem IDs (SSID) within a location. Previously, tables could only be copied within the same DB2 subsystem ID (SSID).

- Automatic Utility Execution—You can select which utilities should automatically be executed after the data has been copied. For example, the PDASTATS option of CA Database Analyzer or RUNSTATS can be executed to reflect the new data statistics.
- Data Conversions—Support is provided for a wider variety of data type conversions than online copy.
- Row Order—ORDER BY criteria specified if EQF is supported. Data sorted in the target tables clustering order could eliminate the need for a reorg.

Access

When the CO (COPY) option code is specified from the header field, RC/Copy is invoked. The Batch Copy facility is invoked through the Mode field on the RC/Copy Options screen.

When **B** is entered in the Mode field on the RC/Copy Options screen, the Batch Specifications screen appears, where the data set to contain the statements necessary to copy the data between the tables is specified.

After specifying the data set, press Enter. The RC/Copy Options screen reappears. A message indicates that the specified data set has been allocated.

More information:

Copy Options (see page 205)

Utility Options Screen

The Utility Options screen appears when **Y** is entered in the Batch Utility Options field from the RC/Copy Options screen. The Utility Options screen allows specification of the utilities to be performed after the data has been copied.

Note: This screen is valid only when the copy is executed through the Batch Copy facility.

The Data Unload Options determine which options should be used during the unload and load of the data to/from the tables.

Note: For information about the fields, press F1 (Help).

Updating ISPF Profile

From the Utility Options screen, the entered option values can be saved to the user's ISPF Profile. Or, the entered option values can be used for only the current copy execution. The following describes option values use.

- SAVE—Enter SAVE in the command line to save the option values displayed from the Utility Options screen to the user's ISPF Profile. The saved values are used for subsequent batch copies until they are changed from the Utility Options screen. The SAVE command is useful for quickly updating a Profile with the most frequently used values.

  Note: After the values are saved, press F3 (End) to proceed.

- END—Press F3 (End) to save the options for the current batch copy only. The ISPF Profile will not be updated. Depending on the AutoMap Column value entered on the RC/Copy Options screen, the Batch Copy Confirmation screen or the Column Mapping screen will appear.

  Note: If no Target Table Name or Target Table Creator values have been specified, the RC/Copy Options screen reappears when F3 (End) is pressed. After entering these values, press Enter to proceed.

Edit Masks

When mapping a CHAR, VARCHAR, or LONG VARCHAR column to a DATE, TIME, or TIMESTAMP data type, an edit mask must be entered that defines the data format within the character column. After pressing Enter to verify column mapping selections, an edit mask specification screen appears so an edit mask can be specified if necessary.

Note: For information about the fields, press F1 (Help).

Press Enter after specifying an edit mask, and press F3 (End) to return to the Column Mapping screen. Press F3 (End) again to proceed. The Batch Copy Confirmation screen appears.

Processing Considerations for the Batch Copy Facility

The Utility Options and Column Mapping screens can be exited by pressing F3 (End).

- Utility Options Screen. If N was entered in the AutoMap field, the Column Mapping screen appears. If Y was entered in the AutoMap field, the Batch Copy Confirmation screen appears.
- Column Mapping Screen. Upon exit, the Batch Copy Confirmation screen appears.

The Batch Copy Confirmation screen lets you accept or reject the generated batch input. Press Enter, and the generated Batch Processor input is written to the user-specified batch data set; or press F3 (End) to cancel and return to the RC/Copy Options screen. If you select cancel, the Batch Processor input is not written to the batch data set.

If Enter is pressed from the Batch Copy Confirmation screen, a message indicates that the Batch Processor statements have been written to the batch data set. Continue by closing and unallocating the data set, specifying more target tables, or having other batch input statements written to the batch data set.

- Close and Unallocate the Data set. If the user only wants the data from the source table to be copied to the target table, the batch data set should be simply closed and unallocated.
- Multiple Copies in a Single Job. Data from one source table can be copied to multiple target tables, or multiple source tables can be copied to multiple target tables.

  For example, data is copied from one source table to a target table. The user decides that the same source table should be used for another batch copy, but the Row Limit field should be changed for the second target table. The user would simply enter the name of the next target table in the Target Table Name and the Target Table Creator fields and specify information in any of the other Batch Copy facility fields.

  The statements to copy the data for all specified target tables will be written to the batch data set. After all target tables have been included, close and unallocate the data set.
- Other Batch Input Statements. Users are not limited to the Batch Copy facility when the batch data set is opened and allocated. For example, the batch input needed to create a tablespace can be written to batch data set. The necessary batch input statements can continue being generated and written to the batch data set until it is closed and unallocated it.

Close and Unallocate the Data Set

An indication must be made as to when the statements to copy the data from the source table to any target tables should stop being written to the batch data set.

Follow these steps:

1. Enter **O** in the Mode field from the RC/Copy Options screen.

   Note: When switching back to online mode, the Online Specification screen appears. From this screen, select the option to close and unallocate the data set.

2. Submit the batch data set to the CA Batch Processor when convenient.

Chapter 11: Data Compare

This section contains the following topics:

Data Compare Component

The Data Compare component lets application developers and database administrators create and maintain data compare strategies to compare data between tables or between the same tables at two different times. These strategies can be used repeatedly, updated, or templated.

The Data Compare facility also provides screens for selecting table pairs to be compared, specifying the column order for processing, mapping the columns between the table pairs, generating the unload control statements, and executing the compare. The Data Compare Report shows the differences between the two tables.

Overview of Data Compare

The Data Compare component enables application developers and database administrators to compare data between tables, or between the same table at two different times. This capability can be used in the following situations.

- Quality Assurance. Use Data Compare to make sure a new program being tested makes the expected changes by comparing the table before and after the program is run.
- Data Synchronization Check. Use Data Compare to make sure that test data matches production data.
- Selective Column Comparison. Use Data Compare to compare certain columns of two tables, without worrying about other columns (such as timestamps) that might not match.

Important! Do *not* use DB2 catalog tables in an data compare strategy. CA RC/Update cannot determine whether a specified table is a catalog table or non-catalog table, which can cause complications during processing of the data comparison.

Using the Data Compare component involves several steps. These steps can be done sequentially at one time or can be spread over a period of time.

Data Compare Strategy Creation Steps

Each of the steps listed for creating and using a Data Compare Strategy is described in detail in the sections that follow.

- Access the Data Compare facility. Use the DC option on the Main Menu to enter the Data Compare Strategy Services screen.
- Create a Data Compare strategy. On the Data Compare Strategy Services screen, define the parameters and strategy name. Users can create a new strategy or update or template an existing strategy.
- Select table pairs to be compared. On the Data Compare Table Mapping screen, each source table is mapped to its target table. Multiple table pairs can be included in a strategy.
- Specify a column order to be used by both tables. On the Data Compare Explode Services panel, a Data Query is specified to define the order in which data will be processed. The number of rows of data that will be compared can also be limited.

- Map the columns between the table pairs. On the Data Compare Column Mapping screen, for each table pair defined, map the columns from the source table to the target table. Automapping is available for columns with the same name in both tables.
- Generate unload control statements and unload the table data for each table. The Data Compare Unload Generation, Data Compare Unload Specification, and Data Compare Execution specification screens allow users to control the unloading of the data from the DB2 tables to data sets for the compare process.
- Run the compare function. From the Data Compare Explode Services panel, issuing the C (Compare) line command initiates the compare on the unload data sets.
- Examine the comparison results. The Data Compare Report shows the differences between the two tables being compared.

Access the Data Compare Facility

To access the Data Compare facility, enter **DC** in the Main Menu Option field.

The Data Compare Strategy Services screen appears. Use this screen to create, delete, update, copy, or execute data compare strategies.

Create a New Data Compare Strategy

Creating a new data compare strategy involves the following:

- Data Compare Strategy Services screen (see page 219)
- Creating a new strategy (see page 220)
- Data set listing (see page 220)
- Strategy and data set options (see page 222)
- Specifying the target subsystem (see page 223)

Data Compare Strategy Services Screen

A strategy defines the DB2 tables whose data should be compared. The Data Compare Strategy Services screen allows users to create, delete, update, copy, or execute strategies. It appears when DC is entered in the Option field of the header.

Note: For information about the fields, press F1 (Help).

After all information has been entered in the Strategy Specifications fields, press Enter to display the list of selected strategies. Only those strategies for which the user has authorization are listed. Strategy specifications can be changed at any time to change the strategies listed.

Data Set Listing

Indented under the strategy name are the names of the unload data sets related to that strategy. (The data sets only contain data after the unload has been performed.) Unload data set names are displayed only when the List option is toggled on. They are displayed in highlighted text, so they are easy to see. To use the list option, enter **L** at the command line, or use the L (List) line command, described in the following subsection.

Create a New Strategy

To create a new Data Compare strategy, entries must be made in the fields as described in this section.

Follow these steps:

1. Make entries in the following fields:

   O (Options)

   Specifies the option next to the appropriate strategy or data set name. Enter **C** to create a new strategy. This option is valid only on the first line.

   STRATEGY

   Specifies the strategy name. Eight-character maximum.

   DESCRIPTION

   Specifies the description of the strategy (25-character maximum).

   CREATOR

   Specifies the User ID of the creator of the strategy. The current ID is automatically inserted when you are creating a new strategy. The strategy name and creator together uniquely identify the strategy.

   TP (Type)

   Specifies the type of strategy. Type **D**. A Data Compare strategy is the only kind of strategy that can be created at this time.

SO (Share Option)

Designates the share option. This is the authorization within CA RC/Update for others to use or update the strategy.

Y

Specifies that other users can use, but not update, your strategy.

U

Specifies that others can use and update the strategy.

N

Specifies that others cannot view or use the strategy. (Strategies created by other users that were saved with a share option of N will not be displayed.)

SRC SSID

Specifies the source DB2 subsystem ID. This is the DB2 subsystem that contains the DB2 objects you want to compare. The source SSID can be different from the subsystem that contains the strategy, and from the target SSID. The source SSID must be entered at definition time. Enter **?** to access the DB2 SSID/Location selection panel.

USER, DATE, AND TIME

Indicates the user who last updated the strategy, and the date and time the strategy was created or last updated. Nothing needs to be entered in these fields when creating a new strategy.

2. Press Enter to continue creating a strategy.

   The Create Data Compare Strategy screen appears.

3. Press F3 (End) to return to the Main Menu.

More information:

Map Source Tables to Target Tables (see page 224)
Strategy and Data Set Options (see page 222)
Specify the Source and Target Subsystems (see page 223)

Strategy and Data Set Options

This section lists and describes the options that are valid in the O (Options) field of the Data Compare Strategy Services screen. Enter these options next to strategy names.

A (Analysis)

Allows users to unload the data (optional) and run the data comparison. It displays the Data Compare Execution Specification screen, where the strategy can be submitted for execution.

B

Allows users to unload table data. It displays the Data Compare Unload Specification screen, where unload statements that have already been generated can be submitted.

C

Creates a new strategy. This command is valid only on the first line. Complete the other processing fields before creating a strategy.

D

Deletes an existing strategy.

L

Lists the data sets associated with this strategy. This command acts as a toggle.

T

Templates an existing strategy to create a new one.

U

Updates an existing strategy.

Enter these options next to data set names.

B

Browses the unload data set.

D

Disconnects the data set-strategy connection. This does not delete the data set, but it severs the data set from the strategy.

E

Edits the unload data set.

S

Selects the data set to be used in the comparison.

More information:

Comparing Table Data (see page 233)
Data Set Listing (see page 220)
Unloading and Comparing Table Data (see page 231)

Specify the Source and Target Subsystems

The next step in creating a Data Compare strategy is to specify the target subsystem that contains the target table to be compared. Once the fields for a new strategy have been entered on the Data Compare Strategy Services screen and Enter has been pressed, the Create Data Compare Strategy screen appears.

The strategy name, creator, description, and share option are displayed in the header section of the screen. All fields in the header, except the creator, are updatable. Below the header are the fields that specify the source and target subsystems.

Follow these steps:

1. Enter the following information for both the source and the target:

   SSID

   Specifies the target SSID. It is updatable (the source SSID field is not).

   Location

   Specifies the subsystem location. The default is LOCAL.

   Obj Name and Creator

   Displays the table name and creator. Select table pairs from the tables that meet the requirements specified here.

   Where

   Specifies **Y** or **S** to access the Extended Query Facility (EQF). The default is N.

   Note: For more information about EQF, see the *CA Database Management Solutions for DB2 for z/OS General Facilities Reference Guide*.

2. Press Enter to continue the creation process.

   The Data Compare Table Mapping screen appears. Press F3 (End) to return to the Data Compare Strategy Services screen. The strategy will not be saved.

Map Source Tables to Target Tables

On the Data Compare Table Mapping panel, you can map source tables to target tables. By mapping tables, table pairs are selected for the strategy.

Note: You can use CA RC/Query reports to display specialized information about your tables (for example, a list all objects dependent on a table). To display the available CA RC/Query report options, enter **?** in the OPT field or Map # field.

Follow these steps:

1. Define a strategy (see page 220) by completing the fields of the Data Compare Strategy Services panel.

   The Create Data Compare Strategy panel appears.

2. Specify the source and target subsystems (see page 223) on the Create Data Compare Strategy panel.

   Note: You can also update an existing strategy to add pairs. To do so, enter **U** next to the strategy on the Data Compare Strategy Services panel, then enter **I** in the command line on the Data Compare Explode Services panel.

   The Data Compare Table Mapping panel appears.

3. Create one or more pairs by typing a source table number in the MAP# field next to a target table (to map a pair together), and then press Enter when all pairs are set up.

   Note: Press F7 (Up) and F8 (Down) to scroll the source table list. Press F10 (Left) and F11 (Right) to scroll the target table list.

   Mapped target tables are added to the Mapped Object Queue and are no longer displayed.

4. (Optional) Enter **S** in the command line to view the queued object.

   The Mapped Object Queue panel appears.

5. (Optional) Enter **S** in the command line to exit the Mapped Object Queue panel.

   The Data Compare Table Mapping panel appears.

6. Press the END key when table pair selections are complete.

   The Data Compare Explode Services panel appears, where you can further update the data compare strategy as needed.

More Information:

Data Compare Table Mapping Screen (see page 225)
Create a New Strategy (see page 220)
Specify the Source and Target Subsystems (see page 223)

Data Compare Table Mapping Screen

The Data Compare Table Mapping screen is where source tables are mapped to target tables. The tables specified by the Object Name and Creator fields on the Create Data Compare Strategy screen are displayed on the Data Compare Table Mapping screen. Once the fields for a new strategy have been entered on the Create Data Compare Strategy screen and Enter has been pressed, the Data Compare Table Mapping screen appears.

The strategy name, creator, description, and share option are displayed in the header section of the screen. All fields but creator can be modified. Below these fields, the source and target header fields permit changes to be made to the list of source and target tables:

REFRESH

Enter **Y** to refresh the list of target options.

SOURCE SSID INFORMATION

Enter the source subsystem. These fields function exactly like those on the Create Data Compare Strategy screen. The table name and creator can be changed to list additional objects.

TARGET SSID INFORMATION

Enter the target subsystem. These fields function exactly like those on the Create Data Compare Strategy screen. The table name and creator can be changed to list additional objects.

Data compare operations on global temporary tables are not supported. Explicit references to Global Temporary Table objects within a data compare strategy yield empty table results by the definition of a Global Temporary Table.

More information:

Map Source Tables to Target Tables (see page 224)

Using the Data Compare Explode Services Panel

The Data Compare Explode Services panel appears after table pairs have been specified in a new strategy or when a request is made to update an existing strategy from the Data Compare Strategy Services screen. The Data Compare Explode Services panel serves as a control center for updating and fine-tuning an existing data compare strategy. Users can perform the following tasks.

- Specify column ordering information via a data query in the Where field.
- Map columns for all table pairs automatically with the MAP primary command.

- Initiate selective column mapping for an individual table pair by displaying the Data Compare Column Mapping screen.
- Initiate updating the data set names currently attached to the data compare strategy by displaying the Data Compare Dataset Specification screen.
- Initiate the generation of unload control statements (used to unload the data from the tables) by displaying the Data Compare Unload Generation screen.
- Compare the data between two tables by generating the Data Compare Report with the C (Compare) command.
- Insert new table pairs into the strategy.
- Delete table pairs from the strategy.

The strategy name, creator, description, and share option are displayed at the top of the screen. All fields in this section except the creator ID are updateable. The middle portion of the screen provides information about the source and target subsystem IDs and locations. These fields are not updatable. The lower portion of the screen provides information about each of the table pairs.

Note: For information about the fields, press F1 (Help).

Specifying Column Ordering

After table pairs have been selected, column ordering information must be specified. This information is used to help ensure that the data comparisons are between corresponding rows in each table. Column ordering is initiated on the Data Compare Explode Services panel. The following precautions help ensure that the Data Compare facility is comparing the correct rows and columns.

- Use the same ordering strategy for the source and target columns. In other words, if the source table is ordered by column 1, then the target table should be ordered by the column that is mapped to source column 1.
- Use a unique ordering strategy if possible. Otherwise, the compare routine might get out of sync when comparing rows.

Data Query Edit Screen

The Data Query Edit screen allows specification of the column order of a table. This affects the sorting when the table unload is performed, which affects the order of the comparisons, which in turn affects the comparison results

To display the Data Query Edit screen from the Data Compare Explode Services panel, first enter **Y** in the one-character part of the Where field for the table to be ordered. Then leave the eight-character part of the Where field blank and press Enter. The Data Query Edit screen appears with a Query Name of Temp.

In the ORD field, enter the order for the table. Use the scrolling keys F7 (Up) and F8 (Down) to display more columns. In the previous example, column 2 is first, column 3 is second, and column 4 is third. When the ordering specification has been completed, press F3. A (for ascending) is inserted after the numbers entered in the ORD field and you are returned to the Data Compare Explode Services panel. The appropriate ORDER BY statement specified by column ordering appears on the data Compare Unload Specification screen when the unload control statements are generated (see Generating Unload Control Statements).

To view or change the column sort order for the other table in the pair, use the Where field to display the Data Query Edit screen again, and change the order as described previously. In general, if a data query is used for one table, it should be used for both tables to guarantee compatible sort order during the table unload.

After column ordering has been specified and the number of rows to be compared has been limited (optional), the table columns can be mapped.

Note: For more information, see the *CA Database Management Solutions for DB2 for z/OS General Facilities Reference Guide*.

More information:

Mapping Columns Between Tables (see page 227)

Mapping Columns Between Tables

Mapping columns between tables involves the following:

- Map command (see page 227)
- Data Compare Column Mapping screen (see page 228)
- Specifying the unload data sets (see page 229)

Map Command

Mapping columns between the source and target tables must start from the Data Compare Explode Services panel.

Use one of the following techniques to map the columns.

- Enter the MAP command at the command line. Columns for all table pairs will be mapped. If columns were already mapped for a table pair, the mappings will be displayed.
- Enter **M** in the line command area next to the table pair for which columns are to be mapped and press Enter. The Data Compare Column Mapping screen appears.

Data Compare Column Mapping Screen

Column mapping in the Data Compare facility works the same way as in the RC/Copy facility.

Note: For information about the fields, press F1 (Help).

Mapping Manually

To map the columns manually, enter the source column number in the Map # field of the target column.

Mapping Automatically

Two commands, AUTOMAP and RESET, are provided to help map the source columns to the target columns. Type a mapping command at the command line and press Enter.

Enter **AUTOMAP** to perform default column mapping. This can save time when only a few of the column mappings need to be changed. Data Compare makes default mappings based on name, type, and length (in that order). For a different order of precedence, use the N, T, and L parameters after the AUTOMAP command. The default is NTL. Specify any combination of these codes to control auto-mapping.

N

Specifies to AUTOMAP only if the column names are equal.

T

Specifies to AUTOMAP only if the column types are equal.

L

Specifies to AUTOMAP only if the column lengths are equal.

AUTOMAP can be abbreviated as AM.

Enter **RESET** to clear the current column mapping so the process can be restarted. Use this command with automapping to experiment with different mapping techniques. The command can be abbreviated as RES.

To begin automapping, enter the command and press Enter.

To cancel processing, enter the CANCEL command or press the appropriate function key.

To Exit the Data Compare Column Mapping Screen, press the F3 (End) key to return to the Data Compare Explode Services panel. The MAP column updates to YES for the mapped tables.

Specifying the Unload Data Sets

Before unloading data from source or target tables, specify which data sets will be used as destinations for unloading the tables. Otherwise, Data Compare uses data sets with names generated in the following manner.

How Unload Data Set Names Are Generated

The unload data set names are generated by Data Compare if not specified. Data Compare uses Model Services for generation. The format depends on the model used to generate the data sets.

To choose user-specified names for the unload data sets, enter the high-level and secondary qualifiers, without quotes. Data Compare adds .DATA to the end of the data set name. For example, USER5.TABLE1 would generate the data set name USER5.TABLE1.DATA.

Note: If the data set names (generated by either method) are the same, Data Compare makes them unique by adding a time stamp. Data is not overwritten.

View or Update Unload Data Set Names

To view or update the data set names currently attached to the data compare strategy, issue the **E** (Edit) line command next to the table pair on the Data Compare Explode Services panel.

After entering **E** and pressing Enter, the Data Compare Dataset Specification screen appears.

The strategy name, creator, description, and share option are displayed. None of these fields can be changed. The creators and table names of both the source and target tables for the current data compare strategy are also displayed. These fields cannot be changed on this screen.

The following fields can be filled in or changed on the screen.

SOURCE DATASET

Enter the data set that holds the unloaded data for the source table.

TARGET DATASET

Enter the data set that holds the unloaded data for the target table.

The source data set name and target data set name are the only fields that can be changed. After changing these data set names, enter **SAVE** on the command line to link the revised names to the Data Compare strategy. Then press F3 (End) to return to the Data Compare Explode Services panel; or press F3 (End) to return to the Data Compare Explode Services panel without making changes to the data set names.

Generating Unload Control Statements

The next step is to generate the unload control statements. These statements are used to unload data from the tables. Users can also specify the names of the data sets that are used as destinations for unloading the tables. Otherwise, Data Compare uses data sets with names generated in the manner described in Specifying the Unload Data Sets.

Specifying Unload Data

To generate the unload control statements, enter **U** in the Sel (Selection) field next to the selected table pair on the Data Compare Explode Services panel and press Enter.

After entering the Unload Data line command, the Data Compare Unload Generation screen displays.

Generate Statements to Unload Data

The Data Compare Unload Generation screen controls the generation of the unload statements, which unload data from the source or target tables.

Notice that there are no unload control statements yet for the source or target table.

Note: For information about the fields, press F1 (Help).

To generate statements to unload data from the source or target tables, complete the fields on the Data Compare Unload Generation screen as follows and press Enter.

- Specify **Y** in the Generate field.
- Specify the table(s) for which the unload statements are to be generated in the Tables field.

The unload statements for the specified tables will be displayed in the middle and bottom of the data display area, respectively.

Each set of unload control statements can be scrolled independently: use F7 (Up) and F8 (Down) for the source table unload statements, and use F10 (Left) and F11 (Right) for the target table unload statements. Press F3 (End) to save the statements and return to the Data Compare Explode Services panel. Press F3 again to save the strategy and return to the Data Compare Strategy Services screen. After the unload statements have been generated, the data can be unloaded.

Options for Unloading Table Data

Before comparing table data, it must be unloaded from the tables. If unsure whether data has been unloaded, check the unloaded data, using the technique described in the Viewing Unloaded Data.

To unload the data, select a Compare Strategy with the A or the B option on the Data Compare Strategy Services screen. Option A allows the user to unload data and proceed directly into the data comparison. Option B only allows the user to unload the data. Option A is discussed in Unloading and Comparing Table Data.

Data Compare Unload Specification Screen

Option **B** on the Data Compare Strategy Services screen displays the Data Compare Unload Specification screen.

On the Data Compare Unload Specification screen, specify whether the statements are being created for unloading the source, target, or both tables.

Note: For information about the fields, press F1 (Help).

Unloading the Data

Once the unload process has been started, the Data Compare Confirmation screen appears. It displays the unload statements. Scroll through the commands using F7 (Up) and F8 (Down). From the Data Compare Confirmation screen, F3 (End) cancels the unload and displays the previous screen. Press Enter to unload the data. After the data has been unloaded, the Audit Message File displays return codes and unload messages. Press F3 (End) to return to the Unload Specification screen.

Unloading and Comparing Table Data

Option **A** on the Data Compare Strategy Services screen displays the Data Compare Execution Specification screen. This option unloads the data (optional), runs the compare, and produces the report.

On the Data Compare Execution Specification screen, specify whether the statements are being created for unloading the tables and performing the Data Compare or only for performing the Data Compare.

Note: For information about the fields, press F1 (Help).

After **S** is entered in the Execute field, press Enter to compare the data. The Online Data Compare Confirmation screen appears, displaying the commands to be executed. Press Enter to execute the commands or press F3 (End) to return to the previous screen. The job can be submitted for execution at a later point. Press F3 (End) from the Data Compare Execution Specification screen to return to the Data Compare Strategy Services screen.

Batch Specification Screen

The Batch Specification screen appears if the mode is B (Batch) when data unload is started from either the Data Compare Unload Specification screen or Data Compare Execution Specification screen.

Note: For information about the fields, press F1 (Help). For more information about batch mode, see the “Using CA RC/Update” chapter.

Viewing Unloaded Data

Before comparing the data, there may be the need to confirm that the data has been unloaded, or a user might want to browse or edit it. These functions can be performed from the Data Compare Strategy Services screen.

View the unload data sets associated with a data compare strategy by using the L (List) line command. The L command can also be entered in the primary command line to list the unload data sets associated with all the strategies. When this command is used, the TYPE field appears, displaying the unload method (CA Fast Unload or UNLOAD utility) that was used to unload the data set. A question mark in the TYPE field signifies that the unload method used will be determined when the comparison is run.

To confirm that the data has been unloaded, users can browse or edit the data sets in question. Enter **B** to browse a data set, or **E** to edit it. An ISPF-like editor appears, which can be used to edit or browse the unloaded data. Press F3 (End) to return to the Data Compare Strategy Services screen.

Selecting Unloaded Data to Compare

If more than one unload data set has been generated for the source or target table, the data sets to be compared must be specified. This can be done on the Data Compare Strategy Services screen.

Comparing Table Data

There are two methods for comparing the data. The first, described in Unloading and Comparing Table Data, is to use the **A** (Analyze) option from the Data Compare Strategy Services screen. The second, described in this section, is to use the **C** (Compare) command from the Data Compare Explode Services panel.

After the data has been unloaded for both the source and target tables, data can be compared on the Data Compare Explode Services panel. Issue the **C** (Compare) line command next to the table pair.

After entering **C** and pressing Enter, a wait message indicates that the comparison is in progress. Once the comparison has been completed, the Data Compare Report appears (see Reading the Data Compare Report).

Reading the Data Compare Report

The Data Compare Report displays all the differences between the two sets of data being compared. The following sample of an online report shows the changes that result from comparing the source table PDJIMB.EMP3 and the target table PDJIMB.EMP4. The labels for the types of changes are highlighted.

```
RUDCRPT        --------- RC/Update Data Compare Report --------
COMMAND ===>                                                SCROLL ===> PAGE

SSID: D81A ------------------------------------------------------ USER3     >

SOURCE DATASET: PDJIMB.EMP3
TARGET DATASET: PDJIMB.EMP4

TYPE ORD COLUMN             DATA
PAIRED INSERT/DELETE
SRC     EMPNO               000020
        FIRSTNME            RONNIE
        MIDINT
        LASTNAME            VANZANT
        WORKDEPT            B01
        PHONENO             3476

TRG     EMPNO               000020
        FIRSTNME           =>BILLY
        MIDINT
        LASTNAME           =>POWELL
        WORKDEPT            B01
        PHONENO             3476
-------------------------------------------------------------------------------
NON-PAIRED DELETE
SRC     EMPNO               000020
        FIRSTNME            MICHAEL
        MIDINT              L
        LASTNAME            COOK
        WORKDEPT            B01
        PHONENO             3476

-------------------------------------------------------------------------------
NON-PAIRED INSERT
TRG     EMPNO               000001
        FIRSTNME            JIM
        MIDINT              Q
        LASTNAME            SHORTS
        WORKDEPT            AAA
        PHONENO             0666

RC/UPDATE DATA COMPARE STATISTICS SUMMARY
   NON-PAIRED INSERTS:            1
   NON-PAIRED DELETES:            1
   PAIRED INSERTS/DELETES:        1
******************************** BOTTOM OF DATA ********************************
```

Viewing the Report

Use the F7 and F8 keys to move through the report. If the columns extend beyond the right hand side of the screen, use the F10 and F11 keys to move horizontally in the report. When finished viewing the report, press F3 (End) to return to the Data Compare Explode Options screen.

Printing the Data Compare Report

Use the PPRINT, QPRINT, PFILE, or QFILE commands to print the Data Compare report or save it to a data set.

Note: For more information about these commands, see the *CA Database Management Solutions for DB2 for z/OS General Facilities Reference Guide*.

Data Compare attempts to recognize updated rows and marks them as CHANGED. Due to the nature of the algorithm being used, updated rows may get posted as INSERT/DELETE pairs.

The following table describes what updates to the target table look like:

Source	Key	Data	Source	Key	Data
aaaa	xxxxxxxxxx	Not Applicable	aaaa	xxxxxxxxxx	Not Applicable
bbbb	xxxxxxxxxx	Not Applicable	aaaz	xxxxxxxxxx	Not Applicable
cccc	xxxxxxxxxx	Not Applicable	bbbb	@@@@@@	Not Applicable
dddd	xxxxxxxxxx	Not Applicable	cccc	xxxxxxxxxx	Not Applicable
Not Applicable	Not Applicable	Not Applicable	dddd	xxxxxxxxxx	Not Applicable

A row with aaaz has been inserted and the row with key bbbb has had its data updated. Data Compare will say that two rows have been inserted (keys of aaaz and bbbb) and one row has been deleted (key of bbbb).

Chapter 12: Log Display

This section contains the following topics:

Overview of the Log Display Facility

Each DDL statement executed through the CA Batch Processor is logged, as well as every edit or browse session, in a history file. The Log Display Facility enables users to review the history log.

DB2 does not record when an object was dropped or altered in the catalog. The Log Display Facility provides an audit trail of changes not available from DB2. The log can be invaluable in tracking changes to DB2 objects.

With the Log Display panel, users can choose to view any portion of the history log. The panel initially defaults to show the actions performed during the current day. Selection criteria can be changed to view the listings of any command executed by any user at any time.

Access a Log Display Panel by entering L (Log) in the Option Field on any screen. An example Log Display is shown in the next two sections.

Note: The Log Maintenance feature is available as option W on the CA Database Management Solutions for DB2 for z/OS main menu. This feature allows the user to display, backup, purge, or restore log records. This feature does not affect the way Log Display works within the product; instead, it gives the user additional control over the Log records, accessible from outside the product. For more information about Log Maintenance, see the *CA Database Management Solutions for DB2 for z/OS General Facilities Reference Guide*.

Controlling the Log Display

The log display is controlled by changing the selection criteria entered in the header or by using the Extended Query Facility (EQF).

Note: For information about the fields, press F1 (Help).

Log Display

The Log Display lists the activity from the log history file according to the criteria entered in the header and through EQF. It is a display-only area of the panel where all information is protected from entry. Scroll the display using scrolling commands.

Note: For information about the fields, press F1 (Help).

Press F3 (End) to return to the Main Menu.

Chapter 13: Object Definition Defaults

This section contains the following topics:

Masking and ID Groups

You can define default definitions for individual users or for groups of users. Defaults can be assigned for groups of IDs using an ID mask. A mask is a valid TSO ID followed by an asterisk (*).

When the DEF command is entered, a search for an object definition defaults entry for the ID is performed. If there is an entry for the ID, it is used. If there is not an entry, we look for the mask that most closely matches the ID. An asterisk by itself represents *all* IDs. A specific ID always takes precedence over a masked ID.

Required Authorization

Use the Product Authorization Facility (described in the *General Facilities User Guide*) to define which users can view and change object definition defaults for which IDs. These privileges are shared between CA RC/Migrator and CA RC/Update.

Define Defaults for Attributes Used during Object Creation

Defaults that you specify for DB2 object attributes are used as the initial attribute values when an object is created, templated, or altered.

Follow these steps:

1. Do one of the following on the command line:
   - From CA RC/Migrator, enter **DEF**.
   - From CA RC/Update, enter **DEFAULTS**.

     Press Enter. The Object Definition Defaults screen appears.

   Note: Defaults for the current user or the mask ID that matches most closely are displayed.
2. In the Userid field, specify the TSO ID of the user for whom object definition defaults are being defined. Enter a percent sign (%) to display a selection list.
3. Specify a description in the Description field of the ID or the mask. This can be up to thirty characters.
4. In the Action field, enter the action to perform on the ID specified in the Userid field. Valid values are:

   R

   Retrieves the object definition defaults.

   S

   Saves the object definition entries.

   D

   Deletes the object definition entries.

   Note: Some users have authority to view and change object definition defaults for all IDs. Others can view and change object definition defaults only for their own IDs. Still others can only view object definition defaults for their own IDs.

Press Enter to process the Action field value.

5. Review and update the default values as needed for the object attribute in the Default Value field. Note the following fields:

   Object Type

   Specifies the type of DB2 object to which the attributes apply. Global defaults apply to all DB2 object types.

   Object Attribute

   Specifies the name of the object attribute.

   Valid Entry Information

   Displays information (for some attributes) to help specify a valid value.

6. Enter **S** in the Action field (once the necessary changes have been made).

   The entries are saved.

Object Type Defaults

The following defaults can be set on the Object Definition Defaults screen:

- Global defaults (see page 242)
- Storage group defaults (see page 244)
- Database defaults (see page 244)
- Tablespace defaults (see page 245)
- Table defaults
- Sequence defaults
- Index defaults
- View defaults (see page 256)
- Alias defaults (see page 257)
- Trigger defaults (see page 257)

Global Defaults

Global Defaults apply to all DB2 object types to which the attribute pertains. However, if an attribute default is also specified for the specific object type, that value is used. Enter a value in the Default Value field to the right of the appropriate attribute name to set the default as needed.

The following defaults let you establish default values for a specific object attribute across all of the object types to which the attribute pertains.

Creator

Specify the default creator to use for a newly created object.

(blank)

Uses the current user ID as the default creator for all object types. This is the default.

userid

Uses the user ID you enter in this field as the default creator for all object types.

SQLID

Uses the current SQLID in effect at creation time as the default creator for an object you create.

If you enter an explicit entry in the Creator field for any object type on this screen, it overrides the Global Defaults value for that type.

Note: The remaining values listed in this section also apply for individual object types, except that explicit (SQLID or non-blank) individual object type values override the Global Defaults value.

VCAT NAME

VSAM catalog. Enter selection criteria to display an object selection list.

STOGROUP

Storage Group. Enter selection criteria to display an object selection list.

PRIMARY QTY

Primary Allocation for storage groups in kilobytes.

SECONDARY QTY

Secondary Allocation for storage groups in kilobytes.

FREEPAGE

Number of pages loaded before a page is left as free space (0—255).

PCTFREE

Percentage of each page left as free space (PCTFREE) (0—99).

BUFFER POOL

Buffer pool. BP0, BP1, BP2, BP32K.

DATABASE

Database. Enter selection criteria to display an object selection list.

PARTITIONED

Indicates whether the tablespace or index will be partitioned. YES or NO.

CLOSE

Indicates whether the VSAM data sets allocated to the tablespace are closed when the tablespace is not in use (resource consideration). YES or NO.

CCSID

(Coded Character Set Identifier) Specify the encoding scheme for the object. Valid values are:

Ascii

Specifies that the data must be encoded using the ASCII CCSIDs specified during installation.

Ebcdic

Specifies that the data must be encoded using the EBCDIC CCSIDs specified during installation.

Default

Specifies that the CCSID argument is to be omitted from the generated DDL. When creating an object, the default encoding scheme that is used depends on the object type and how it is created. The rules used in determining which encoding scheme to use are decided by DB2.

Note: For more information about default encoding schemes, see the IBM DB2 *SQL Reference* or *Administrator* guides.

When altering an object, the encoding scheme of the object is changed to the current default encoding scheme if it is different than the encoding scheme currently used by the object.

Unicode

Specifies that the data must be encoded using the Unicode CCSIDs specified during installation. This value is only supported by DB2 versions 7.1 and later.

blanks

Specifies the same meaning as Default.

Storage Group Defaults

The following defaults can be specified for storage groups.

CREATOR

Authorization ID of the owner of the storage group.

VCATNAME

Name of the VSAM catalog used to track the DASD volumes defined by the storage group. Enter selection criteria to display an object selection list.

VOLUME1 - VOLUME6

VOLSERs for space allocation. Enter selection criteria to display an object selection list.

Database Defaults

The following defaults can be specified for databases.

Note: For more information about these fields, see the “Database” chapter.

CREATOR

Authorization ID of the owner of the database.

STOGROUP

The database's storage group. All table and indexspaces will use this storage group by default unless they specify their own. Enter selection criteria to display an object selection list.

BUFFER POOL

The database's buffer pool. All table and indexspaces will use this buffer pool by default unless they specify their own.

CCSID

(Coded Character Set Identifier) Specifies the CCSID encoding scheme.

Tablespace Defaults

The following defaults can be specified for tablespaces.

Note: For more information about these fields, see the “Tablespace” chapter.

CREATOR

Primary authorization ID of the owner of the tablespace.

DATABASE

Name of the database that contains the tablespace.

VCAT NAME

The name of the VSAM catalog used to track the DASD volumes defined by the tablespace. Each partition can reference a different VSAM catalog.

STOGROUP

Partition's storage group. This can be different for each partition.

PRIMARY QTY

Primary allocation in kilobytes.

SECONDARY QTY

Secondary allocation in kilobytes.

FREEPAGE

Number of pages loaded before a page is left as free space.

PCTFREE

Percentage of each page left as free space.

PARTITIONED

Denotes whether the tablespace is partitioned.

BUFFER POOL

The tablespace's buffer pool.

CLOSE

Denotes if the VSAM data sets allocated to the tablespace are closed when the tablespace is not in use (resource consideration).

SEGSIZE

The number of pages in each segment of the segmented tablespace (must be a multiple of 4 from 0 to 64). Zero if tablespace is not segmented.

COMPRESS

Indicates whether ESA compression is used on the data rows for this tablespace or this partition of the tablespace.

LOCKSIZE

Lock size of the tablespace. Determines how DB2 locks the pages in the tablespace.

LOCKMAX

The maximum number of page or row locks that an application can hold in the tablespace before locks are escalated from the level indicated in the LOCKSIZE field.

DEFINE

Indicates whether to defer the physical allocation of DB2-managed data sets.

TS TYPE

Indicates the type of tablespace.

CCSID

(Coded Character Set Identifier) Specifies the CCSID encoding scheme.

Table Defaults

The following defaults can be specified for tables.

Note: For more information about table fields, see the “Table” chapter.

CREATOR

Specifies the primary authorization ID of the owner of the table.

DATABASE

Specifies the table's database.

TABLESPACE

Specifies the table's tablespace.

AUDIT

Determines which value is specified for the AUDIT clause of the CREATE TABLE statement. The AUDIT value determines the types of access to this table that will cause auditing to be performed.

COLUMN DATA TYPE

Specifies the data type for columns.

COLUMN NULL INDICATOR

Specifies the value used for the NULL clause of the CREATE TABLE and ALTER TABLE statements.

COLUMN DEFAULT IND.

Specifies the value used for the DEFAULT clause of the CREATE TABLE and ALTER TABLE statements. The possible entries for this field depend on the value entered (Y or N) for the Column Null Indicator.

DATA CAPTURE

Indicates whether SQL changes to the table should be written to the Log.

Table Type

Specifies the type of table.

CCSID

Specifies the Coded Character Set Identifier (CCSID) encoding scheme.

Volatile

Specifies whether DB2 uses index access (Y) for SQL operations on this table or SQL access (N) based on the current statistics. If Y, the VOLATILE clause is generated and inserted into the DDL. If N, the NOT VOLATILE clause is generated and inserted into the DDL if necessary.

APPEND

(DB2 9 and later) Specifies how data rows are stored in a table during INSERT and LOAD operations:

YES

Appends the rows at the end of the table and ignores clustering during INSERT and LOAD operations.

NO

Does not use append processing for the table. This is the default.

Sequence Defaults

The following defaults can be specified for sequences:

Schema

Specifies the default schema name (qualifier) to be used when creating an explicitly qualified sequence.

Note: The name *cannot* begin with the character string *SYS* (unless it is *SYSADM*).

Default: Blanks

As Type

Specifies the default data type to be used for the sequence. The data type can be any exact numeric data type (SMALLINT, INTEGER, BIGINT, or DECIMAL), or a user-defined distinct type sourced on any on these types with a scale of zero.

Default: INTEGER

Note: If you do not specify a value, the corresponding field on the Sequence Create panel will default to an asterisk (not INTEGER).

Type Schema

Specifies the default type schema name (qualifier) of the data type to be used for the sequence when data type is a distinct type.

Default: SYSIBM

Note: If you do not specify a value, the corresponding field on the Sequence Create panel will default to an asterisk (not SYSIBM).

Precision

Specifies the default precision for the sequence. Precision depends on the data type of the sequence. You can enter a specific value for Precision only if the specified data type is one of the following:

- A user-defined type (from which a defined precision can be picked up)

  Note: When the sequence's data type is SMALLINT, INTEGER, BIGINT, or a distinct type that is sourced on SMALLINT, BIGINT, INTEGER, or DECIMAL, you cannot edit the Precision field. Instead, the field displays a precision that is inherent to (or defined for) the specified type.

- DECIMAL

If the data type is not user-defined or DECIMAL, Precision is set to 10 (for an INTEGER type) or 5 (for a SMALLINT type).

Default: 10

START WITH

Specifies the default start value for the sequence. The value can be any positive or negative value that could be assigned to a column of a data type (without non-zero digits existing to the right of the decimal point).

Default: Blanks

INCREMENT

Specifies the default increment interval between consecutive values of the sequence. The value can be any positive or negative value (or 0) within the scope of an INTEGER that could be assigned to a column of a data type (without any non-zero digits existing to the right of the decimal point).

Default: Blanks

MINVALUE

Specifies the default minimum value at which one of the following occurs:

- A descending sequence cycles or stops generating values.
- An ascending sequence cycles after encountering the maximum value (or the NO keyword).

Default: Blanks

MAXVALUE

Specifies the default maximum value at which one of the following occurs:

- An ascending sequence cycles or stops generating values.
- A descending sequence cycles after encountering the minimum value (or the NO keyword).

Note: In CA RC/Update, this field does not allow the NO keyword when the DB2 version is earlier than V8.

Default: Blanks

CYCLE

Specifies whether the sequence should continue generating values after reaching its maximum value or minimum value. The boundary of the sequence can be reached with the next value landing exactly on the boundary condition or by overshooting the boundary. Valid values are YES, NO, and blanks.

Default: NO

CACHE

Specifies one of the following:

- Maximum number of values of the sequence that DB2 can preallocate and keep in memory
- NO

Default: 20

ORDER

Specifies whether the sequence numbers must be generated in order of request. Valid values are YES, NO, and blanks.

Note: In CA RC/Update, this field does not appear when the DB2 version is earlier than V8.

Default: NO

Procedure Defaults

The following defaults can be specified for stored procedures:

Schema

Specifies the SQL short identifier used to explicitly qualify the stored procedure that you are altering, creating, or templating.

Language

Specifies the language interface convention to which the procedure body is written.

Fenced

Specifies whether the procedure runs in an external address space.

DBINFO

Specifies whether additional status information is passed to the stored procedure when it is invoked.

Result Sets

Specifies the maximum number of query result sets that the stored procedure can return.

Debug Mode

Specifies whether the JAVA procedure (or this version of the native SQL procedure) can be run in debugging mode.

Parm CCSID

Specifies the encoding scheme for the PARAMETER clause for string (character and graphic data type) parameters.

Parm Style

Specifies the linkage convention used to pass parameters to and from the stored procedure.

Deterministic

Specifies whether the stored procedure returns the same results each time the stored procedure is called with the same IN and INOUT arguments.

SQL Access

Specifies the data access classification level of SQL statements, if any, executed in the procedure or any routine that is called from the procedure.

Program Type

Specifies whether the stored procedure runs as a main routine or a subroutine.

COLLID

Specifies the package collection that is to be used when the stored procedure is executed. (This is the package collection into which the DBRM that is associated with the stored procedure is bound.)

Stay Resident

Specifies whether the stored procedure load module is to remain resident in memory when the stored procedure ends.

WLM Environment:

Specifies the WLM (workload manager) environment in which the stored procedure is to run when the DB2 stored procedure address space is WLM-established:

name

Specifies the WLM environment in which the stored procedure must run. If another stored procedure or a user-defined function calls the stored procedure, and that calling routine is running in an address space that is not associated with the specified WLM environment, DB2 routes the stored procedure request to a different address space.

*****

Customizes the WLM environment specification as follows: Supplying a name (SQL qualifier) in the first field and setting the second field to asterisk (*) specifies the WLM environment in which a stored procedure runs when an SQL application program directly calls the stored procedure. If another stored procedure or a user-defined function calls the stored procedure, the stored procedure runs in the same WLM environment that the calling routine uses.

Stop After Failures

Specifies whether the procedure is to be put in a stopped state after some number of failures.

Security

Specifies how the stored procedure interacts with an external security product to control access to non-SQL resources.

Commit On Return

Specifies whether DB2 commits the transaction immediately on return from the stored procedure.

ASUTIME

Specifies the total amount of processor time (in CPU service units) that a single invocation of the stored procedure can run (unrelated to the ASUTIME column of the resource limit specification table).

Special Registers

Specifies how special registers are set by DB2 on entry to the routine.

Package Path

Specifies the package path to use when the procedure is run. This is the list of the possible package collections into which the DBRM associated with the procedure is bound.

Run Options

Specifies the Language Environment run-time options to be used for the stored procedure (or, for a REXX procedure, the options to be passed to the REXX language interface to DB2).

PARAMETER DECLARATION:

Specifies the number of parameters of the stored procedure, the data type of each parameter, and the name of each parameter.

Note: The parameter names are optional when the procedure is not an SQL procedure.

Type

Identifies the type of parameter as input, output, or both.

Schema

Is the high-level qualifier of the data type.

CCSID

Is the encoding scheme for a string (character and graphic data type) parameter.

ASL

Specifies whether a locator to the value of the parameter is passed to the procedure instead of the actual value.

LANGUAGE C: Parm VARCHAR

Specifies, for the PARAMETER clause for when Language is C, the representation of variable-length character string parameters.

LANGUAGE JAVA: JAR Schema

Specifies the part of the external-java-routine-name that identifies the schema of the jar-id given to the JAR when it was installed in the database.

SQL-NATIVE: Qualifier

Specifies the implicit qualifier that is used for unqualified names of tables, views, indexes, and aliases that are referenced in the procedure body.

SQL-NATIVE: Active

Specifies whether this version of the stored procedure is the current active version.

SQL-NATIVE: Degree

Specifies whether to attempt to run a query using parallel processing to maximize performance.

SQL-NATIVE: Application CCSID

Specifies the default encoding scheme for SQL variables in static SQL statements in the procedure body.

SQL-NATIVE: Validate

Specifies whether to recheck, at run time, errors of the type "OBJECT not FOUND" and "NOT AUTHORIZED" that are found during bind or rebind.

SQL-NATIVE: Package Owner

Specifies the owner of the package that is associated with the first version of the procedure.

SQL-NATIVE: WLM Debug Environment

Specifies the WLM (Work Load Manager) application ENVIRONMENT FOR DEBUG MODE that is used by DB2 when debugging the native SQL procedure.

SQL-NATIVE: OPTHINT

Specifies whether query optimization hints are used for static SQL statements that are contained within the body of the procedure.

SQL-NATIVE: SQL Path

Specifies the SQL path that DB2 uses to resolve unqualified user-defined distinct type, function, and procedure names in the procedure body.

SQL-NATIVE: Prepare

Specifies whether to defer preparation of dynamic SQL statements that refer to remote objects, or to prepare them immediately.

SQL-NATIVE: REOPT

Specifies whether DB2 will determine the access path at run time by using the values of SQL variables or SQL parameters, parameter markers, and special registers.

SQL-NATIVE: Current Data

Specifies whether to require data currency for read-only and ambiguous cursors when the isolation level of cursor stability is in effect. Current Data also determines whether block fetch can be used for distributed, ambiguous cursors.

SQL-NATIVE: Explain

Specifies whether information will be provided about how SQL statements in the procedure will execute.

SQL-NATIVE: Dynamic Rules

Specifies the values that apply, at run time, for the following dynamic SQL attributes:

- Authorization ID that is used to check authorization
- Qualifier that is used for unqualified objects
- Source for application programming options that DB2 uses to parse and semantically verify dynamic SQL statements

SQL-NATIVE: Immediate Write

Specifies whether immediate writes are to be done for updates that are made to group buffer pool dependent page sets or partitions.

SQL-NATIVE: Isolation Level

Specifies how far to isolate the procedure from the effects of other running applications.

SQL-NATIVE: Keep Dynamic

Specifies whether DB2 keeps dynamic SQL statements after commit points.

SQL-NATIVE: Release At

Specifies when to release resources that the procedure uses, either at each commit point or when the procedure terminates.

SQL-NATIVE: Rounding

Specifies the desired rounding mode for manipulation of DECFLOAT data.

SQL-NATIVE: Decimal

Specifies the maximum precision to use for decimal arithmetic operations.

SQL-NATIVE: Date Format

Specifies the date format for result values that are string representations of date or time values.

SQL-NATIVE: Time Format

Specifies the time format for result values that are string representations of date or time values.

SQL-NATIVE: FOR UPDATE CLAUSE

Specifies whether the FOR UPDATE clause is required for a DECLARE CURSOR statement if the cursor is to be used to perform positioned updates.

Index Defaults

The following defaults can be specified for indexes.

Note: For more information about these fields, see the “Index” chapter.

CREATOR

Specifies the primary authorization ID of the index (and indexspace) owner.

TABLE CREATOR

Specifies the creator of the table on which the index is based.

VCAT NAME

Specifies the name of the VSAM catalog under which you user defined data sets have been created.

STOGROUP

Specifies the storage group for index (if not partitioned) or partition (if partitioned). This can be different for each partition.

PRIMARY QTY

Specifies the primary allocation in kilobytes.

SECONDARY QTY

Specifies the secondary allocation in kilobytes.

FREEPAGE

Specifies the number of pages loaded before a page is left as free space.

PCTFREE

Specifies the percentage of each subpage or nonleaf page that is left as free space.

UNIQUE

Specifies whether the UNIQUE keyword is included in the CREATE INDEX statement.

CLUSTER

Specifies whether the CLUSTER keyword is included in the CREATE INDEX statement.

PARTITIONED

Specifies whether the index is partitioned.

BUFFER POOL

Specifies the buffer pool for the index.

CLOSE

Specifies whether the VSAM data sets allocated to the index are closed when the index is not in use (resource consideration).

TYPE

Specifies the index type.

DEFINE

Specifies whether to defer the physical allocation of DB2-managed data sets.

PADDED

Specifies whether varying-length string columns that are stored in the index will be padded to their maximum length (YES or NO) or have the process controlled by the installation default specified by the PAD INDEXES BY DEFAULT field on the installation panel DSNTIPE (DEFAULT).

COMPRESS

Specifies whether to use compression for index data (YES or NO).

DEFER

Secifies whether the index is built during the execution of the CREATE INDEX statement (YES or NO).

View Defaults

The following defaults can be specified for views.

Note: For more information, see the “View” chapter.

CREATOR

Creator of the view.

WITH CHECK OPTION

Specifies if the CHECK OPTION is specified in the CREATE VIEW statement

Alias Defaults

The following defaults can be specified for aliases.

Note: For more information about these fields, see the “Alias” chapter.

CREATOR

Authorization ID of the owner of the alias.

TABLE LOCATION

Location of the table on which the alias is based.

Trigger Defaults

The following defaults can be specified for triggers.

SCHEMA

Specifies the SQL short identifier used to explicitly qualify the trigger.

TB OWNER

Specifies the owner of the table the trigger is associated with.

WHEN

Specifies whether the trigger is a before trigger or an after trigger. Valid values are Before and After. The default is After.

EVENT

Specifies the specific event that activates the trigger. Valid values are Insert, Delete, and Update. The default is Insert.

FOR EACH

Specifies whether DB2 executes the triggered action for each row of the subject table or executes the triggered action once for the triggering SQL operation. Valid values are Statement and Row. The default is Statement.

MODE

Specifies the mode of the trigger. DB2 supports only mode DB2SQL.

OLD AS

Specifies the correlation name that identifies the state of the row prior to the triggering SQL operation.

NEW AS

Specifies the correlation name that identifies the state of the row as modified by the triggering SQL operation and by any SET statement in a BEFORE trigger already executed.

OLD TABLE AS

Specifies the table name of a temporary table that identifies the state of the complete set of modified rows by the triggering SQL operation prior to any actual changes.

NEW TABLE AS

Specifies the table name of a temporary table that identifies the state of the complete set of rows as modified by the triggering SQL operation and by any SET statement in a BEFORE trigger already executed.

Chapter 14: RC/Alter

This section contains the following topics:

Overview of RC/Alter

RC/Alter allows you to perform the following types of alters that are not possible through the DB2 ALTER commands:

- Alters where changes require dropping and re-creating objects and their dependencies
- Alters where changes are too complex and require help generating the DDL (using a fuller and more extensive analysis)

Note: RC/Alter is also used to perform certain drop requests that require a fuller and more extensive analysis.

Optionally, you may request that RC/Alter be used to create, alter, and drop objects that do not otherwise require extensive analyses. To make this request, set the CA RC/Update Operation Mode to A or respond with A on the DDL Execution Confirmation, DDL Execution Warning, or Implicit Drop Warning pop-up screens.

RC/Alter simply requires that the user indicate the alterations to be made to the object. CA RC/Update automatically determines whether DB2 or RC/Alter is the most efficient way to make the alteration. For specific information regarding which changes require RC/Alter, see Alter Requirements.

The Alteration Analysis screen serves as the RC/Alter main screen, and allows users to specify the execution options for the alteration and to name the data set that will contain the commands necessary to perform the alteration. The analysis can be performed online or as a batch job. From the Alteration Analysis screen, you can specify the following options:

- Fast Submit. The Submit Dataset option allows smooth access to the Batch Processor Interface screen, which activates the Batch Processor. To use this Alteration Analysis option, simply enter Y in the Submit Dataset prompt to submit the specified data set to the Batch Processor for execution.
- Alteration Recovery. The Recovery option allows information to be saved that can be used to recover from an object alteration. This includes dropping the new version of the object, its data and any dependents, and recreating the original version of the object, along with its data and dependents. The recovery information is saved in a recovery data set that is actually an alteration analysis file. The Recovery option is accessed from the Alteration Analysis screen.
- Execution Mode. RC/Alter can perform the analysis online, in batch mode, or in RC/Alter mode. If the analysis is performed online, the commands are written to a specified data set. If batch mode is specified, a Strategy Batch Analysis screen appears to permit information entry for the batch job. This allows an alteration to be submitted at a later time and permits processing during down time.

Three screens branch from the Alteration Analysis screen:

- Analysis Options screen
- Recovery Options screen
- Batch JCL Specification screen (if batch mode is specified)

The Analysis Options screen allows unload options and utility parameters to be defined for the alteration data set. The Recovery Options screen allows the user to name the data set to contain the commands necessary to restore the altered object to its original state, as if the alteration never happened. The unload options and utility parameters for this recovery data set can be specified as well. The Batch JCL Specification screen allows entry of information necessary for a batch job.

Alteration Strategy

RC/Alter writes all the SQL and Batch Processor commands necessary to perform the requested alteration to a data set executed through the Batch Processor. This alteration data set contains commands that accomplish the following tasks.

Important! Do *not* use DB2 catalog tables in an alteration strategy. CA RC/Update cannot determine whether a specified table is a catalog table or non-catalog table, which can cause complications during processing of the alteration.

- Unload data from any tables to be dropped as part of the RC/Alter process. The size of each unload data set is dynamically calculated at execution time by the unload program.
- Create the recovery alteration strategy if the Recovery option was selected.
- Drop the object to be altered. This also causes DB2 to drop all dependent objects, including authorizations.
- Re-create the object with the new attributes.
- Generate the necessary DDL to re-create all dependent objects that were dropped when the altered object was dropped. Any attribute changes made to the altered object are automatically propagated to all lower level objects. The necessary referential integrity statements are also generated.
- Generate the necessary DDL to re-create all authorizations for the dropped objects. RC/Alter rebuilds authorization chains that were created as a result of the WITH GRANT option.

  The PBP AUTH command is invoked to dynamically change to the appropriate GRANTOR before the GRANT statement is issued. Security exits are provided for controlling who can execute the AUTH statement.

  Note: For more information about the AUTH command, see the *CA Database Management Solutions for DB2 for z/OS Batch Processor Reference Guide*.
- Execute the DB2 load program to load all data into the newly re-created tables. Any necessary data conversions are automatically performed.

  Note: For more information about data conversions, see the *CA Database Management Solutions for DB2 for z/OS Batch Processor Reference Guide*.
- Execute the necessary DB2 utilities to update statistics, check pending flags, and rebind affected plans.

The RC/Alter design permits the separation of the request for an alteration from the actual approval and implementation of the alteration. This separation of tasks fits the internal requirements of many companies. To make the alteration effective, the generated data set is executed using the BP (Batch Processor) option.

More information:

Analysis Output (see page 284)
Recovery Option (see page 262)

Recovery Option

The Recovery option enables the recovery of the original version of an object altered with RC/Alter. This feature is useful when the new version of the object does not meet specific needs and you want to return to the previous version. The Recovery option is also useful when part of an alteration analysis has been executed.

If the Recovery option is specified, RC/Alter creates an alteration strategy that has three parts. Part of a sample alteration strategy analysis data set appears in Alteration Analysis Screen.

If the Recovery option is set to Y (ON) and data is being saved, enough space is needed for two full sets of data: the object alteration unload data set and the recovery unload data set. If concerned about the usage of additional space by the recovery unload data set, a user might want to send the recovery unload data set to tape. To do so, modify the user-defined UNIT symbolic in the model ID.

Alteration Analysis Screen

The CA RC/Update Alteration Analysis screen appears whenever RC/Alter must be used to make a change that requires dropping and re-creating objects.

Note: When alterations can be made by DB2 without dropping and re-creating the object, the CA RC/Update Alter Confirmation screen appears with the DDL required to make your changes. The DDL may then be immediately executed online.

Note: For information about the fields, press F1 (Help).

RC/Alter Strategy Options

The RC/Alter Strategy Options allow submission of multiple alteration analysis jobs simultaneously in batch mode. Users can assign a unique name to each alteration strategy, or use the same name for all strategies and have a unique SEQUENCE NUMBER assigned to each strategy. For example, RCUBAT00, RCUBAT01, RCUBAT02, and so on. The SEQUENCE NUMBER is then automatically incremented after an RC/Alter analysis is submitted for batch processing. You can use the RC/Alter Strategy Options to:

- Change the prefix of the strategy name (the first one to seven characters) to use a value other than the default RCUBAT.
- Control the number of unique strategy names to be maintained through the MAX SEQUENCE option.
- Choose alphabetic or decimal incrementing of the SEQUENCE NUMBER through the INCREMENT STYLE option.

The RC/Alter Strategy Options are applicable only when performing an alteration analysis in batch mode. When an analysis is performed in online mode, the RC/Alter Strategy Options are not used. However, RC/Alter still validates the information in the fields and therefore they must contain valid data or values (unless INCREMENT STYLE is set to N, in which case the SEQUENCE NUMBER and MAX SEQUENCE options are not validated).

Important! When an analysis is submitted in batch mode and a previously submitted batch job has not yet completed, the current job may use the same OUTPUT DATASET member name as the previous job. This is because the member names are generated based on the last used member name in the specified output data set. This can cause the DDL generated from a previous analysis to be overwritten with DDL generated from subsequent analyses.

To avoid this problem, review the member name (in the Dataset Name field on the Alteration Analysis screen) when submitting multiple batch jobs simultaneously, and edit the member name to make it unique and/or use a different data set name if necessary.

The RC/Alter Strategy Options include:

STRATEGY NAME

Enter the name of the batch strategy. Valid characters are A through Z, 0 through 9, @, #, and $. STRATEGY NAME must not contain embedded blanks and may not begin with a numeric character. The initial setting for this field is RCUBAT.

Note: STRATEGY NAME does not apply to and is therefore not used in online analyses. It is used for batch analyses only.

The maximum allowable length for STRATEGY NAME is dependent on the INCREMENT STYLE chosen:

- When INCREMENT STYLE is D, STRATEGY NAME may contain one to six characters.
- When INCREMENT STYLE is A, STRATEGY NAME may contain one to seven characters.
- When INCREMENT STYLE is N (sequencing is disabled), STRATEGY NAME can contain up to eight characters.

If a blank name is entered and INCREMENT STYLE is set to D or A, STRATEGY NAME will default to RCUBAT; otherwise, it will default to RCUBATCH.

If the STRATEGY NAME is changed, the values in the other RC/Alter Strategy Options fields (SEQUENCE NUMBER, MAX SEQUENCE, and INCREMENT STYLE) will not be affected.

SEQUENCE NUMBER

Enter the SEQUENCE NUMBER for this strategy. SEQUENCE NUMBER is appended to the STRATEGY NAME for the current RC/Alter analysis. The initial setting for this field is 00.

This number is automatically incremented for subsequent RC/Alter analyses and is used to identify a strategy from other strategies with the same name (for example, RCUBAT00, RCUBAT01, RCUBAT01). The value entered must be less than or equal to the value specified for MAX SEQUENCE.

Valid characters and the number of characters required for SEQUENCE NUMBER are dependent on the INCREMENT STYLE chosen, as described in the following table:

Increment Style	Valid Characters for Sequence Number
A (alphabetic)	A through Z (one character allowed)
D (decimal)	0 through 9 (two digits required)
N (none)	(not applicable)

If blanks are entered, the SEQUENCE NUMBER will default to 00 (when INCREMENT STYLE is D) or A (when INCREMENT STYLE is A).

When the SEQUENCE NUMBER reaches the value specified as the MAX SEQUENCE, it is automatically reset to 00 (when INCREMENT STYLE is D) or A (when INCREMENT STYLE is A) for the next RC/Alter analysis.

Note: This option is applicable only for Batch analysis mode.

MAX SEQUENCE

Enter the maximum value for the SEQUENCE NUMBER. This option allows users to limit the number of maintained RC/Alter strategy names. The initial setting for this field is 10. Valid characters and the number of characters allowed for MAX SEQUENCE are dependent on the INCREMENT STYLE chosen, as described in the following table:

Increment Style	Valid Characters for Max Sequence
A (alphabetic)	A through Z (one character allowed)
D (decimal)	0 through 9 (one digit required)
N (none)	(not applicable)

When SEQUENCE NUMBER reaches the value specified as MAX SEQUENCE, SEQUENCE NUMBER is reset to 00 (when INCREMENT STYLE is D) or A (when INCREMENT style is A) for the next RC/Alter analysis.

If this field is blank, the value will default to the maximum value supported for the specified INCREMENT STYLE, as described in the following table:

Increment Style	Maximum supported value
A (alphabetic)	Z
D (decimal)	99
N (none)	(not applicable)

Note: This option is applicable only for Batch analysis mode.

INCREMENT STYLE

Specify the format for the SEQUENCE NUMBER increments. The initial setting is N (none).

A

Increments the SEQUENCE NUMBER of the RC/Alter STRATEGY NAME using alphabetic sequencing, from A to Z. This allows a maximum of twenty-six unique strategy names.

D

Increments the SEQUENCE NUMBER of the RC/Alter STRATEGY NAME using numeric sequencing, from 00 to 99. This option allows a maximum of 100 unique strategy names.

N

Disables automatic sequencing of the STRATEGY NAME. No sequence number will be appended to the strategy name. When automatic sequencing is disabled, up to eight characters for the STRATEGY NAME can be specified.

If the INCREMENT STYLE is changed from N to A or D, and SEQUENCE NUMBER and/or MAX SEQUENCE contain values that are not valid for the new INCREMENT STYLE chosen, both SEQUENCE NUMBER and MAX SEQUENCE are reset according to the new INCREMENT STYLE, as follows.

- If the new style is A, SEQUENCE NUMBER and MAX SEQUENCE are set to A and Z, respectively.
- If the new style is D, SEQUENCE NUMBER and MAX SEQUENCE are set to 00 and 99, respectively.

Note: This option is applicable only for Batch analysis mode.

Execution Mode

The execution mode involves the following:

- Online analysis (see page 267)
- Batch analysis (see page 267)
- Job statement specification (see page 267)
- Processing the batch JCL specifications (see page 267)
- Analysis JCL (see page 269)
- Return codes (see page 270)

Online Analysis

If **O** is entered in the Execution Mode field, the Alteration Analysis Wait screen appears.

To stop the analysis process, press the ATTN key or the equivalent keystrokes on your system. By contrast, under the Candle CL/SUPERSESSION product, the user must press ATTN, followed by RESET, followed by PA1. If an analysis is interrupted, the analysis data set is affected as follows.

- For PDS, the member is unchanged from its previous version.
- For sequential data sets, an error message is written to the data set.

During analysis, analysis messages are written to a reviewable file. This file is specified as MSGFILE in the analysis JCL. The DDL necessary to complete the alteration is written to the specified data set. For online execution, after analysis is complete, the analysis message file is automatically displayed so any analysis messages can be reviewed.

When finished reviewing the message file, press F3 and the data set containing the DDL appears. The DDL can be reviewed and edited at this point. When the data set is exited, the Analysis Options screen reappears.

Batch Analysis

If the **B** option is selected, the Batch JCL Specification screen appears.

The Batch Execution Specifications field is where the batch job destination is input.

Note: For information about the fields, press F1 (Help).

Job Statement Specification

A valid job statement must be entered.

Processing the Batch JCL Specifications

Once all information has been entered, including a valid destination, press Enter.

- If selection criteria was entered in the Output Dataset or Model JCL Member fields, a Member Selection list appears. Select a member and press F3 (End) to return to the Batch JCL Specification screen.
- If Y was entered in the Edit Model JCL field, the input data set appears in an Edit session. Press F3 (End) to return to the Batch JCL Specification screen.
- If Y was entered in the Update Values field, the Model JCL Substitution appears.
- If J was entered for the destination, the job is submitted directly to JES.

- If D was entered for the destination, the job is written to the specified data set.
- If P was entered for the destination, the JCL is previewed in a Browse session. Press F3 (End) to return to the Batch JCL Specification screen.

A message appears, indicating whether the job was successfully sent to the specified destination. Press F3 (End) to return to the Batch JCL Specification screen.

Analysis JCL

Following is an example of the JCL created during a batch analysis. This is the JCL that is written to a data set the user will execute to generate the commands necessary to complete the migration, alteration or comparison.

```
 EDIT ----- MODEL.JCL ---------------------------------------- COLUMNS 001 072
 COMMAND ===>                                                 SCROLL ===> 000
 ****** ***************************** TOP OF DATA ******************************
        //USER01   JOB  (11020,230),'D.SMITH',CLASS=A,
        // MSGCLASS=Z
        /*JOBPARM ROOM 21
        //*
[1]     //ANALYZE  EXEC PGM=PTLDRIVM,REGION=4M,PARM='EP=RML@MAIN'
        //STEPLIB  DD   DISP=SHR,DSN=high-level.CDBALOAD
        //         DD   DISP=SHR,DSN=D81A.PRIVATE.DSNEXIT
        //         DD   DISP=SHR,DSN=DSN.V810.SDSNLOAD
        //PTILIB   DD   DISP=SHR,DSN=high-level.CDBALOAD
        //         DD   DISP=SHR,DSN=D81A.PRIVATE.DSNEXIT
        //         DD   DISP=SHR,DSN=DSN.V810.SDSNLOAD
        //PTIPARM  DD   DISP=SHR,DSN=PTIPROD.RB708AA.CDBAPARM
        //MSGFILE  DD   SYSOUT=*
        //REPFILE  DD   SYSOUT=*
        //ABNLIGNR DD   DUMMY                SUPPRESS ABENDAID DUMPS
        //DDLFILE  DD   DISP=SHR,DSN=USER02.KAT.TEST(RCUA0002)
        //PARMFILE DD   *
[2]      CREATOR  USER3
         STRATEGY RCUPDATE
         STRTSSID D81A
         UNLOAD
         HEADER
         TRAILER
         REPINDDL
         DELTEMP
         DELUNLD
         RECOVIX
         REORGTS
         REORGIX
         PREFIX   USER3
         MODEL4   @DEFAULT
         MODEL4C  *
        /*
 ****** ************************** BOTTOM OF DATA ***************************
```

[1] Program Location Information

PTLDRIVM identifies the CA RC/Update driver. The STEPLIB and PTILIB DD names identify the necessary DB2 and product load libraries. MSGFILE specifies the file to which analysis messages will be written. DDLFILE identifies the data set to which the results (DDL and PBP commands) are written.

[2] Specified Analysis Options

This section identifies the strategy being analyzed (RCUPDATE), the DB2 subsystem containing the strategy (STRTSSID). The other Analysis options displayed identify the options specified on the Analysis screen.

Return Codes

Return codes are displayed at the end of analysis execution. They indicate any problems encountered during execution. This section lists and describes analysis return codes:

- 0—Successful
- 4—Warning in DDL
- 8—Error in DDL
- 12—SQL Error
- 16—Miscellaneous Errors
- 20—GETNAIM Error

Note: Return codes greater than 4 stop execution.

Create Analysis Screen

The CA RC/Update Create Analysis screen appears when Operation Mode (located at the top of CA RC/Update screens) is set to A (RC/Alter) and you are creating an object.

You can use the Create Analysis screen to perform the following actions:

- Specify the execution options for your create.
- Name the data set to which the create DDL is to be written.

RC/Alter will then analyze your create request.

Note: The analysis can be performed online or as a batch job.

Fields on this screen are based on fields on the Alteration Analysis screen.

Note: For information about the fields, press F1 (Help).

Template Analysis Screen

The CA RC/Update Template Analysis screen appears when Operation Mode (located at the top of CA RC/Update screens) is set to A (RC/Alter) and you are templating an object.

You can use the CA RC/Update Template Analysis screen to perform the following actions:

- Specify the execution options for your template.
- Name the data set to which the create DDL is to be written.

RC/Alter will then analyze your create request.

Note: The analysis can be performed online or as a batch job.

Fields on this screen are based on fields on the Alteration Analysis screen.

Note: For information about the fields, press F1 (Help).

Drop Analysis Screen

The CA RC/Update Drop Analysis screen appears if you are dropping an object and one of the following settings is present:

- On any CA RC/Update screen, Operation Mode is set to A (RC/Alter).
- On the CA RC/Update Default Drop Analysis screen, Execution Mode is set to A.

You can use the Drop Analysis screen to perform the following actions:

- Specify the execution options for your drop.
- Name the data set to which the create DDL is to be written.

Note: The analysis can be performed online or as a batch job.

Recovery Options Screen

The Recovery Options screen appears if Y or U are entered in the Alteration Analysis screen's Recovery field. The Recovery Options screen allows users to specify the parameters to be used to generate the recovery data set.

The recovery data set contains the commands necessary to drop the altered version of the object (and its data and dependents) and re-create the object in its original state. Any utilities requested as recovery options are executed against the newly created original-version objects during the execution of the recovery data set. The Recovery Options appear in the RECREATE section of the alteration analysis data set.

Note: For information about the fields, press F1 (Help).

Recovery Options Processing

From the Recovery Options screen, you can SAVE your changes or END the session.

Use the SAVE command to save the options displayed for this strategy in the ISPF profile. This command is useful when there is the need to quickly update the profile with the values used most often. When recovery options are saved, they become the default for all recovery analyses. These options will be used by default whenever an alteration recovery is performed. The recovery options can be changed before recovery begins. These options are also used for recovery in CA RC/Migrator. Accordingly, if recovery options are saved through CA RC/Migrator, they are used for alteration recovery.

Press F3 (End) to save the options for this recovery only and return to the Alteration Analysis screen. The ISPF profile will not be changed.

Analysis Options Screen

The Analysis Options screen appears when Y or U is entered in the Update Options field on any of the following screens:

- Create Analysis screen
- Alteration Analysis screen
- Template Analysis screen
- Drop Analysis screen

The Analysis Options screen is used to specify the unload and utility parameters that will be used for the alteration. A default SQL ID can also be specified. These options are used to create the alteration analysis data set.

Enter a question mark (?) in any option field to get help on that option.

Processing for Analysis Options

From the Analysis Options screen, the SAVE command can be used to save the options displayed for this strategy in the ISPF profile. When analysis options are saved, they become the default for all alteration analyses. These options are used by default whenever an alteration analysis is performed. Analysis options can always be changed before analysis begins. These options are also used for CA RC/Migrator strategy analysis. Accordingly, if analysis options are saved through CA RC/Migrator, they will be used for alteration analysis.

From the Analysis Options screen, press F3 (End) to save the options for this analysis only and return to the previous screen.

Control Options

SECURITY

Specify whether to include the security structure that belongs to the objects being altered. All security associated with the objects is incorporated, including security granted by users with the GRANT WITH GRANT OPTION. This option makes it easy to include security for altered objects, which is essential when object alterations affect users' access to data.

N

Does not include security. This is the default.

Y

Includes all security.

The BP AUTH command is issued to switch to the appropriate GRANTOR before issuing the GRANT statement. Appropriate authorities are required to execute the AUTH command.

Note: For more information, see the *CA Database Management Solutions for DB2 for z/OS Batch Processor Reference Guide*.

DEFAULT SQLID

Enter an SQL ID in the Default SQLID field to generate a SET CURRENT SQLID statement in front of the CREATE statements for explicitly qualified objects. Explicit objects include tables, indexes, views, and aliases. A SET statement might be mandatory if views contain references to unqualified table names.

The SET CURRENT SQLID statement might be necessary to guarantee that the correct ID is checked for authorizations at execution time. There can be three authorization IDs associated with the creation of an explicitly qualified object:

- The qualifying ID, which is stored as the CREATOR ID
- The primary authorization ID of the user who executed the CREATE statement, which is stored as the CREATEDBY ID
- The ID used to check the CREATE authorizations

The last ID is often a secondary ID that holds the authorizations necessary to CREATE the object. The user's primary ID (CREATEDBY) might have no DB2 authorities at all. Unless the SQL ID information is saved, the correct authorizations might not be included in the recovery data set. Leave this field blank to suppress generation of a SET CURRENT SQLID statement before CREATE statements of explicitly qualified objects.

DROP & RE-CREATE

Specifies whether to generate DROP and CREATE DDL for an object that is alterable. Valid values are as follows:

Y

Specifies to drop and re-create all objects that have changed.

N

Specifies to use the ALTER statement whenever possible to make the requested changes.

Note: To avoid versioning of objects due to the online schema evolution (extended ALTER functionality introduced in DB2 V8), you can use this option to force the dropping and creating of DB2 objects as was done prior to DB2 V8.

REGENERATE VIEWS

Specify whether to regenerate views:

N

Does not regenerate views, even if an alteration is made to a parent object upon which a view is dependent. No ALTER VIEW *view name* REGENERATE statements will be inserted into the generate DDL.

Y

Regenerates any view where an alteration was made to a parent object upon which the view is dependent. This affects views that are not dropped and re-created. This is the default.

REFRESH MQTs

Specify whether to refresh MQTs.

N

Specifies to not refresh MQTs that are dropped and re-created.

Y

Specifies to refresh MQTs that are dropped and re-created.

Utility Options

STATS

Specifies one of the following options:

Y

Runs the statistics utility on all affected indexes and tablespaces. If Y is entered, statistics utility control statements are generated for all involved indexes and tablespaces when the object is recovered. If you are going to REBIND plans, running the statistics utility will help the Optimizer make correct access path decisions when linking the plans. Any indexes whose tables have not been recovered will have the statistics utility executed on the indexes only.

N

Does not run the statistics utility. This is the default.

S

Generates SQL update statements using the object's current statistics utility values. RC/Alter will fetch the current object's statistics values from the DB2 catalog and generate UPDATE SQL statements to put the current statistics values back into the DB2 catalog. These SQL statements will be generated instead of using the statistics utility.

CHECK DATA

Specifies Y to run the check utility for any tables loaded with ENFORCE=NO. (All foreign key alters will have been loaded with ENFORCE=NO.) CA Fast Check or the CHECK DATA utility checks tablespaces for violations of referential constraints and reports information about detected violations. If the check utility is not run, tables with referential integrity can be placed in a CHECK PENDING status. The default is N.

IMAGE COPY

Specifies whether an Image Copy is performed on all partitions of all tablespaces or indexspaces involved in an alteration, on altered partitions only, or whether it is not performed at all.

Y

Generates IMAGE COPY control statements for all partitions of all tablespaces or indexspaces involved in an alteration. This causes the Image Copy utility to be run during execution of the alteration.

P

Generates IMAGE COPY control statements only for altered partitions of tablespaces or indexspaces involved in an alteration. This causes the Image Copy utility to be run during execution of the alteration.

N

Does not generate any IMAGE COPY control statements. This is the default.

BIND/REBIND

Specifies whether to bind or rebind all plans dependent on altered objects:

R

Rebinds any dependent plans. The REBINDs will execute after all object creation and table loads.

B

Binds any dependent plans. The BINDs will execute after all object creation and table loads. If the original BIND used a concatenation of PDS libraries for its DBRMs, check the order of the concatenation in the analysis file to make sure it is correct.

N

Does not bind or rebind any plans. This is the default.

C

Rebinds any dependent plans and only the most current version of any dependent packages.

O

Binds only operative dependent plans and only operative dependent packages. This is the same as the B option, except that only operative plans are bound.

V

Rebinds only valid dependent plans and only valid dependent packages. This is the same as the R option, except that only dependent plans are rebound.

REORG TS/IX/B/N

Specifies whether to include a REORG utility in the DDL produced by the analysis. These options are in effect only if a reorganization is needed when one or more of these attributes is altered: FREEPAGE, PCTFREE, PRIQTY, SECQTY, ERASE, VCAT, or STOGROUP (and COMPRESS for a tablespace).

TS

Generates tablespace REORG utilities.

IX

Generates index REORG utilities.

B

Generates both tablespace and index REORG utilities.

N

Does not generate REORG utilities. This is the default.

REBUILD INDEX

Specifies whether to include the CA Fast Recover or REBUILD INDEX utilities in the analysis. This option applies only if the index was created using the DEFER clause of the CREATE INDEX statement.

Y

Includes the utilities in the analysis.

N

Does not include the utilities in the analysis. This is the default.

Output Listing Options

ANALYSIS HEADER

Enter Y to include a header in the analysis output. The analysis header lists all the strategy and analysis options you have selected. The default is Y.

IMPACT ANALYSIS

Enter Y to include an impact analysis in the strategy analysis. Impact analysis lists the names and creators of the objects included in the alteration. (Tablespaces are listed by name and database, rather than name and creator). It also indicates whether objects have been edited. The default is Y.

RPT IN DDLFILE

Indicate where the Change Analysis Report is to appear.

Y

Includes the report at the end of the output DDLFILE and not in the message file (MSGFILE).

N

Writes the file to MSGFILE.

More information:

Analysis Output (see page 284)

Data Unload Options

When RC/Alter makes changes to tables that are not supported by the DB2 ALTER TABLE statement, it drops the table and re-creates it. By default, all data is reloaded into the altered table. This includes tables included because of parent object (tablespace) alterations. The Data Unload options enable users to limit the data reloaded into altered tables.

ALL ROWS

Specifies whether to retrieve all rows. Enter **Y** to retrieve all rows. Specify All Rows or Number Rows (not both).

NUMBER ROWS

Specifies the maximum number of rows to be etrieved. Specify All Rows or Number Rows (not both).

DATA STATISTICS

Specifies whether to perform SELECT COUNT(*) statements for rows in tables that will be unloaded. This information appears as part of the alteration Summary Statistics at the end of the data set. Use this option to gauge the volume of data to be unloaded. This option does slow down the process. Valid values are Y (yes) or N (no). The default is N.

TRUNCATE

Specifies whether you want the CA Fast Unload utility to truncate column values that exceed the size of their target columns. This situation can occur if you have altered the definition of a table column, and the new definition is shorter than the original column definition.

Model Options

Model members are used when creating the DDL for the alteration.

Note: For more information about model members, see the *CA Database Management Solutions for DB2 for z/OS General Facilities Reference Guide*.

MODEL ID

Enter a model ID if utilities will be executed as part of the alteration. If a model ID is not specified, @DEFAULT is used, and a selection list of model IDs that match the model creator criteria appears.

If a model ID is specified that contains errors, the General Model Utilities panel automatically displays and errors are flagged. The model ID can then be corrected for the recovery. These changes are temporary and are not saved. The temporary changes can be saved by using the Update Model option to update the model ID. The temporary changes will appear in the model ID, which can then be reviewed and saved.

MODEL CREATOR

Enter a Model Creator if utilities will be executed as part of the execution. If no model creator is specified, a selection list of model creators that match the model ID criterion appears, or the model creator of the @DEFAULT model ID is used.

Note: For more information, see the *CA Database Management Solutions for DB2 for z/OS General Facilities Reference Guide*.

UPDATE MODEL

Enter Y to update the model ID. The General Model Utilities panel appears.

Note: For more information about model IDs, see the *CA Database Management Solutions for DB2 for z/OS General Facilities Reference Guide*.

Dataset Deletion Options

Data set deletions are performed as the last phase in the strategy.

UNLOAD

When migrating data, RC/Alter creates data sets to hold the unloaded data. Enter Y to delete data sets created for unloaded data. The default is N (no).

Note: For more information about how RC/Alter unloads data, see the *CA Database Management Solutions for DB2 for z/OS Batch Processor Reference Guide*.

TEMPORARY

Enter Y to delete temporary data sets used by the load, check, and copy utilities. The default is N (no). The temporary data set names are set in model members.

Note: For information about model members, see the *CA Database Management Solutions for DB2 for z/OS General Facilities Reference Guide*.

Exclusive Options

No .AUTHS

Indicate whether to generate an analysis output data set with no Batch Processor .AUTH statements.

YES

Generates the analysis output data set with no .AUTH statements. All options are generated as usual, but with no .AUTHs. Use this option if the analysis output contains no statements that require APF authorization and the strategy is to be executed online.

Note: RC/Alter might not be able to re-create objects exactly as they currently exist. Also, the user ID executing the analysis output must hold the necessary authority to perform all statements.

NO

Generates the analysis output data set with the .AUTH statements generated as needed.

RC/Alter Datasets and Batch Mode

When batch mode is used, the user supplies the name of an output data set. This data set contains the commands necessary to perform the create, drop, template, and (DB2 supported) alters requested in the current batch mode session. When the mode is changed to Online or when you exit, the data set is closed and unallocated.

RC/Alter also prompts the user for the name of a data set: the data set that will contain the commands necessary to perform the requested non-DB2 alteration. The name of this data set must be different than the Batch mode data set. (If the same name is specified, RC/Alter displays a warning that the specified data set is already allocated.) A different data set name is required for each RC/Alter request.

Therefore, to perform all requested operations, users must execute the Batch mode data set and all RC/Alter data sets.

Multiple DB2 Subsystem Processing

The Batch Processor automatically logs to the subsystem of the initial CONNECT. When the Batch Processor executes an Alteration Analysis job that performs loads and unloads on multiple DB2 subsystems, it now returns a system completion code (SC) when it finishes processing on one subsystem.

This code (SC) provides restartability even when multiple DB2 subsystems are involved. If the user tries to override a data set that completed with a code of SC, a warning message is issued and the analysis process is terminated. This check prevents the accidental reuse of a data set that might need to be restarted to finish an alteration execution. When the Batch Processor issues a return code of SC, a sync point is written, and the message DISCONNECTING FROM SSID ssidname, CONNECTING TO SSID ssidname is issued.

User-Defined Symbolic Parameters

For each table selected for data migration, the data is unloaded to a data set. Each table's data set name must be unique, or an error will occur during the unload phase of an alteration. The image copy utility (CA Quick Copy or COPY) creates an image copy of the tablespace or a data set within a tablespace. Duplicate image copy output data sets are not permitted; each data set name must be unique.

To alleviate the need to edit the strategy to enter unique unload and image copy data set names, the copy model member is used. This model member is accessed through Model Services, which supports the use of symbolic variables to define data set names. These symbolic variables are automatically expanded at processing time to full data set names. The symbolic variable capability permits users to automatically generate unique data set names without editing the resulting strategy data set.

Note: For more information about symbolic parameters, see the *CA Database Management Solutions for DB2 for z/OS General Facilities Reference Guide*.

Available Symbolic Parameters

Control statements are generated for the alteration of an object using model members. The model members have symbolic parameters that can be altered. A symbolic parameter must start with a percent sign (%).

To have a period separating a symbolic parameter from the next field in the name, two periods must be used. For example, %USERID..%TBNAME..DATA would expand to PTIPH.EMPLOYEE.DATA if the User ID is PTIPH and the table name is EMPLOYEE.

Data Set Names

A default data set name is defined in each model member. It can be customized by changing the value of the data set symbolic parameters in the appropriate model member through Model Services.

Storage Devices

Specify the storage devices to be used by each model member by customizing the unit symbolic parameter in the appropriate model member.

Space Allocations

In all space allocations, the alteration strategy data set or recovery analysis data set can be edited.

Unload Data Sets

At execution time, the unload program dynamically determines the amount of space required to hold the unloaded data. A SELECT COUNT is performed to return the number of rows. The total number of rows is multiplied by the table's row size. Space is allocated in 6200 character blocks. One tenth of the total space required is used to calculate the primary and secondary extent amounts.

For example, if 1,000,000 bytes of storage is required to unload a table, the generated space allocation would be (6200,(16,16),RLSE). This technique reduces the initial primary space allocation requested.

Discard/Error Data Sets

RC/Alter allocates any necessary discard and error data sets to the unit specified in the appropriate model member symbolic parameter. The space allocation is (CYL,(10,10),RLSE) with a data set disposition of (MOD,CATLG,CATLG).

Processing the Alter

After entering information in the Alteration Analysis screen, there are three options. When the alteration data set has been created, execute it at any time through the CA Batch Processor.

Do one of the following:

- To return to the Alter screen without processing the alter, press F3 (End). More changes can be made to the screen before processing.
- To return to the object selection panel without processing the alter, enter the CANCEL command.
- Enter **S** in the Start Analysis field and press Enter to invoke RC/Alter and create the alteration data set. The Processing Screen appears.

More information:

Analysis Output (see page 284)

Alter Requirements

This section provides information about the DB2 objects you can change using RC/Alter. You must use RC/Alter to alter the objects described in the following table. Information is organized according to objects. A listing is made for each object, outlining the changes (if any) that require the use of RC/Alter.

Object	Use RC/Alter to Change
Storage Group	Storage Group name VCAT name
Database	Database name Creator ID **Note:** If you have DB2 V2.2 or earlier, RC/Alter is also used to change the Storage group and the Buffer Pool.
Tablespace	Tablespace name Database Partition Information Segment Information
Trigger	Trigger Name Table Correlations Trigger Text Trigger columns
Table	Column: name, data type, length, null attribute, fieldproc, or location. Table: name, creator, database, tablespace, or EDITPROC. To add a column with the NOT NULL attribute. To delete a column.
Index	Partitioned Only—RC/Alter used to make the alteration because related tablespace must be dropped.
View	Alters are made by dropping and recreating the view.
Synonym	Because synonyms have no dependent objects, RC/Alter does not have to be called. Alters are made by dropping and recreating the synonym.

Object	Use RC/Alter to Change
Alias	Because aliases have no dependent objects, RC/Alter does not have to be called. Alters are made by dropping and recreating the alias.

Propagation of Changes

This section outlines the propagation of changes to dependent objects made by RC/Alter. Objects are listed hierarchically, with the storage group listed first, and the lowest object listed last. Other DB2 objects have no propagation ramifications.

The following defaults can be specified:

- For storage groups, CREATOR, VCATNAME, and VOLUME1 - VOLUME6.
- For databases, CREATOR, STOGROUP, BUFFER POOL, and CCSID.
- For tablespaces, CREATOR, DATABASE, VCAT NAME, STOGROUP, PRIMARY QTY, SECONDARY QTY, FREEPAGE, PCTFREE, PARTITIONED, BUFFER POOL, CLOSE, SEGSIZE, COMPRESS, LOCKSIZE, LOCKMAX, DEFINE, TS TYPE, CCSID.
- For tables, CREATOR, DATABASE, TABLESPACE, AUDIT, COLUMN DATA TYPE, COLUMN NULL INDICATOR, COLUMN DEFAULT IND., DATA CAPTURE, Table Type, CCSID, and Volatile.

Analysis Output

Analysis output consists of two files: the analysis output file and the message file.

The analysis output file contains the commands necessary to execute the alteration. The analysis output file by default is divided into three parts: header, commands, and impact analysis.

The message file contains any analysis messages.

The analysis message file appears immediately after online analysis, if there are messages. For batch execution, the analysis message file is specified as MSGFILE in the analysis JCL and is routed to the destination specified on the //MSGFILE DD statement. The default destination is the SYSOUT class specified in the PRINT CLASS field of the Print Parameters Profile screen. The analysis messages are listed and described in the *Messages Guide*.

Analysis Output File

The analysis output file by default is divided into three parts.

Header

Describes the selected options.

Commands

Lists the commands to unload and load the altered version of the object, commands to create the recovery strategy when the alteration strategy is executed, and commands to execute the alteration.

Impact Analysis

Lists, by type, the objects included in the alteration.

Analysis Output File Options Screen

The Data Statistics and Analysis Listing Options on the Analysis Options screen permit users to specify whether to include the data statistics, analysis header, and impact analysis in the analysis output file.

Analysis Listing Options

The Analysis Listing Options control the informational sections of the analysis output file.

ANALYSIS HEADER

Enter Y in this field to generate a report of strategy specifications and analysis options in the analysis output data set. The header can be excluded by specifying N in this field.

IMPACT ANALYSIS

Enter Y in this field to generate a summary report at the bottom of the analysis output data set. This summary report will list all the objects for which DDL has been generated. It will also display an indication of whether the object definition has been edited. Exclude the Impact Analysis section by entering N in this field.

DATA STATISTICS

Enter Y in this field for the analysis program to issue a SELECT COUNT (*) statement for each table whose data is to be moved. These numbers will be reported in the Impact Analysis section of the output. To exclude data statistics with the impact analysis, enter N. Data statistics are described in the Impact Analysis.

Header

The header is the first part of the analysis output file and is included by default. To exclude the header, N must be specified in the Header prompt of the Analysis Options screen. The analysis header is especially helpful when reviewing output generated several days earlier. The analysis output header indicates the strategy used to generate the output, the strategy definition, and the analysis options.

The information in the header is also displayed during execution. Batch Processor comments appear as part of the execution output; however, they are truncated to 72 characters. Part of a sample header for an object alteration analysis is shown in the following example. Notice that all the lines are Batch Processor (PBP) comments. PBP comments must have two hyphens (--) as the first two characters of the line:

```
--STRATEGY INFORMATION:
--STRATEGY ==> RCUPDATE  DESCRIPTION  ===> RC/UPDATE ALTERATION
--CREATOR  ==> USER1     SHARE OPTION ===> N (U,Y,N,X,L)  SRC SSID ===> D81A
--
--
--ALTERED OBJECTS:
--  OBJECT TYPE         NAME              CREATOR
--  TABLE               EDIT_DEMO         USER02
--
--ANALYSIS OPTIONS:
```

The header is broken down into three parts: Strategy Information, Altered Objects, and Analysis Options.

Strategy Information

The Strategy Information is the first part of the header and lists the strategy name, creator, description, share option, and source SSID. If the analysis was performed in batch mode, the Strategy Name specified on the Alteration Analysis screen displays as the strategy name. If the analysis was not performed in batch mode, RCUPDATE displays as the strategy name. The description will always be RC/Update Alteration.

Altered Objects

The Altered Objects section of the header lists the objects for which an alter request has been issued. This section does not list the objects altered as a result of the request.

The objects are arranged by DB2 object type. The information includes the DB2 object type, object name and creator ID.

Analysis Options

The Analysis options portion of the report lists all selected analysis options.

The following fields delimit the options portion of the report.

ANALYSIS OPTIONS

The Analysis Options selected are displayed in a format similar to the format of the Alteration Analysis screen.

END OF ANALYSIS HEADER

This signals the end of the header and the beginning of the commands section.

RC/Alter Commands

There are three parts to the commands section:

- Commands to create the recovery data set
- Commands to execute the alteration
- Commands to unload and load data

Recovery Commands

The commands used to create the recovery data set appear only if the Recovery option is specified from the Alteration Analysis screen.

Part of a sample alteration analysis data set appears in the following example:

```
     .AUTH B0625TO

[1] .CALL UNLOAD
     .DATA
      * * *
     .ENDDATA

     .SYNC 5     'UNLOAD TABLE B0625TO.EMP'

     .SYSTEM SQLDDL
     .CONNECT PTI2

[2] .CALL SNAPSHOT
     .ALLOC FI(RCVRFILE) +
       DA('B0625TO.PIPE.TST(RECOVER1)') OLD
     .DATA
     .AUTH B0625TO
        DROP TABLE B0625TO.EMP;
     .SYNC
     TABLE     B0625TO    EMP
     .AUTH B0625TO
        DROP ALIAS B0625TO.TABEMP;
     .SYNC
     ALIAS     B0625TO    TABEMP
     .AUTH B0625TO
        DROP ALIAS B0625TO.TWOVIEW;
     .SYNC
     ALIAS     B0625TO    TWOVIEW
      RECREATE
      STRTSSID PTI2
      DELTEMP
      DELUNLD
      HEADER
      LOG
      TRAILER
      UNLOAD
      GENUNIT SYSDA
      SQLID B0625TO
      UNLDDSN PTI.%TBNAME..RECDATA
      UNLDUNIT SYSDA
      UNLDLBL 1,SL,EXPDT=90032
     .ENDDATA
     .FREE FI(RCVRFILE)

     .SYNC 10

[3] .AUTH B0625TO
         DROP TABLE B0625TO.EMP;
```

[1] Alter UNLOAD

The first part of the strategy unloads the data that will be needed to load the altered version of the object. For this example, the actual UNLOAD syntax is replaced with asterisks (* * *).

[2] SNAPSHOT

The second part of the strategy calls the SNAPSHOT program, which gathers information about the object being altered. When the alteration is executed, SNAPSHOT generates the commands necessary to create the recovery data set. SNAPSHOT first allocates the recovery file, then unloads the object data. The unload program is called internally through SNAPSHOT. So while no UNLOAD commands will be included in this part of the strategy when executed, unloads will be performed. The Batch Processor output displays the tables being unloaded.

The commands that appear in SNAPSHOT's .DATA section are the commands that will perform the recovery. These commands will drop the altered version of the object, re-create the original version of the object, and load the original data. The commands are not executed when the alteration is executed, but rather are written to the recovery data set along with all other necessary information. The RECREATE portion of the SNAPSHOT program provides RC/Alter with the recovery option information it needs to correctly generate the recovery data set. When the recovery data set is executed, the commands are executed to recover the original version of the object.

Once all necessary information is written to the recovery file, the file is freed. The recovery file is the same as an ordinary alteration strategy except that the unloads for the objects have already been performed.

[3] Alteration Commands

The third part of the alteration analysis data set contains the commands that will perform the requested alteration when executed.

Create Commands

There are many commands to assist with a View Creation session. A brief description is presented here.

Note: For more information about these commands, see the *CA Database Management Solutions for DB2 for z/OS General Facilities Reference Guide*.

CAPS

Works as a toggle, turning CAPS lock on and off.

COLS ON or OFF

Controls the display of a second scrollable area at the bottom of the screen called the Column Area.

FORMAT

Lets you format the view text into a more readable format with indentation.

TEXT

Is intended for fast entry of text using fast typing techniques. When the TE command is entered, the CMD area is removed and automatic cursor skip will occur upon reaching the end of the text line. This permits quick entry of SQL statement text without regard to cursor position. When Enter is pressed, TE is removed and the user is placed back into standard text entry. If words are split between lines, and PARSE is on, the split will be fixed when Enter is pressed.

WORD WRAP

Turns word wrap on and all lines are restructured to fit as much on a line as possible. Enter the WW command to reflow fragmented statement text into a uniform text lines.

WORD SPLIT

Separates the SQL statement text into tokens and places each token on a separate line. After entering the WS command and performing any edits, enter the WW (Word Wrap) command to reflow the tokens into uniform text lines.

NULLS

Works as a toggle, turning on and off the NULLS feature. When NULLS is on, space at the beginning or end of the line is considered null. When NULLS is off, space at the beginning or end of the line is considered blank spaces. By default, NULLS is ON.

PARSE ON or OFF

Turns on and off the parsing feature of the editor. When PARSE is on, the entered SQL statement is parsed looking for special characters.

Impact Analysis

The impact analysis appears at the end of the file (if Impact Analysis was requested on the Analysis Options screen). Impact analysis for object alterations provides the following information:

- The number of each object type included in the strategy.
- The name and creator of each object included in the strategy.
- For edited objects and objects changed through global changes, the old and new name and creator ID. Edited objects are flagged.
- For table data, the number of rows per table included in the strategy and the total number of rows, if Data Statistics were requested.

Note: For information about the fields, press F1 (Help).

If Data Statistics were requested, they will appear after the TABLE information in the Impact Analysis.

Alteration Considerations

The following sections provide information about which DB2 object changes will require dropping and recreating the object (as opposed to using a DB2 ALTER statement). Information is organized according to objects. A listing is made for each object, outlining the changes that require dropping and recreating the object.

Storage Group Considerations

Storage group alteration considerations are:

- Storage group name
- VCAT name

Database Considerations

Database alteration considerations are:

- Database name
- Creator ID
- Storage Group
- Buffer Pool

Tablespace Considerations

Tablespace alteration considerations are:

- Tablespace name
- Database
- Partition Information
- Segment Information

Table Considerations

Table alteration considerations are:

- Change a column's name, data type, length, null attribute, fieldproc, or location
- Add a column with the NOT NULL attribute
- Delete a column
- Change the table's name, creator, database, tablespace, or EDITPROC

Index Considerations

If the index is partitioned, RC/Alter is used to make the alteration because the related tablespace must be dropped. If the index is non-partitioned, changes can be made by dropping and recreating the index. RC/Alter will use a DB2 ALTER, when possible, to change the index.

View Considerations

All alters are made by dropping and recreating the view.

Synonym/Alias Considerations

Alters are made by dropping and recreating the synonym or alias. Because synonyms and aliases have no dependent objects, no other objects are involved with a synonym or alias alter.

Referential Integrity Alterations

The following complexities are involved in referential integrity changes:

- Foreign key changes (see page 292)
- Unique constraint changes (see page 293)
- Referential integrity rebuild strategy (see page 293)

Foreign Key Changes

Foreign key changes are handled by dropping and recreating the foreign key.

Unique Constraint Changes

Possible unique constraint changes include:

- Data Type or Length Change

  Data is unloaded from any tables that are affected by the unique constraint change. The tables dependent on the unique constraint are also dropped and re-created using the new definition for the unique constraint. The unload program will handle all necessary data conversion for all tables.

- Deletion of Columns From the unique constraint

  The unique constraint index is automatically adjusted for the deleted column. Any dependent table's foreign key is updated to remove the deleted columns from the foreign key definition.

- Addition of New Columns To the Unique Constraint

  RC/Alter does not propagate the addition of a column to the Foreign key because it does not know which additional columns in the dependent tables to include in the new Foreign key definitions. ALTER statements for the old foreign keys are generated for each table affected by the unique constraint change. The ALTER statements can be changed to add the necessary foreign key columns to match the new unique constraint definition.

- Reordering of Unique Constraint Columns

  If the same columns are involved in the unique constraint, the columns will be reordered in the unique constraint index and in all related foreign key definitions.

Referential Integrity Rebuild Strategy

This section discusses the strategy followed by RC/Alter for rebuilding referential integrity rules. There will be other SQL and CA Batch Processor commands present in the alteration file in addition to those mentioned in the following, but they are not related to referential integrity.

- Data is unloaded from every table (including dependent tables) affected by a unique constraint data type or length change. All necessary conversions are handled as part of the unload process. (The unloading and loading of data is an option on the Analysis Options screen.)
- All dropped tables are re-created with their unique constraints designated.
- All affected indexes are rebuilt.
- ALTER statements are executed to re-create the necessary foreign keys for any tables (including dependent tables) affected by the alterations. By using ALTER statements to add the foreign keys, any potential problems from cyclical table definitions or self-referencing tables are eliminated.

- All loads are performed with “ENFORCE=NO” for any tables with foreign keys to disable referential integrity checking. After the LOAD is performed, the tables are placed into a check pending status. Performing the loads with “ENFORCE=NO” prevents any potential load problems resulting from cyclical table definitions or self-referencing tables. (The unloading and loading of data is an option on the Analysis Options screen.) For tables with primary keys and no foreign keys, loads are performed with “ENFORCE=YES.”
- CA Fast Check or CHECK DATA is performed for each tablespace placed into a check pending status (this is an option on the Strategy Analysis Options screen in CA RC/Migrator.)
- The statistics utility is executed for any tablespaces affected by the alteration (this is an option on the Strategy Analysis Options screen in CA RC/Migrator.)
- REBINDs are performed on plans affected by the alteration (this is an option on the Strategy Analysis Options screen in CA RC/Migrator.)

Chapter 15: Storage Group

This section contains the following topics:

Overview of Storage Groups

You can use Storage Group objects to create, template, alter, and drop DB2 storage groups. A storage group is a DB2 object that represents a named set of DASD volumes controlled by a specified VSAM catalog (VCAT), on which DB2:

- Allocates storage for tablespaces and indexes
- Defines the necessary VSAM data sets
- Extends and deletes VSAM data sets as required

DB2 monitors and maintains the storage groups, and uses them to store DB2 table and indexspaces. A storage group can be assigned to a database, tablespace, or index space. All tables that reside in a given tablespace use that tablespace's storage group. Following are some of the ways DB2 helps manage external storage requirements.

- When a tablespace is created, DB2 uses VSAM access method services to define the necessary VSAM data sets. After the data sets have been created, they can be processed with access method service commands that support VSAM control-interval (CI) processing.
- When a tablespace is dropped, DB2 automatically deletes the associated data sets.
- When a data set in a simple tablespace reaches its maximum size of 2 GB, DB2 can automatically create a new data set.
- When needed, DB2 can extend individual data sets.

Storage Group Functions

The functions to create and maintain storage groups include:

- Volume and VCAT Selection Help. When defining a storage group, the names of the volumes or VCAT that the storage group references must be entered. Instead of having to remember and enter specific volume and VCAT names, they can be selected by entering selection criteria to receive a list of volumes or VCATs.
- Line Commands. Insert and Delete line commands are supported to assist in manipulating the volume list.
- Template Option. The Template option permits creation of a new storage group, using an existing storage group as a model. Enter selection criteria to receive a more specific list of storage groups.
- Alter Option. The Alter Storage Group option lets you change any characteristic of the storage group. While DB2 allows only the addition or removal of volumes from the storage group, we let you change the VCAT or even the storage group name. This is done by dropping, and then recreating the storage group. Like the Template function, the Alter function lets you select from lists of storage groups, VCATs, and volumes.

Access Storage Group Functions

You can access storage group functions by requesting to create, alter, template, or drop a storage group.

To access storage group functions, do one of the following:

- To create a storage group, enter **C** in the Option field and **S**G in the Object field of the main header and press Enter.
- To alter, template, or drop a storage group, enter **A**, **T**, or **D**, respectively, in the Option field and **SG** in the Object field of the main header and press Enter. Enter a storage group name in the Item Name field and a user ID in the Creator field to get a more specific selection list.

Selection Panels

When altering, dropping, or creating (by template) a view, a selection panel is presented to select the view if not already specified. Make the view selection list more specific by specifying selection criteria for the Item (view) Name and Creator ID prompts in the header.

Use the EQF facility to further specify the views displayed, and use scrolling commands to scroll through information.

Note: For more information about EQF and scrolling commands, see the *CA Database Management Solutions for DB2 for z/OS General Facilities Reference Guide*.

The View Selection screen displays all the fields available in the SYSIBM.SYSTABLES table. This information is available to make selection of the views easier.

To choose items from the listing, type **S** next to the items to select. Press Enter to register the selection.

Note: For information about the fields, press F1 (Help).

Selection Criteria

The Item Name and Creator fields let you specify a specific storage group name or search conditions for retrieving multiple objects. These fields accept the following SQL search criteria:

Asterisk (*)

Selects all objects. All storage groups in the system are displayed.

Percent sign (%)

Is a wild card character that will match zero or more characters.

Underscore (_)

Indicates that any one character can occupy that position.

Note: For more information about SQL search criteria, see the *CA Database Management Solutions for DB2 for z/OS General Facilities Reference Guide.*

The Stogroup selection panel displays information from the SYSIBM.SYSSTOGROUP table to make selection of the storage groups easier.

To choose items from the listing, type an S (select) in the S column next to the items you want to select.

Note: You can also enter a question mark (?) or query option, to display a CA RC/Query menu or report for the option specified. For lists of report options per object type, see the *CA RC/Query User Guide*. For information about the fields, press F1 (Help).

Press Enter to register the selection:

- For templates and alters, each storage group selected is processed individually, beginning with the storage group first on the selection list, and continuing in the order of the selection list.
- For standard drops, all selected storage groups are dropped together.
- For recoverable drops, each storage group is dropped one at a time.

VCAT Selection List

If the name of the VCAT you want to use is not known, enter an asterisk (*) in the Vcat field of the storage group screen. This provides a complete listing of VCATs currently referenced within DB2. Selection criteria can also be entered for a partial listing. When you press Enter, a list of VCATs appears from which you can select.

Insert a Volume from Another Storage Group

The Volser field on the storage group screen indicates the names of the volume serial IDs that you want to include.

Follow these steps:

1. Insert a new volume (unless overwriting an existing volume).
2. Enter the first letters of the volume name followed by a percent sign (%) in the Name field.

   Note: If the volume name is unknown, you can enter an asterisk (*) to display a list of *all* volumes from *all* current storage groups. However, this can be very time-consuming.
3. Press Enter.

   The Volume Selection List screen appears.
4. Type **S** next to each volume to select and press Enter.

   Note: Enter the S (shrink) toggle command in the command line to view the volumes that have been queued.
5. Press F3 (End) to return to the Storage Group screen.

   The selected volumes are automatically inserted.

Storage Groups vs. VSAM Data Sets

To maintain closer control over the physical storage of tables and indexes, users can define and manage their own VSAM data sets instead of using storage groups. To do this, there must be at least one VSAM catalog. The VSAM data sets can then be defined and managed by selecting a specific (explicit) VCAT when creating tablespaces and indexes.

Storage Group Commands

The HEADER and COMPARE commands can make working with storage groups easier as follows:

HEADER

Toggles the header on and off. This is helpful when viewing a long list of VCATs or volumes. Enter **H** or **HEADER** in the command line to toggle the header.

COMPARE

Displays old and new versions of the storage group for comparison. Use the RESET command to reset a changed volume to its original status.

Note: For more information about the primary commands, see the online help.

Storage Group Create Option

The Storage Group Create option performs all functions of the SQL CREATE STOGROUP statement. This statement defines a set of VSAM controlled volumes on which storage can later be allocated for tablespaces and indexes. When defining a storage group, you must enter the names of the volumes or VCATs that the storage group will reference. Selection lists can be used.

More information:

VCAT Selection List (see page 298)
Insert a Volume from Another Storage Group (see page 298)

Storage Group Create Screen

The Storage Group Create screen provides an easy way to supply the information necessary for storage group creation.

Note: For information about the fields, press F1 (Help).

The insert and delete commands can be used to manipulate the volume entries. (Block mode is also supported.) Although move and copy are valid commands, they are irrelevant, as volume order is not important. Also, replicate is not supported, as the same volume name cannot be designated more than once. Enter an asterisk (*) to select an SMS-managed volume.

Processing Considerations for Storage Group Creation

DB2 does not validate the volume names or VSAM catalog name when the storage group is created. These objects are only validated when a tablespace or index is defined that references the storage group. Therefore, when a tablespace or index that references a storage group is created with an invalid VSAM catalog name or volumes, the tablespace or index create will fail. In the case of invalid volumes, DB2 will issue a CONNECT command to the master console for the missing volumes.

Confirming the Creation

The Creation Confirmation screen enables users to accept, edit, or reject the DDL to be used to create the object.

Storage Group Template Option

The template option allows the creation of a new storage group using an existing storage group as a template. A template session is actually a create session with the additional step of selecting a storage group as a starting point. The name of the templated storage group must be changed on the Storage Group Template screen. The other fields can be changed as needed. Dependent objects are not included in the template operation. Press F3 (End) when all changes have been made.

More information:

Storage Group Create Screen (see page 299)

Storage Group Alter Option

The Storage Group Alter screen performs all functions of the SQL ALTER STOGROUP statement. The alter statement changes the definition of a storage group. The DB2 ALTER statement can change all the parameters used initially to create the storage group, except the VSAM catalog name and the name of the storage group itself.

You can add or delete volumes using the DB2 ALTER STOGROUP statement. The changes do not affect the storage of existing objects in the storage groups, but do affect the future assignments of storage within the group: an object defined later can be assigned to an added volume, but cannot be assigned to a deleted volume.

To change the storage group name, VCAT name, or creator name, you must drop and re-create the storage group. If this type of change is made, the RC/Alter Specification screen appears and you will be prompted for information about the data set to which the DDL should be written. The DDL must then be executed in batch mode.

Note: For more information, see the “Object Definition Defaults” chapter.

If the alter is one that can be performed by a DB2 ALTER STOGROUP statement, a confirmation screen appears where you can accept, edit, or reject the DDL to be executed.

More information:

Storage Group Create Screen (see page 299)

Storage Group Drop Considerations

Rules of DB2 specify that a storage group cannot be dropped if it contains tablespaces or indexspaces. The tablespaces or indexspaces necessary for the storage group drop are automatically dropped, and then that storage group is dropped. If the default storage group of a database is dropped, an explicit storage group name must be used when creating a tablespace or index in the database.

When using the standard drops (S or SO), if one or more of the requested drops cannot be completed, none of the drops will be committed.

Note: For information about the various drop options (S, SO, R, RO), see the “Using CA RC/Update” chapter.

DDL Execution

When F3 (End) is pressed from the Create, Alter, Template or Drop screen, a confirmation screen appears before the DDL is executed. Accept, edit, or reject the DDL to be used to complete the request.

Note: For more information, see the “Using CA RC/Update” chapter.

If your changes are complex, require dropping and recreating objects (and their dependencies), or the current CA RC/Update Operation Mode is set to A (RC/Alter), you can perform a complete analysis of your changes that includes rebuilding the DDL and optionally including utility statements and extended DDL generation and analysis options.

Chapter 16: Database

This section contains the following topics:

Overview of Databases

In DB2, a database is a set of DB2 objects. When a DB2 database is defined, a name is given to an eventual collection of tables and associated indexes, as well as to the table and indexspaces in which those tables and indexes reside. A single database, for example, can contain all the data associated with one application or with a group of related applications. Collecting that data into one database lets you start or stop access to all the data in one operation, and grant authorization to access all the data in a single unit.

You can use the database object option to create, template, alter, and drop databases.

Shared Read-Only Data (ROSHARE)

Shared read-only data allows multiple DB2 subsystems to share the same DASD copy of a database. This is accomplished by creating a database on the owning subsystem and specifying ROSHARE OWNER in the CREATE DATABASE syntax. That database and all of its dependent objects must then be created on the other DB2 subsystems from which you want to read the data. These databases are identified as shared read-only databases.

When tables are created on the reading subsystems, they must specify the OBID (object ID) of the corresponding table on the owning subsystem database. DB2 uses the OBID attribute to identify which VSAM data sets should be accessed and read. Data sets are created only on the owning system, and they are shared by all shared read-only databases on other systems. You can specify the ROSHARE option for each database and the OBID for each table using the Table Database Create screen. The proper SQL statements will be generated.

Access Database Functions

You can access database functions by requesting to create, alter, template, or drop a database.

To access database functions, do one of the following:

- Type **C** in the Option field and type **DB** in the Object field of the main header, then press Enter to create a database.
- Type **A**, **T**, or **D**, respectively, in the Option field and type **DB** in the Object field of the main header, then press Enter to alter, template, or drop a database.

  You can enter a database name in the Item Name field and a user ID in the Creator field to get a more specific selection list.

Selection Panels

When altering, dropping, or creating (by template) a view, a selection panel is presented to select the view if not already specified. Make the view selection list more specific by specifying selection criteria for the Item (view) Name and Creator ID prompts in the header.

Use the EQF facility to further specify the views displayed, and use scrolling commands to scroll through information.

Note: For more information about EQF and scrolling commands, see the *CA Database Management Solutions for DB2 for z/OS General Facilities Reference Guide*.

The View Selection screen displays all the fields available in the SYSIBM.SYSTABLES table. This information is available to make selection of the views easier.

To choose items from the listing, type **S** next to the items to select. Press Enter to register the selection.

Note: For information about the fields, press F1 (Help).

Commands

The HEADER and COMPARE commands can make working with databases easier as follows:

HEADER

Toggles the header on and off. This is helpful when viewing a long list of databases. Enter **H** or **HEADER** in the command line to toggle the header.

COMPARE

Displays old and new versions of the database for comparison. This can be useful when you are templating or altering a database. Use the RESET command to reset a changed attributes to their original status.

Note: For more information about the primary commands, see the online help.

Database Create Option

The Database Create option performs all the functions of the SQL CREATE DATABASE statement. We provide a form-like screen where users supply the information to generate the DDL to be used to create the database. When defining a database, a default storage group can be specified directly or from a selection list.

Database Create Screen

The Database Create screen provides an easy way to supply the information necessary for database creation.

Note: For information about the fields, press F1 (Help).

Press F3 (End) to process or cancel to exit.

Confirming the Creation

The Creation Confirmation screen enables users to accept, edit, or reject the DDL to be used to create the object.

Database Template Option

The template option allows the creation of a new database using an existing database as a template. A template session is actually a create session with the additional step of selecting a database as a starting point. The name of the templated database must be changed on the Database Template screen. The other fields can be changed as needed. Dependent objects are not included in the template operation. Press F3 (End) when all changes have been made. A Confirmation screen will appear. Accept, edit, or reject the DDL to be used to create the databases.

Database Alter Option

The Database Alter screen performs all functions of the ALTER DATABASE statement. The alter statement changes the definition of a database and can be used to change all the parameters used initially to create the database, except for the Database Name and Creator ID.

Note: For more information about the fields on this screen, see the online help.

After you make changes, press PF3 and a confirmation screen appears where you can accept, edit, or reject the DDL.

Note: To alter or change a database name or creator ID, you can use RC/Alter or CA RC/Migrator. For more information, see the *CA RC/Migrator User Guide*.

When changing the database Roshare field from Owner to None or None to Owner, RC/Alter will generate the correct VSAM alters for any user defined tablespace or index. The alter statements change the share options parameter of the data set. This parameter must be SHAREOPTION(1,3) for shared objects.

Note: RC/Alter does not fully support a read to owner or owner to read alter in the Roshare field. Users must stop and start the databases. For more information about altering the ROSHARE option of a database, see the IBM *DB2 Administrative Guide, Volume 2*.

Temporary and Workfile Databases

To create or template a database for declared temporary tables, set the Database Create, Alter, or Template panel's AS option to TEMP.

You can create only one temporary database for each DB2 subsystem or data sharing member. A temporary database cannot be shared between DB2 subsystems or data sharing members.

A temporary database must have its CCSID option set to blanks.

Specifying the AS option to TEMP causes the AS TEMP clause to be inserted into the generated DDL.

To create or template a work file database, set the Database Create, Alter, or Template panel's AS option to Workfile. You can only do this in a data sharing environment and you can only create one work file data base for each DB2 member.

Database Drop Considerations

The Drop Database screen permits users to drop one or more databases.

Important! Use extreme caution in dropping databases because dropping a database deletes all other objects in the database. For more information about the various drop options (S, SO, R, RO), see the “Using CA RC/Update” chapter.

DDL Execution

When F3 (End) is pressed from the Create, Alter, Template or Drop screen, a confirmation screen appears before the DDL is executed. Accept, edit, or reject the DDL to be used to complete the request.

Note: For more information, see the “Using CA RC/Update” chapter.

If your changes are complex, require dropping and recreating objects (and their dependencies), or the current CA RC/Update Operation Mode is set to A (RC/Alter), you can perform a complete analysis of your changes that includes rebuilding the DDL and optionally including utility statements and extended DDL generation and analysis options.

Chapter 17: Tablespace

This section contains the following topics:

Overview of Tablespaces

A tablespace is a DB2 object consisting of VSAM ESDS data sets containing one or more DB2 tables. When the tablespace is created, its database and storage group are also designated. You can use the Tablespace option to create, (new or template), alter, and drop tablespaces.

Features for Maintaining Tablespaces

We provide the following sophisticated features for creating and maintaining DB2 tablespaces:

- Selection help for database, VCAT, and storage group. When defining a tablespace, you must specify the database in which the tablespace will reside, as well as a VCAT name or storage group for partitioned databases. Using selection lists, you can enter selection criteria to receive a list of objects from which you can select the database, VCAT, and storage group.
- Space Calculation Screen. A sophisticated space calculation facility is available to assist you in calculating space allocations for the tablespace. If the tablespace is partitioned, space calculations can be made on each partition. Once you have completed your space calculations, the information is passed back to the tablespace screen.

- Space Locator. The Space Locator facility helps you locate a volume or storage group with adequate free space for your tablespaces.
- VSAM Define Screen. If you specify VSAM data sets instead of specifying a storage group, you must specify the data set name and type and supply any required passwords. The VSAM Define screen will assist you in defining your VSAM data sets.
- RC/Alter Support. DB2 lets you make only certain changes to tablespaces. If the changes you want to make are not supported by DB2, we invoke RC/Alter support to drop and re-create the tablespace. All dependents, data, and authorizations are automatically restored and all changes are automatically propagated to any dependent object types. This extremely powerful feature is transparent to the end user. The user makes the request for the change, and the changes are made using the appropriate methods.

Accessing Tablespace Functions

Tablespace functions can be accessed by requesting to create, alter, template, or drop a tablespace. To create a tablespace, enter **C** in the Option field and **TS** in the Object field of the main header.

To alter, template, or drop a tablespace, enter **A**, **T**, or **D** in the Option field and **TS** in the Object field of the main header; or enter a tablespace name in the Item Name field and/or a user ID in the Creator field for a more specific selection list. The appropriate Tablespace screen appears.

Selection Panels

When altering, dropping, or creating (by template) a view, a selection panel is presented to select the view if not already specified. Make the view selection list more specific by specifying selection criteria for the Item (view) Name and Creator ID prompts in the header.

Use the EQF facility to further specify the views displayed, and use scrolling commands to scroll through information.

Note: For more information about EQF and scrolling commands, see the *CA Database Management Solutions for DB2 for z/OS General Facilities Reference Guide*.

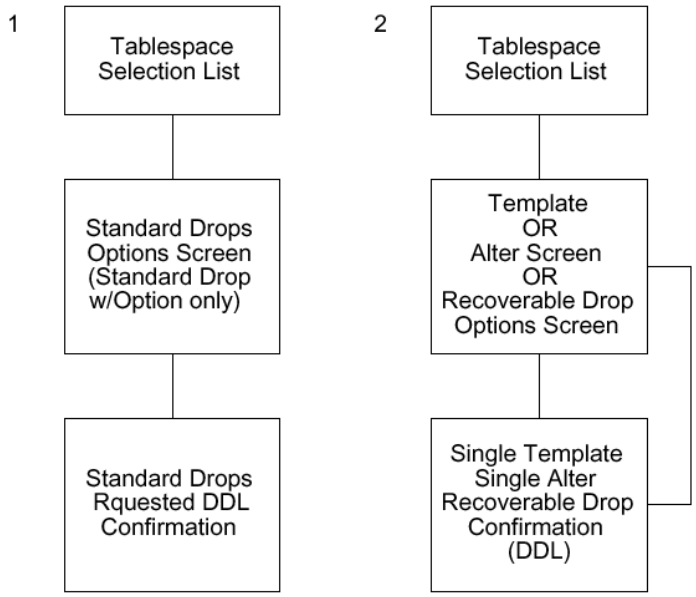
The View Selection screen displays all the fields available in the SYSIBM.SYSTABLES table. This information is available to make selection of the views easier.

To choose items from the listing, type **S** next to the items to select. Press Enter to register the selection.

Note: For information about the fields, press F1 (Help).

Selecting Tablespaces

To select a tablespace, enter **S** next to those to select. More than one tablespace can be selected at a time. For templates or alters, each selected tablespace will be processed individually in the order of the selection list. For standard drops, all tablespaces will be dropped together. Recoverable drops are processed one at a time. Both processes are summarized in the following diagram:



Part 1 of the diagram shows that all standard drops are processed together in a single batch of DDL. Accept or reject the batch of drops.

Part 2 shows that in templates or alters, each template or alter is processed individually. After confirming the DDL used to template or alter the tablespace, the user is returned to the Template or Alter screen for the tablespace next on the selection list. This continues until all selected objects have been processed.

Part 2 also shows that recoverable drops are processed individually.

Note: For information about screen flow for standard and recoverable drops, see the "Using CA RC/Update" chapter.

Tablespace Create

All the functions of the DB2 CREATE TABLESPACE command are provided. This statement allocates and formats a tablespace in which one or more tables can be created. A list of databases and storage groups appears to select objects for use. It will also help calculate primary and secondary space quantities through the Space Calculation screen, and locate volumes or storage groups with adequate space through the Space Locator. VSAM data sets can be defined using VSAM Define screen.

Once all Create information has been given, a confirmation screen will appear. This screen displays the actual DDL that will be used to create the tablespace. Accept, edit, or reject the Creation at this point.

Create a Simple Tablespace

A simple tablespace is a one that is neither segmented nor partitioned, and can contain many tables. They are not frequently used because the whole tablespace (instead of only one table) must be locked at once; the whole tablespace must be scanned to scan one table; and the space of a dropped table cannot be reused immediately. An advantage of a simple tablespace is that the user can control the order of the rows stored in a table.

Follow these steps:

1. Access the Tablespace Create screen.
2. Enter data in the following fields to define some basic information:
   - Table Space
   - Creator

   Other fields contain default specifications.

   Note: Defaults can be set through the DEF command for all fields except Tablespace Name and Partitioned. For more information, see the “Object Definition Defaults” chapter. For information about the fields, press F1 (Help).

3. Enter storage-related information about the tablespace in the Partition Information fields.

   Because this is a simple tablespace, only enter information on the first line, which represents the whole tablespace. A VCAT or storage group must be entered to override the storage group default (SYSDEFT). The rest of the information is optional.

   The following fields are optional:

   - VCAT
   - STOGROUP (defaults to SYSDEFLT)
   - PRIQTY (defaults to 3k)
   - SECQTY (defaults to 3k)
   - Erase (defaults to 0)
   - FRPAGE (defaults to 0)
   - %FR (defaults to 5)

   Note: Defaults can be set through the DEF command for all these fields except Erase.

4. Press F3 (End) to process the creation.

Create a Segmented Tablespace

A segmented tablespace is a non-partitioned tablespace that has been segmented. A segmented tablespace can contain more than one table; but each segment contains rows from only one table. Every segment in a segmented tablespace is the same size. A table will use only as many segments as it needs. Segmented tablespaces provide the following advantages over simple tablespaces.

- To scan a table, users only need to scan the segments containing data from that table.
- Only one table can be locked in the tablespace.
- The space of a dropped table can be reused immediately.

One drawback to a segmented tablespace is that some DB2 utilities, such as COPY, REORG or LOAD REPLACE operate on a tablespace or partition basis only.

Note: These are the same steps used to create a simple tablespace, except the segment size must be specified.

Follow these steps:

1. Access the Tablespace Create screen.
2. Enter data in the following fields to define some basic information:
   - Table Space
   - Creator

   Other fields contain default specifications.

   Note: Defaults can be set through the DEF command for all fields except Tablespace Name and Partitioned. For information about the fields, press F1 (Help).
3. Enter storage related information about the tablespace in the Partition Information fields.

   Because this is a segmented tablespace, you should only enter information on the first line. A VCAT or storage group must be entered to override the storage group default (SYSDEFT).

   Because the tablespace is not partitioned, the only valid line commands are V (VSAM Define), S (Space Calculation), and U (Undo any changes, set back to old definition).

   Note: Defaults can be set through the DEF command for all fields except Erase. For more information, see the “Log Display” chapter.
4. Press F3 (End) to process the creation.

Create a Partitioned Tablespace

You can create a partitioned DB2 tablespace as described in this section.

Follow these steps:

1. Enter data in the following fields on the Tablespace Create screen to define basic information:

   - Table Space
   - Creator

   Other fields contain default specifications.

   Note: You can set defaults through the DEF command for all these fields except Table Space and Partitioned. For more information, see the “Object Definition Defaults” chapter.

   Your settings are applied to the tablespace.

2. Specify values in the Partition Information fields to define storage-related information about the tablespace, then press the END key to process the creation.

   Note: You can use standard ISPF line commands to create and manipulate partitions. You must enter a VCAT or storage group if you do not want to accept the storage group default (SYSDEFLT). You can set defaults through the DEF command for all of these fields except Erase. For more information, see the “Object Definition Defaults” chapter.

   Your tablespace is defined.

Create a Range-Partitioned Tablespace

In addition to being partitioned, a range-partitioned tablespace is also segmented.

Note: A range-partitioned tablespace can contain only one table, which must be a table-controlled partitioned table.

You can create a range-partitioned DB2 tablespace as described in this section.

Follow these steps:

1. Complete fields as follows on the Tablespace Alter screen:
   - Specify a name in the Table Space field.
   - Specify a creator in the Creator field.
   - Type **RANGE** in the TS Type field.
   - Specify a non-zero value in the Segment Size field.

     Note: Because a value exists that defines a segment size, you cannot specify YES for Mbr Cluster (which would indicate that inserted data should not be clustered by a clustering index).

   Note: You can use the DEF command to set defaults for all fields except Table Space, Partitioned, and Erase. For more information, see the “Object Definition Defaults” chapter.

   Press Enter.

   Your settings are applied to the tablespace. If a value was not specified for Lockmax, a message appears indicating that a default value was created.

2. Specify values in the Partition Information fields to define storage-related information about the tablespace, then press the END key to process the creation.

   Note: You can use standard ISPF line commands to create and manipulate partitions. You must enter a VCAT or storage group if you do not want to accept the storage group default (SYSDEFLT). You can set defaults through the DEF command for all of these fields except Erase. For more information, see the “Object Definition Defaults” chapter.

   Your tablespace is defined.

Create a Partition-by-Growth Tablespace

A partition-by-growth (PBG) tablespace is a universal tablespace whose size can grow to accommodate data growth. This type of tablespace starts with one partition, and DB2 adds partitions as needed. (A universal tablespace is partitioned and segmented.)

Note: Partitions *cannot* be explicitly added, rotated, or altered. The tablespace *must* be DB2-managed (not user-managed) so that DB2 can create data sets as partitions become full.

A PBG tablespace has no limit keys. All partition data sets share the same partition information.

Follow these steps:

1. Access the Tablespace Create panel by completing the Option field and Object field on the CA RC/Update main menu.

   The Tablespace Create panel appears.

2. Complete the following fields:

   - Type **GROWTH** in the TS Type field.

     Note: Do *not* specify GROWTH for a tablespace that is in a work file database.

   - Type **YES** in the Partitions field.
   - Specify a name in the Table Space field.
   - Specify a creator in the Creator field.
   - (Optional) Specify a non-zero value in the Segment Size field, which determines how many pages to assign to each tablespace segment.

     Note: If you do not define a value, Segment Size defaults to 4.

   Note: You can use the DEF command to set defaults for all fields except Table Space, Partitions, and Erase. For more information, see the “Object Definition Defaults” chapter.

   Press Enter.

   Your settings are applied to the tablespace. The panel now contains the Max Parts field, which controls the maximum number of partitions to which a PBG tablespace can grow.

3. Complete the following fields for defining partition characteristics of the PBG tablespace:

   Max Parts

   Specifies the maximum number of partitions to which a PBG universal tablespace is allowed to grow. The value must be an integer in the range of 1 to 4096, depending on the corresponding value of DSSIZE and Buffer Pool.

   Note: For information about the maximum allowable value for Max Parts in relation to the page size and DSSIZE value for the tablespace, see the IBM *DB2 for z/OS SQL Reference*.

   Any unacceptable value specified for Max Parts is automatically adjusted downward to the nearest acceptable value. When DSSIZE is blank, 4 (GB) is assumed.

   DSSIZE

   Specifies a value, in gigabytes, that indicates the maximum size for each partition (or for each data set for LOB tablespaces). Valid values are 1, 2, 4, 8, 16, 32, and 64.

   Note: When DSSIZE is blank, 4 (GB) is assumed in determining the maximum value for Max Parts.

   Buffer Pool

   Specifies the name of the tablespace's buffer pool, which also determines the page size of the tablespace. For 4 KB, 8 KB, 16 KB, and 32 KB page buffer pools, the page sizes are 4 KB, 8 KB, 16 KB, and 32 KB, respectively.

   Note: The specified name must identify an activated buffer pool. For an explanation of valid names, see the online help.

   You may also specify DEFAULT, indicating that you want DB2 to use the default buffer pool of the database for your tablespace.

   Press Enter.

   Partition information is configured for your PBG tablespace.

   Note: VCAT is *not* allowed for a PBG tablespace.

4. Press the END key.

   The Creation Confirmation panel appears.

5. Press Enter, and enter **Y** on the Execution Confirmation prompt to initiate creation.

   The AUDIT Message File appears, with details about the creation and a description of any problems that occurred.

   Your tablespace is defined.

Tablespace Create Screen

The Tablespace Create screen is where users supply the information necessary for tablespace creation.

The fields below the header control the information on the Tablespace Create screen.

Note: For information about the fields, press F1 (Help).

Partition Information Fields

The Partition Information Fields let you enter storage related information about each partition in the tablespace or the whole tablespace (non-partitioned).

Effect of Changing Tablespace's Partitioned Field

If partition information is entered, and then the tablespace is changed to non-partitioned, all lines except the first are deleted. If the tablespace is changed back to a partitioned tablespace, the old partition values are returned.

Note: For information about the fields, press F1 (Help).

Group Bufferpool Cache Setting for Tablespace Partitions

When you issue the G line command beside a partition on a tablespace create, alter, or template panel, the Group Bufferpool Cache Setting panel appears.

To specify what pages are written to the group buffer pool in a data sharing environment, enter one of the following values in the GBP Cache for partition field:

A (GBP CACHE ALL)

Caches both changed and unchanged pages in the group buffer pool.

N (GBP CACHE NONE)

Caches no pages in the group buffer pool. DB2 uses the group buffer pool only for cross-invalidation. If you specify this option, the tablespace or partition must not be in rebuild pending status and must be in the dropped state.

S (GBP CACHE SYSTEM)

Caches only the changed system pages within the LOB tablespace in the group buffer pool. A system page is a space map page or any other page that does not contain actual data values. This is valid only for a LOB tablespace.

C (GBP CACHE CHANGED)

Caches only changed pages in the group buffer pool. This is the default.

Processing Considerations for Tablespace Creation

When you are creating a tablespace, a valid storage group and database can be used. DB2 will allow the definition of a storage group and database using invalid values. If an attempt is made to create a tablespace that references invalid storage groups or databases, an SQL error will result.

Tablespace Template

The template option allows creation of a new tablespace using an existing tablespace as a template. In fact, a template session is actually a create session with an additional step; a tablespace can be selected as a starting point. The other fields can be changed, but the name of the templated tablespace must be changed. The tablespace's dependent objects are not included in the template operation.

The Tablespace Template screen displays the values used to create the template tablespace. The fields for this screen are similar to those on the Tablespace Create screen. Change the fields for the new tablespace according to the procedures outlined in the Tablespace Create. Once changes have been made, press F3 (End) to process the template.

LC Line Command

The LC line command lets you browse the LISTC Report and view space amounts for user-defined data sets. Use this command from the Tablespace Alter or Tablespace Template screen. Enter **LC** in the CMD field of the appropriate data set, and the IDCAMS System Services LISTC Report appears.

Note: For more information about the specifics of this report, see the IBM IDCAMS documentation.

Confirming the Template

A Confirmation screen will appear. Accept, edit, or reject the DDL to be used to create the tablespace.

Tablespace Alter

The DB2 ALTER TABLESPACE command is fully supported.

Note: The Primary (PQTY) and Secondary (SQTY) Quantity alterations are done through RC/Alter even though a DB2 ALTER could partially do them. If an attempt is made to change these quantities with a DB2 ALTER statement, the change will not be complete in any usable way until a REORG or DROP/CREATE is performed. For this reason, PQTY and SQTY are not included in the previous list.

If the requested changes can be made via DB2 ALTER TABLESPACE statements, a confirmation screen appears. Accept, edit, or reject the DDL to be used to make the alterations.

If the requested changes cannot be made through a DB2 ALTER TABLESPACE statement, RC/Alter must be used. RC/Alter must be used if the tablespace name, database, partition information, segment information are changed.

These changes are made by dropping and recreating the tablespace. All dependents, data, and authorizations are automatically restored and all changes are automatically propagated to any dependent object types.

If the tablespace must be dropped and re-created, the message Change Requires Drop/Recreate appears when F3 (End) is pressed to process the screen. This means that RC/Alter must be used to alter the tablespace, and the RC/Alter Specification Screen displays. Provide the necessary information about the data set to which the DDL will be written. The DDL must then be executed in batch mode.

Note: For more information, see the “Using CA RC/Update” chapter.

The Tablespace Alter screen also allows changes to multiple partitions on one screen without having to issue separate ALTER TABLESPACE statements. To change the Free Page or Percent Free for partitions in DB2, a separate ALTER TABLESPACE command must be issued for each partition.

More information:

DDL Execution (see page 60)

Change Tablespace Type from Simple to Segmented

You can change the tablespace type from simple to segmented by changing the value of the Segment Size field on the Tablespace Alter screen.

To change tablespace type from simple to segmented, change the Segment Size field from 0 to segment size.

Change Tablespace Type from Simple to Partitioned

You can change the tablespace type from simple to partitioned as described in this section.

Note: After performing the conversion, you need to alter the index on the table that exists in the tablespace. If the table in the tablespace does not have an index, you need to create the index. For more information about creating the index, see the “Index” chapter.

Follow these steps:

1. Enter **YES** in the Partitioned field on the Tablespace Alter screen.

   Your settings are applied to the tablespace. A PART field appears in the Partition Information portion of the screen.

2. (Optional) Perform the following steps as many times as needed to produce additional partitions:

   a. Enter **R** in the CMD line next to a partition to repeat the partition.

   b. Change the PRIQTY, SECQTY, Freepage, and PCTfree fields for the partition, if necessary, then press Enter.

   The partition is added according to your specifications.

Change Tablespace Type from Simple to Range-Partitioned

You can change the tablespace type from simple to range-partitioned as described in this section.

Follow these steps:

1. Complete fields on the Tablespace Alter screen as follows:
   a. Change the value in the Partitioned field from NO to **YES**.
   b. Change the value in the TS Type field to **RANGE**.
   c. Specify a non-zero value in the Segment Size field.

   Press Enter.

   Your settings are applied to the tablespace. A PART field appears in the Partition Information portion of the screen.
2. (Optional) Perform the following steps as many times as needed to produce additional partitions:
   a. Enter **R** in the CMD line next to a partition to repeat the partition.
   b. Change the PRIQTY, SECQTY, Freepage, and PCTfree fields for the partition, if necessary, then press Enter.

      The partition is added according to your specifications.

More information:

Convert a Non-Partitioned Table to a TCP Table (see page 326)

Change Tablespace Type from Partitioned to Range-Partitioned

To change the tablespace type from partitioned to range-partitioned, complete fields as follows on the Tablespace Alter screen:

- Change the value in the TS Type field to **RANGE**.
- Specify a non-zero value in the Segment Size field.

Press Enter.

Your settings are applied to the tablespace.

Note: If an index-controlled partitioned table exists in the tablespace, the table is automatically converted to table-controlled partitioning.

Change Tablespace Type from Segmented to Partitioned

You can change the tablespace type from segmented to partitioned as described in this section.

Note: After performing the conversion, you need to alter the index on the table that exists in the tablespace. If the table in the tablespace does not have an index, you need to create the index. For more information about creating the index, see the “Index” chapter.

Follow these steps:

1. Complete fields on the Tablespace Alter screen as follows:
   a. Change the value in the Partitioned field from NO to **YES**.
   b. Change the value in the Segment Size field to **0**.

   Press Enter.

   Your settings are applied to the tablespace.
2. (Optional) Perform the following steps as many times as needed to produce additional partitions:
   a. Enter **R** in the CMD line next to a partition to repeat the partition.
   b. Change the PRIQTY, SECQTY, Freepage, and PCTfree fields for the partition, if necessary, then press Enter.

   The partition is added according to your specifications.

More information:

Alter the Index for a Table in a Partitioned Tablespace (see page 325)

Change Tablespace Type from Segmented to Range-Partitioned

You can change the tablespace type from segmented to range-partitioned as described in this section.

Follow these steps:

1. Complete fields as follows on the Tablespace Alter screen:
   a. Change the value in the Partitioned field from NO to **YES**.
   b. Change the value in the TS Type field to **RANGE**.

   Press Enter.

   Your settings are applied to the tablespace.
2. (Optional) Perform the following steps as many times as needed to produce additional partitions:
   a. Enter **R** in the CMD line next to a partition to repeat the partition.
   b. Change the PRIQTY, SECQTY, Freepage, and PCTfree fields for the partition, if necessary, then press Enter.

   The partition is added according to your specifications.

More information:

Convert a Non-Partitioned Table to a TCP Table (see page 326)

Alter the Index for a Table in a Partitioned Tablespace

When you perform any of the following conversions, you also need to alter the index on the table that exists in the tablespace:

- Changing the tablespace from simple to partitioned
- Changing the tablespace from segmented to partitioned

Note: If the table in the tablespace does not have an index, you must create the index. For more information, see the “Index” chapter. For more information about configuring key columns and limit keys, see the "Table" chapter.

Follow these steps:

1. Enter **YES** in the Partitioned field on the Table Alter screen.

   The Table Partitioning Key Col Selection & Maint screen appears.

2. Set up your key columns according to instructions on the screen, then press the END key.

   The table is converted to a partitioned table, and the Table Alter screen appears.

3. Enter **LIMITS** on the command line.

   The Table Partitioning & Limit Key Values screen appears.

4. Specify limit keys for each partition, then press the END key.

   The index is altered.

More information:

Change Tablespace Type from Segmented to Partitioned (see page 324)
Change Tablespace Type from Simple to Partitioned (see page 322)

Convert a Non-Partitioned Table to a TCP Table

When a tablespace is changed to be range-partitioned, CA RC/Migrator converts any existing index-controlled partitioned table within the tablespace to a table-controlled partitioned (TCP) table. However, simple and segmented tablespaces do *not* contain a partitioned table that can be converted. Therefore, you must convert the existing non-partitioned table to a TCP table after performing either of the following tasks:

- Changing a tablespace from simple to range-partitioned
- Changing a tablespace from segmented to range-partitioned

To convert a non-partitioned table to a TCP table

1. Enter **YES** in the Partitioning field on the Table Alter panel.

   Note: For more information about accessing table functions and panels, see the "Table" chapter.

   The Table Partitioning Key Col Selection & Maint screen appears. You must define at least one partitioning key column.

2. Type **S** beside the name of each column that you want to include in the table's partitioning key, then press Enter.

   The selected columns are inserted into the key.

   Note: You can use line commands to arrange the columns in the key if needed.

3. Press the END key.

   The Table Alter panel appears.

4. Press the END key.

   The Table Partitioning & Limit Key Values panel appears.

5. Enter a limit value for the first key column of each partition.

   Any values you entered are now reflected on the panel.

6. (Optional) Specify additional limit values, then press the END key.

   The Alteration Strategy Services panel appears.

7. Press the END key.

   CA RC/Migrator saves the changes to your strategy to convert the non-partitioned table to a TCP table.

More information:

Change Tablespace Type from Segmented to Range-Partitioned (see page 325)
Change Tablespace Type from Simple to Range-Partitioned (see page 323)

Change Tablespace Type from Partitioned or Segmented to Partition-by-Growth

You can change the tablespace type from partitioned or segmented to partition-by-growth (PBG).

Follow these steps:

1. Access the Tablespace Alter panel by doing one of the following:

   - Complete the fields on the CA RC/Update Main Menu.
   - Access tablespace functions from an alteration strategy, migration strategy, or comparison strategy in CA RC/Migrator.

   The Tablespace Alter panel appears.

2. Complete the following fields:

   - Type **GROWTH** in the TS Type field.
   - Enter a non-zero value in the Segment Size field.

     Note: If you do not define a value, Segment Size defaults to 4.

   Press Enter.

   Your settings are applied to the tablespace. The panel now contains the Max Parts field, which controls the maximum number of partitions to which a PBG tablespace can grow.

   Note: If an index-controlled partitioned table exists in the tablespace, the table is automatically converted to table-controlled partitioning.

3. Complete the following fields for defining partition characteristics of the PBG tablespace:

   Max Parts

   Specifies the maximum number of partitions to which a PBG universal tablespace is allowed to grow. The value must be an integer in the range of 1 to 4096, depending on the corresponding value of DSSIZE and Buffer Pool.

   Note: For information about the maximum allowable value for Max Parts in relation to the page size and DSSIZE value for the tablespace, see the IBM *DB2 for z/OS SQL Reference*.

   Any unacceptable value specified for Max Parts is automatically adjusted downward to the nearest acceptable value. When DSSIZE is blank, 4 (GB) is assumed.

   DSSIZE

   Specifies a value, in gigabytes, that indicates the maximum size for each partition (or for each data set for LOB tablespaces). Valid values are 1, 2, 4, 8, 16, 32, and 64.

   Note: When DSSIZE is blank, 4 (GB) is assumed in determining the maximum value for Max Parts.

   Buffer Pool

   Specifies the name of the tablespace's buffer pool, which also determines the page size of the tablespace. For 4 KB, 8 KB, 16 KB, and 32 KB page buffer pools, the page sizes are 4 KB, 8 KB, 16 KB, and 32 KB, respectively.

   Note: The specified name must identify an activated buffer pool. For an explanation of valid names, see the online help.

   You may also specify DEFAULT, indicating that you want DB2 to use the default buffer pool of the database for your tablespace.

   Press Enter.

   Partition information is configured for your PBG tablespace.

4. Press the END key.

   The Alteration Analysis panel appears.

5. Complete the fields on the Alteration Analysis panel, and press Enter.

   Alteration analysis commences.

Change Tablespace Type from Simple to Partition-by-Growth

You cannot create simple tablespaces in DB2 9 and above. However, simple tablespaces that you created with an earlier DB2 version are still supported.

You can change the tablespace type from simple to partition-by-growth as described in this section.

Follow these steps:

1. Access the Tablespace Alter panel by doing one of the following:

   - Complete the fields on the CA RC/Update Main Menu.
   - Complete fields on a panel from an alteration strategy, migration strategy, or comparison strategy in CA RC/Migrator.

   The Tablespace Alter panel appears.

2. Complete fields as follows:

   a. Change the value in the Partitions field from NO to **YES**.

   b. Change the value in the TS Type field to **GROWTH**.

   c. Specify a non-zero value in the Segment Size field.

   Press Enter.

   Your settings are applied to the tablespace. The panel now contains the Max Parts field, which controls the maximum number of partitions to which a PBG tablespace can grow.

3. Complete the following fields for defining partition characteristics of the PBG tablespace:

   Max Parts

   Specifies the maximum number of partitions to which a PBG universal tablespace is allowed to grow. The value must be an integer in the range of 1 to 4096, depending on the corresponding value of DSSIZE and Buffer Pool.

   Note: For information about the maximum allowable value for Max Parts in relation to the page size and DSSIZE value for the tablespace, see the IBM *DB2 for z/OS SQL Reference*.

   Any unacceptable value specified for Max Parts is automatically adjusted downward to the nearest acceptable value. When DSSIZE is blank, 4 (GB) is assumed.

   DSSIZE

   Specifies a value, in gigabytes, that indicates the maximum size for each partition (or for each data set for LOB tablespaces). Valid values are 1, 2, 4, 8, 16, 32, and 64.

   Note: When DSSIZE is blank, 4 (GB) is assumed in determining the maximum value for Max Parts.

   Buffer Pool

   Specifies the name of the tablespace's buffer pool, which also determines the page size of the tablespace. For 4 KB, 8 KB, 16 KB, and 32 KB page buffer pools, the page sizes are 4 KB, 8 KB, 16 KB, and 32 KB, respectively.

   Note: The specified name must identify an activated buffer pool. For an explanation of valid names, see the online help.

   You may also specify DEFAULT, indicating that you want DB2 to use the default buffer pool of the database for your tablespace.

   Press Enter.

   Partition information is configured for your PBG tablespace.

4. Press the END key.

   A prompt appears for you to choose whether to create a partitioned index.

5. Perform one of the following:

   - Enter **Y** to create a partitioned index, and complete the fields on the Index Create panel.
   - Enter **N** to decline partitioned index creation.

6. Complete the fields on the Alteration Analysis panel, and press the END key.

   Alteration analysis commences.

More information:

Convert a Non-Partitioned Table to a TCP Table (see page 326)

Tablespace Alter Screen

The Tablespace Alter screen allows changes to the values used to create the tablespace. It is similar to the Tablespace Create and Tablespace Template screens. If partition information is entered, change the tablespace to non-partitioned. All lines except the first are deleted. If the tablespace is changed back to a partitioned tablespace, the old partition values are returned.

Edit the information as in a Tablespace Create screen. Once the changes have been made, press F3 (End) to process the alter.

Processing Considerations for Tablespace Alterations

Note the following processing considerations:

- A change to the CLOSE Rule is effective as soon as the DDL is executed.
- A change to the BUFFER POOL is effective the next time the tablespace's data sets are opened.
- A change to the LOCKSIZE will only apply to SQL statements that will be executed later. It has no effect on currently executing SQL statements. Application plans are only updated if they are rebound.
- A change to the FREEPAGE or PCTFREE values will take effect when records are loaded into the tablespace or the tablespace is reorganized.

When the changes are being made through the DB2 ALTER TABLESPACE, CA RC/Update takes the following steps to make the alter:

1. The tablespace is stopped.
2. The ALTER statement is issued.
3. The tablespace is started in UT (utility) mode.
4. The REORG statement is issued.
5. The tablespace is started in RW (read-write) mode.

LC Line Command

The LC line command lets you browse the LISTC Report and view space amounts for user-defined data sets. Use this command from the Tablespace Alter or Tablespace Template screen. Enter **LC** in the CMD field of the appropriate data set, and the IDCAMS System Services LISTC Report appears.

Note: For more information about the specifics of this report, see the IBM IDCAMS documentation.

Confirming the Alter

After all information has been entered, press F3 (End). Depending on the type of alter performed, one of the following results occurs:

- If the changes can be made with a standard DB2 ALTER TABLE command, a confirmation screen appears, from which you can perform any of the following actions:
  - Accept the generated DDL.
  - Reject the generated DDL.
  - Make modifications to the generated DDL before accepting it.
- If the changes require an CA RC/Migrator analysis or if the current operation mode of CA RC/Update is A, the RC/Alter screen appears, from which you can perform one of the following actions:
  - Enter the required information, press Enter, and then use the Batch Processor (BP) facility to execute the generated DDL.
  - Enter END to return to the Table Alter screen without invoking RC/Alter.
  - Enter CANCEL to exit RC/Alter and discard your changes.

  Note: If a CA RC/Migrator analysis is required to generate the DDL necessary to make your changes, an O or B Operation Mode setting is ignored.

  The analysis phase generates all statements that are necessary to alter the table or to drop and re-create the table, data, dependents, and authorizations.

For complete information, see DDL Execution in the “Using CA RC/Update” chapter.

Calculate Space Requirements for Tablespaces

The Space Calculator facility can calculate the primary and secondary space quantities needed for your tablespaces and eliminates time-consuming calculations. This is especially important for users who want to calculate the space needed for segmented tablespaces and for multiple tables within one tablespace.

Multiple table calculations require an average row size to calculate accurately the total space needed for non-segmented tablespaces. The average row size approximates multiple tables' row length to compensate for the rows being interleaved on one page.

You can create a new stand-alone tablespace space strategy that saves space requirements for a tablespace or edit an existing space strategy as needed. If the tablespace is partitioned, space calculations can be made on each partition.

Note: For more information about using the Space Calculator, see the *General Facilities Reference Guide*.

Follow these steps:

1. Do *one* of the following:

   - Type **SS** (Space Calculator) in the Option field on the CA Database Management Solutions for DB2 for z/OS main menu and press Enter.
   - Type **SSC** on the command line from any product and press Enter.

   The SpaceCalc Strategy Select panel appears.

2. Do *one* of the following:

   - Type **CT**, the name of the strategy, and the name of the tablespace on the create line (first) to create a new stand-alone tablespace strategy.
   - Press Enter to see a list of existing strategies and then type **E** next to a strategy you want to edit.

   Press Enter.

   The Tablespace Strategy panel appears.

   Note: This panel has the same fields as the Tablespace Alter, Tablespace Create, and Tablespace Template panels that can be accessed from CA RC/Migrator and CA RC/Update.

3. Type **S** next to a tablespace and specify a VCAT or storage group name in the VCAT or Stogroup fields, and press Enter.

   The Space Calculation panel appears. The name of the tablespace and the number of the partition appear on the first line. The control parameters default to the values last used on the Tablespace Create and Table Alter panels.

4. Edit the control parameters and the table information fields as needed and press Enter.

   Note: All fields must contain a value before the Space Calculator can calculate the results.

   The space requirements for the tablespace or partition are calculated and the Space Calculator fills in the columns for each table included in the tablespace.

5. (Optional) Type **E** (explode) next to a table or multiple tables to view detailed information about the selected tables and press Enter.

   Detailed statistics for the selected table or tables appear on the Space Calculation panel. Multiple commands are processed in the order they appear on the Space Calculation panel.

   Note: The statistics shown are applicable only for the selected table.

6. Type **T** (totals) on the command line and press Enter.

   The calculated results for the tablespace appear in the Space Statistics section on the Space Calculation panel.

   Note: Totals for segmented tablespaces are calculated based on the minimum or maximum number of segments needed for the given number of rows, cylinders, or tracks as specified in the Qty Type field on the Space Calculation panel. The totals for non-segmented tablespaces are calculated based on the average row size.

7. Press F3 (End) when you have completed your calculations.

   The calculations are used for the selected tablespace or partition.

   If you accessed the Space Calculator from CA RC/Migrator or CA RC/Update, you are returned to the tablespace panel you were at in the product. The calculations are updated on the panel and the information is used for the selected tablespace or partition.

   Note: You cannot change the values of an actual catalog object or the associated actualized space strategy. If you make and save changes to the statistics, an object-linked space strategy is created.

8. Type **S** and an eight-character strategy name in the SSTRAT field to save the strategy and press Enter.

   Note: You can enter the name of a new strategy or reference an existing (previously saved) strategy. When a new space strategy is created, the strategy name or object link name is put into this field.

   The SpaceCalc Strategy Save panel appears.

9. Press Enter to save the strategy with the specified attributes.

   The strategy is saved, the action code of the SSTRAT field is reset to N, and you are returned to the Space Calculation panel.

Locate Free Space

You can use the Space Calculator to locate volumes and storage group with adequate free space for your tablespaces and indexspaces. For example, you can request a list of volumes, determine which volumes have available space, and create a storage group using those volumes. If you execute most jobs in batch, ensuring you have enough space available before job execution helps to ensure that the job does not execute unsuccessfully because of space limitations.

Use the FIT command to toggle between showing all available volumes or storage groups or only those with adequate free space available.

Note: For more information about the Space Calculator, see the *General Facilities Reference Guide*.

Follow these steps:

1. Edit an existing strategy or perform a tablespace or indexspace space calculation.

   The Space Calculation panel for the tablespace or indexspace appears.

   The total space requirements needed for the tablespace or indexspace appear in the Space Statistics section. This information is used to determine which storage groups or volumes have available space.

2. Type **V** (volume) or **S** (storage group) and an object name or selection criteria in the Volume field and press Enter.

   The Space Calculation panel lists and highlights the storage groups or volumes that match your selection criteria. The specified volume serial IDs (volsers) in the storage groups that contain the space amounts are also shown.

   Note: Note the track largest and cylinders largest field values to ensure you select a storage group that can accommodate the space you need allocated. SMS in the Volser field indicates that this is an SMS-managed storage group. If the device type is N/A and the remaining values are zero (0), this indicates that the volumes defined in the storage group do not exist, although they might have existed at the time the storage group was created.

3. (Optional) Type **FIT** on the command line and press Enter.

   The Space Calculation panel lists only the storage groups or volumes that have adequate free space available. You can scroll through the list vertically and horizontally.

4. Review the following Free Space Verification fields and type **S** next to a storage group or volume to select it. You can select one storage group and up to six volumes.

STOGROUP

Identifies the name of the storage group that matches your selection criteria entered in the Volume field or Name field. The names that are highlighted have enough free space to satisfy the space calculator request. (This field displays only when stogroups are selected.)

VOLSER

Identifies the volume serial ID matching the selection criteria you entered in the Volume field or Name field or associated with the specified storage group. Volsers containing available space are shown.

If SMS, the storage group is SMS-managed, rather than having volumes assigned to it.

UNIT TYPE

Identifies the device type of the pack (DASD device) associated with the Volser.

If N/A (not applicable), the storage group is SMS-managed or the volume shown in the Volser field does not exist, is offline, or is otherwise inoperative.

FREE EXTENTS

Identifies the number of contiguous free tracks or cylinders on a DASD device.

TRACKS (SIZE and LARGEST)

Identifies the total number of free tracks available and the largest number of free tracks that exist in exactly one free extent.

CYLINDERS (SIZE and LARGEST)

Identifies the total number of free cylinders available and the largest number of free cylinders that exist in exactly one free extent.

TOTAL TRACKS

Identifies the total number of tracks that exist on the device.

TRK/CYL

Identifies the number of tracks per cylinder for the device.

PERCENT USED

Identifies the percentage of space that is currently allocated.

FREE DSCBS

Identifies the number of free data set control blocks (DSCB) that exist in the volume table of contents (VTOC). A DSCB is a label of the disk data set. This field is useful when you want to confirm how many records are in the VTOC. If the VTOC is full, the space cannot be allocated.

PUBLIC/PRIVATE

Identifies the mount attribute of the volume. Valid values are PUBLIC, PRIVATE, or STORAGE.

SMS

Identifies whether the volume is managed by IBM System Managed Storage.

Press Enter.

The Space Calculation panel reappears and shows the name of the storage group or volume located with free space in the Volume field. The Action value is reset to N. If more than one volume is selected, a plus sign (+) appears next to the name.

Note: Volumes are temporarily removed from the panel if selected and placed in a queue for processing. They are used to automatically generate the appropriate VSAM define cards. You can use the S (SHRINK) command to view the selected volume queue and update as needed. When you process the create, alter, or template request from CA RC/Migrator or CA RC/Update, the selected volumes are included in the DDL to create or alter the tablespace or index.

VSAM Cluster Definitions

Define the VSAM cluster by accessing the VSAM Define screen. Use the Space Calculation screen before defining the VSAM data sets, as the space statistics calculated are saved and brought to this screen. Any volser selections made from the Volume Space Locator will be used to define the VSAM cluster.

To define a VSAM cluster, enter **V** (VSAM) in the line command area next to the partition or tablespace whose VSAM cluster should be defined. The VSAM Define screen appears.

Note: For information about the fields, press F1 (Help).

Information for the unit type, primary and secondary space allocation, and volume names must be entered in the Space Allocation fields. If space information has been calculated through the Space Calculation screen, the information is passed to this screen and converted to the proper unit for display.

If the VSAM Catalog being used requires the use of VSAM passwords, enter them in the Passwords fields.

Once all necessary information has been entered, press F3 (End) to process the information and return to the tablespace Create, Template, or Alter screen.

LOB Tablespaces

A LOB tablespace is used for holding an auxiliary table, which stores LOB data for a LOB column. If a column is defined as LOB in a table, it requires its own auxiliary tablespace and index.

The LOB Tablespace screens let you alter, create, and template DB2 LOB tablespaces. These screens appear when you specify LOB in the TS Type field on the Tablespace Create, Alter, or Template screens.

Note: On these screens, Database shows the name of the database in which the tablespace is created. This is a required field. For LOB tablespaces, the database must be the same database in which the base table and its tablespace resides.

The LOB Tablespace screens let you alter, create, and template DB2 LOB tablespaces. As with regular tablespaces, you can:

- Apply user defaults to the index definition using the Apply Def command.
- Display the original and current definitions of an object using the Compare command.
- Toggle the header on and off using the HEADER command.
- Enter selection criteria to bring up lists of objects to choose from.

  Note: For more information about these fields, see the online help.
- Create an explicit VSAM data set for the tablespace through the VSAM definition facility.
- Estimate the space required by a tablespace using the space calculation feature. If the tablespace is partitioned, space calculations can be made on each partition.

Tablespace Drop Considerations

The Drop Tablespace option enables users to drop one or more tablespaces. Select the tablespaces to drop from a selection list. Once selected, accept, edit, or reject the DDL to be used to make the drop. If a tablespace is dropped, all tables within the tablespace and all dependent objects are also dropped. A tablespace cannot be dropped if it is being used by a DB2 utility.

Note: For information about how to drop an object with the various drop options (S, SO, R, RO), see the “Using CA RC/Update” chapter.

DDL Execution

When F3 (End) is pressed from the Create, Alter, Template or Drop screen, a confirmation screen appears before the DDL is executed. Accept, edit, or reject the DDL to be used to complete the request.

Note: For more information, see the “Using CA RC/Update” chapter.

If your changes are complex, require dropping and recreating objects (and their dependencies), or the current CA RC/Update Operation Mode is set to A (RC/Alter), you can perform a complete analysis of your changes that includes rebuilding the DDL and optionally including utility statements and extended DDL generation and analysis options.

Confirmation Options

A Confirmation screen appears before the DDL is executed.

Press Enter to execute the displayed DDL (online mode), write it to a data set (batch mode), or pass to RC/Alter and CA RC/Migrator to control DDL statement generation (RC/Alter mode). During the online DDL execution, the Audit Message File appears. This screen displays completion messages and return codes. If there are errors, all statements are rolled back. Use the scrolling keys to review all messages. Press F3 (End) or Enter to return to the previous screen.

Enter **E** (Edit) to edit the generated DDL statements using the CA RC/Migrator DDL Edit Facility.

To refrain from processing the DDL, press F3 (End). The previous screen reappears, where changes can be made to the request. Enter the CANCEL command to cancel the option and return to the Main Menu or process the next items for Alter and Template options.

Chapter 18: Table

This section contains the following topics:

Overview of Tables

You can use the table object options to create, template, alter, and drop DB2 tables. A DB2 table is a collection of rows all having the same columns. All data in DB2 is stored in tables, including the system catalog information. A DB2 table is the main object type within DB2.

Features for Maintaining Tables

The following functionality is provided for creating and maintaining DB2 tables:

- DB2 Object Selection Help

  When defining a table, the object names of the table's database and tablespace must be entered. Instead of entering a specific object name, selection criteria can be entered, from which a list of objects appears.

- Column Selection Help

  You can retrieve other table columns that have already been defined into the current table definition. This lets you create standard column definitions that can be copied into other tables.

- Row Size Indicator

  Automatically displays the effects different data type decisions will have on the total row size as a table is created or changed.

- Column Editor

  When defining columns, any of the standard ISPF editor commands can be used to copy, replicate, delete, and move columns within the table definition. Four-way ISPF scrolling is supported for the table column display and split screen mode is fully supported. A help screen is available for selecting the column data types.

- Column Explode

  Instead of scrolling horizontally to view all the attributes of a column, you can use the column explode option. This option shows all column attributes on one screen.

- Referential Integrity

  You can create unique constraints as part of the table create or table alter process. Table and column selection panels make selecting the right columns easy, and let you specify the sequence number for each column.

- Table-Controlled Partitioning

  You can fully exploit table-controlled partitioning when creating, altering, and dropping DB2 objects.

- Alteration Support

  DB2 permits only certain alterations to a DB2 table using the DB2 ALTER command. If the alterations to a table cannot be performed using the DB2 ALTER command, you can use the Alter Table Option to automatically invoke RC/Alter to perform a drop and recreation of the table with the changes. All dependents, data, and authorizations are automatically restored and all changes, including the impact on any indexes using columns, are automatically propagated to any dependent object types. This is a very powerful feature and is transparent to the end user. When the end user makes the change, we will make the change using the appropriate methods.

- COMPARE Command

  While modifying a table, you can display a summary of all changes by entering the COMPARE command. A new screen appears showing the old and new versions of the table, and the associated changes. Use the RESET command to reset the changed column back to its original values.

- CURRENT SQLID Support

  Create tables for other users using the SET CURRENT SQLID command to perform the necessary ID switching.

Note: The term “table screen” is used to refer to the general layout of the Table Create screen, Table Template screen, and Table Alter screen. All these screens share the same fields and layout, but the available options are different depending on the mode (create, alter, template).

Access Table Functions

To create, alter, template, or drop a table, do one of the following:

- Type **C** in the Option field and type **T** in the Object field of the main header, then press Enter to create a table.
- Type **A**, **T**, or **D** in the Option field and type **T** in the Object field of the main header, then press Enter to alter, template, or drop a table.

  You can enter a table name in the Item Name field and a user ID in the Creator field to get a more specific selection list. The appropriate table screen appears.

Selecting Tables

When altering, dropping, or creating (by template) a table, a selection panel is presented for selecting the tables if not already specified. The Table selection panel displays all the fields that are available in SYSIBM.SYSTABLES.

To change the list, enter selection criteria in the header fields. You can also use the EQF facility to further specify the tables displayed.

Note: For more information about EQF, see the *CA Database Management Solutions for DB2 for z/OS General Facilities Reference Guide*.

Editing or Browsing Table Data

You can edit or browse table data in views using RC/Edit.

Table Create

The Table Create screen automatically generates all statements associated with the SQL CREATE TABLE statement including:

- CREATE TABLE
- COMMENT ON TABLE
- COMMENT ON COLUMN
- LABEL ON TABLE
- LABEL ON COLUMN

From the Table Create screen, you can:

- Use object selection lists to retrieve the name of the database or tablespace.
- Create auxiliary tables for LOB columns by specifying the AUXILIARY option in the Table Type field.
- Copy, replicate, delete, and move columns within the table definition and copy column definitions into the table from other tables.
- View the affects that different data type decisions will have on the total row size. A dynamic row size indicator appears showing the current row size in the Row Size field.
- Column Explode and Column Type screens are provided for entering column data.
- Create primary and foreign key rules as part of the table create (or table alter) process. Table and column selection panels make selecting the right columns easy.

When you finish creating your table, execute, edit, or cancel the DDL.

Create a Table

The Table Create facility offers help in creating a table accurately and efficiently. However, these screens can camouflage the easy process of creating a table. This section summarizes the required and optional steps for creating a table.

Follow these steps:

1. Go to the Table Create screen.
2. Enter data in the required table information fields (Table and Creator), and optionally update other fields.
3. Enter the required column fields (Column Name and Column Type), and optionally update other fields.
4. Press F3 (End) to process the creation.

More information:

Table Definition (see page 345)
Access Table Functions (see page 343)

Table Definition

The table definition fields appear on the top part of the Table Create screen (Table Create). These fields determine the storage characteristics of the table as well as any table comments or labels. You can temporarily hide these fields using the H (Header) command. If the H (Header) command is used, only the Column Definition section appears.

Note: For information about the fields, press F1 (Help).

You can also specify whether the logging of SQL INSERT, UPDATE, and DELETE operations on the table is augmented by additional information.

Column Definition

The Column Definition section of the Table Create screen defines the attributes of each column in the table.

The Table Column List is the scrollable region on the Table Alter, Table Create (see Table Create for an example), and Table Template screens. It lets you maintain the columns in your table, insert new columns, and delete existing columns using simple line commands and by directly entering data into the displayed fields within the list. You can define up to 750 columns per table.

Note: If you are creating, altering, or templating a materialized query table (MQT) (see page 362), a column list is not available. However, to view or change column list information, you can change Table Type to REGULAR. When you are finished, you can enter MQT for Table Type to redisplay the MQT panel.

From the Table Column List, you can:

- Specify line commands to define your columns.

  Note: For detailed information about valid commands, see the online help.
- Specify the numeric place of the column in the table.
- Designate the unique name for the column. You can use column selection criteria to retrieve column names from other tables.
- Identify whether the column is a built-in or user-defined data type (UDT). To select from a list, enter selection criteria. Masking can be used.

  Note: For a detailed list of valid types, see the online help.

- Specify the length of the column for DECIMAL, CHAR, VARCHAR, GRAPHIC, VARG, BLOB, CLOB, and DBCLOB data types. For DECIMAL, this value is its precision.

  The Scale of the column is also displayed in this field, following the size.

  Indicate whether the column can contain a null value. To access a selection list, enter **?** in the N column.

- Indicate the type of default to be used. Valid values depend on the columns data type and the setting of the null indicator (N column). To access a selection list, enter **?** in the D column.

  Note: For more information, see the online help.

- Specify a subtype for a character string column. This option only applies to columns whose data types are CHAR, VARCHAR, and LONG VARCHAR.

  Note: For more information about valid values, see the IBM DB2 *SQL Reference* guide.

- Indicate the position of the column if the column is part of a Primary key.

The following fields can be viewed using the horizontal scrolling keys:

Note: If the horizontal scrolling mode is set to Column (the default), the Comments field will not be displayed until the entire field can be displayed.

Schema

Specifies the high level qualifier of the column type (see the Column Type field). If Column Type is DISTINCT, this displays the schema of the distinct type. Otherwise, the value is SYSIBM.

Note: When creating a table, the initial default value for schema can be set using the DEF (object definition defaults) primary command.

Label

Contains any label information (up to 30 characters) for the column. The default is none.

Comments 1-50

Column comment information, up to 50 characters. To edit or enter more than the 50 characters displayed on the Table Screen, use the E (EXPLODE) line command. The Explode Screen lets you enter and edit the full 254 characters supported by DB2. Any editing changes made in this field affect the first 50 characters of the comment. All other characters after the first 50 characters are not affected. The default is none.

FLDPROC

Specifies the name of an edit routine that transforms or translates a field to or from its stored condition. If blank, the column does not have a field procedure. You can add or alter this field. The default is none.

FLDPARM 1-50

Field procedure parameters. You can enter or modify a parameter list that will be passed to the FLDPROC routine. Allows entry of up to 50 characters as the parms to pass to the field procedure. This field can be up to 254 characters. See the Comments 1-50 field description for more information about entering more than 50 characters of information.

Column Default Value Screen

The Column Default Value Specifications screen lets you specify an initial (default) value for a new column on an altered or newly created table. The value you specify is dependent on the size of the column and the column's data type. User-defined data types are supported.

A default value can also be specified for an existing column when the Column Default Indicator (D field) is set to **C** (Constant).

Note: For all Default Types (with the exception of C), the default value specified is used only when the table is unloaded. When the default type is C, the value specified is used by DB2 and will remain in effect for new rows inserted and all existing rows updated.

To access the Column Default Value Specifications screen, enter **V** in the CMD field for a newly created column on the Table Create, Table Template, or Table Alter screen.

To provide additional information for user-defined data types (UDTs), this screen also displays the source schema, data type (including length and for decimal values scale), null indicator, and default type.

Inserting Columns from Other Tables

You can insert columns from other tables, even if the column or table name is unknown. You can list the columns currently defined in the SYSIBM.SYSCOLUMNS table.

Enter selection criteria to limit the number of columns displayed. If a column is selected for inclusion into the new table, all attributes of the columns are copied into the table definition. The column copy feature makes it very easy to create standard column definitions that are reused by different tables.

Insert a New Column

You can insert a column.

Follow these steps:

1. Insert a new column (or overwrite an existing column) on the Table Create, Alter, or Template screens.
2. Type in the column name.

   Press Enter.

   The Column Type selection panel appears. The possible data values are displayed. This information can be different depending on what version of DB2 you are running. The Creator/Schema type defaults to SYSIBM.

   For a description of the fields and possible column types, see Column Definition.
3. Select a column type and press Enter.

   A pop-up dialog box provides you with possible ranges and default values for the column type you selected.
4. Enter the column size and press Enter.

Insert a User-Defined Column

You can insert a user-defined column.

Follow these steps:

1. Insert a new column (or overwrite an existing column) from the Table Create, Table Alter, or Table Template screens.
2. Specify a new column name and press Enter. The Column Type selection panel appears. All column type values default to those provided within SYSIBM.

3. In the Creator field, replace SYSIBM with an asterisk (*) and press Enter.
4. Press F8 (Down) to scroll past the SYSIBM defaults to view valid user-defined data types.

   For all user-defined data types, the schema, source schema, source type, and owner are displayed.

   Note: For information about the fields, press F1 (Help).
5. Select the user defined data type you want to use for your column type.

Confirming the Table Creation

When all information has been entered, press F3 (End). A Creation confirmation screen will appear. Accept, edit, or reject the generated DDL.

Table Template

The template option lets you create a new table using an existing table as a template. A template session is actually a create session with the additional step of selecting a table as a starting point. You must change the table name, and you can also change other fields. Dependent objects are not included in the template operation. The Table Template screen displays the values used to create the template table.

Change the fields for the new table according to the procedures outlined in Table Create. After changes have been made, press F3 (End) to process the template.

Table Alterations

You can alter a table by using the following:

- DB2 ALTER TABLE statement
- Table alteration functionality within CA RC/Update

Changes that you cannot make by using a DB2 ALTER TABLE statement must be made by dropping and recreating the table with the new definition. If supported changes are combined with unsupported changes, the table must be dropped and re-created.

Note: When the table is dropped, all data, dependents, and security are also dropped.

The RC/Alter facility within CA RC/Update fully supports the alteration of any DB2 table attribute. If DB2 does not support the requested alteration, the RC/Alter facility performs a drop and re-creation. This activity includes saving and restoring all data, dependents, and security of the table.

Note: CA RC/Update does not differentiate between catalog tables and non-catalog tables. Therefore, if a table alteration involves changing or removing the database name and the Data Cap setting, a table drop is required.

When RC/Alter is invoked, a data set is populated with the DDL for dropping and recreating the table. To perform the alteration, you must execute the DDL through the Batch Processor. The actual alteration can be scheduled for a particular time.

To use the table alteration functionality, you access the Table Alter panel from the CA RC/Update main menu. On this panel, you can edit table information as needed.

Note: You can use the COMPARE command to compare the current version of the table with the edited table. For more information about the COMPARE command, see the online help.

Online Schema Support Regarding Table Alterations

The product provides the following support for online schema evolution (extended ALTER functionality introduced in DB2 V8):

- Updates of column types and lengths through ALTER TABLE statements with the ALTER COLUMN *column-name* SET DATA TYPE clauses, with one exception: DROP and CREATE DDL will be generated for the table if a DECIMAL column is altered to a floating point data type and the column is used in an index.
- Changing of the VOLATILE attribute to NOT VOLATILE (and vice versa).

To avoid versioning of objects due to the new ALTER extensions, you can use the Drop & Re-create analysis option to force the dropping and creating of DB2 objects as was done prior to DB2 V8. REORG requests are honored when requested for tablespaces that are placed in AREO status when an extended ALTER occurs.

Setting the Drop & Re-create analysis option to N generates ALTER TABLE statements with ALTER COLUMN *column-name* SET DATA TYPE clauses (instead of dropping and re-creating the table). If you execute this ALTER, the following results can occur:

- The tablespace can be left in Advisory Reorg Pending (AREO) status.
- One or more indexes can be left in AREO or Rebuild Pending (RBDP) status.
- If you are running DB2 V8 in Compatibility Mode (CM) or Enabling New Function Mode (ENFM), and you increase the length of a VARCHAR column that participates in an index, the index will be left in Rebuild Pending (RBDP) status.

Note: You can select the REORG analysis option TS, IX, or B to automatically generate the REORG TABLESPACE or REORG INDEX control cards in the analysis output DDL file and remove the pending status.

Confirming the Alter

After all information has been entered, press F3 (End). Depending on the type of alter performed, one of the following results occurs:

- If the changes can be made with a standard DB2 ALTER TABLE command, a confirmation screen appears, from which you can perform any of the following actions:
  - Accept the generated DDL.
  - Reject the generated DDL.
  - Make modifications to the generated DDL before accepting it.
- If the changes require an CA RC/Migrator analysis or if the current operation mode of CA RC/Update is A, the RC/Alter screen appears, from which you can perform one of the following actions:
  - Enter the required information, press Enter, and then use the Batch Processor (BP) facility to execute the generated DDL.
  - Enter END to return to the Table Alter screen without invoking RC/Alter.
  - Enter CANCEL to exit RC/Alter and discard your changes.

  Note: If a CA RC/Migrator analysis is required to generate the DDL necessary to make your changes, an O or B Operation Mode setting is ignored.

  The analysis phase generates all statements that are necessary to alter the table or to drop and re-create the table, data, dependents, and authorizations.

For complete information, see DDL Execution in the “Using CA RC/Update” chapter.

ALTER Return Codes

The following describes the ALTER return codes:

4

Indicates that a condition has been detected that might need user attention. The DDL will most likely execute, but there might be a problem with the view. You should examine the view text. Warning messages will supply you with further information.

8

Indicates that a condition has been detected that needs user attention. The resulting DDL will not execute, and a DB2 SQL error will result. You should review the DDL to determine what needs to be done to correct the source of the error return code.

Table Drop Considerations

When a table is dropped in DB2, its data, dependents, and authorizations are also dropped. If the table has any referential integrity rules, all dependent tables' referential integrity rules are also removed (but not the tables). Under the rules of DB2, a table cannot be dropped if it is defined in a partitioned tablespace. However, CA RC/Migrator automatically drops the partitioned tablespace necessary for the table DROP and then drop the table. A table cannot be dropped while a DB2 utility has control of the table's tablespace.

Table-Controlled Partitioning

Table-controlled partitioning (TCP) lets you control partition boundaries for a table using values defined in the CREATE TABLE statement. This contrasts with index-controlled partitioning (ICP), where partition boundaries are controlled by CREATE INDEX statement values.

TCP is more flexible than ICP. With TCP, you can create more than one partitioned index in a tablespace, and a partitioning and clustering index are not required.

Note: For a complete discussion of TCP, including the differences between TCP and ICP, see the IBM *DB2 Universal Database for z/OS Administration Guide*.

CA RC/Update fully supports TCP when creating, altering, and dropping objects on DB2 V8 and above. If your DB2 version is less than V8, TCP and the features described in this section are not available on your DB2.

When creating, altering, or templating a table, you can manage its TCP as follows:

- Convert a table using ICP to use TCP.
- Convert a table using TCP to use ICP.
- Manage partitioning key columns.
- Manage partitions and their limit values.
- Manage relationships between partition limit values.

Furthermore, if your table uses ICP, you can effortlessly have DDL automatically generated for you so that DB2 will implicitly convert your table so that it uses TCP instead of ICP by simply doing any one of the following things from the Table Alter screen (assuming no other changes are included that would require a drop and recreate of the table):

- Change the Partitioning option from NO to YES.
- Enter the KEY command on the primary command line, causing the Table Partitioning Key Col Selection & Maint screen to display, and then enter CANCEL (or END without making any changes to the table's partitioning key.)
- Enter the LIMITS command on the primary command line, causing the Table Partitioning & Limit Key Values screen to display, and then enter CANCEL (or END without making changes to the table's partitioning that would cause a reduction in its original count of partitions.)

Convert from ICP to TCP

Although there are several methods that can be used to indicate that you want a table to be implicitly converted by DB2 to use TCP instead of ICP, this is the simplest direct method recommended that uses the Table Alter screens.

Follow these steps:

1. Go to the Table Alter screen, specifying the table that you want to convert.
2. Verify that the Partitioning option is **NO**.

   Note: If the Partitioning option is YES, your table is already using TCP and the remainder of this procedure will not work.
3. Make sure that the TS Parts field displayed to the right of the Partitioning option on the Table Alter screen displays an integer value indicating the number of partitions that are defined for the table's tablespace.

   Note: If the TS Parts field displays Unkn or NONE, your table does not reside in a partitioned tablespace and the remainder of this procedure will not work.

4. Change the Partitioning option on the Table Alter screen from NO to **YES** and press Enter.

   Note: The message RO435W Your table will be converted to use TCP instead of ICP should display. If a different message displays, press HELP to get a detailed explanation as to the possible problems and correct.

   The PS column in the Table Column List will display the Partitioning Sequence of each column that will be a member of your table's partitioning key (derived from the table's clustering index key.)

5. Verify that the Operation Mode in the upper right portion of the screen is set to **O**, and enter END.

   The CA RC/Update Alter Confirmation screen should display, showing the generated ALTER TABLE statement(s) that will implicitly convert your table to use TCP.

6. Press the Enter key to continue and to have the generated DDL executed online by the Batch Processor. (Depending on profile settings for your product, you may be presented with a DDL Execution Confirmation pop-up window. If this happens, respond with **Y** to execute the DDL.)

More information:

Access Table Functions (see page 343)

Convert a TCP Table to Use ICP

Because DB2 provides no mechanism to cause an implicit conversion of an existing table from TCP to ICP, this procedure will generate DDL that will cause the table and its indexes to be dropped and then recreated:

Important: The table must have a clustering partitioned index defined because this clustering partitioned index is re-used for the clustering index when the table is re-created.

Follow these steps:

1. Go to the Table Alter screen, specifying the table that you want to convert.
2. Verify that the Partitioning option is **YES**.

   Note: If the Partitioning option is NO, your table is not using TCP and the remainder of this procedure will not work.

3. Change the Partitioning option on the Table Alter screen from YES to **NO** and press Enter.

   The PS column in the Table Column List will display blanks for each column that was previously shown as being a member of the table's partitioning key.

4. Enter END.

   Regardless of the Operation Mode setting in the upper right portion of the screen, the CA RC/Update Alteration Analysis screen will display.

5. Specify analysis options as needed, and press Enter.

   The DDL is generated and written to the output data set.

6. Review the generated DDL, and execute the DDL using the Batch Processor.

   Note: If the generated DDL contains utility statements, such as UNLOAD and LOAD statements, it cannot be executed online and must be executed through a batch job.

More information:

Access Table Functions (see page 343)
Considerations when Converting a Table Object from TCP to ICP (see page 355)

Considerations when Converting a Table Object from TCP to ICP

Changing the Partitioning option from YES to NO causes an implicit delete of all of your table's partitioning key columns and limit values; and, depending on the index objects originally defined for your table, it will (or may not) be converted to use ICP according to these following rules and processes:

- An automatic complete conversion of your table from TCP to ICP, complete with a type-2 clustering index defined for the table, does NOT occur if your table does not have a type-P clustering index defined; in which case, the generate DDL will drop the table's tablespace and then recreate the table and its partitioned tablespace, only; no partitioned clustering index will be created and your table will have neither TCP nor ICP after executing the generated DDL.
- If, however, your table was originally defined as using TCP and *also* has a type-P clustering index defined, the generated DDL will drop the table's tablespace and then recreate the partitioned tablespace, the table, and also redefine the type-P clustering index as a type-2 clustering index using the table's original partitioning information for the redefined index's key and limit values.

Important: If the type-P clustering index's key is a superset of the table's partitioning key, the extra key columns are included in the re-definition of the index from a type-P to a type-2, but with no limit values for the extra columns in the generated DDL.

Notwithstanding the above considerations, if there are other additional type-P and/or type-D indexes on the table, DDL will be generated for recreating all these other index objects; however, the DDL for these other index objects will not be able to be successfully executed and the objects recreated because the partitioning information, keys and limit values for the table will have been lost because the table, after being re-created using the generated DDL, will no longer be using TCP.

Additionally, when the table's partitioning key information is implicitly deleted by this method, subsequently changing the Partitioning option back to **YES** within the same Table Alter session will not cause the deleted information to be automatically reinstated. Instead, the Table Partitioning Key Col Selection & Maint screen automatically displays so that you can specify new key columns.

If you want to reinstate the original partitioning information for the table without canceling and starting over again, use the **RESET** command on the Table Alter screen. This will reset the table object back to its original definition.

Manage Key Columns for a TCP Table

You manage your table's partitioning key columns using the Table Partitioning Key Col Selection & Maint screen. This screen helps facilitate the selecting and arranging of your table's partitioning key columns. From this screen, you can:

- Select columns to include in your table's partitioning key.
- Move and re-arrange columns in the key.
- Delete columns from the key.
- Specify each key column's order as ascending or descending.

To access the Table Partitioning Key Col Selection & Maint screen, enter the **KEY** command on the Table Create, Table Template, or Table Alter screen.

Note: If the table is using ICP, the table will be converted to use TCP, deriving its partitioning key columns and limit values from the table's clustering index.

If the table's tablespace is not defined, or it is not defined as partitioned, you must first specify **YES** in the Partitioning field on the Table Create, Alter, or Template screen before entering the KEY command.

This screen contains two dynamic areas that are scrollable UP and DOWN. The top list (Column Name Selection) contains all of the columns belonging to the table object that have not yet been selected for the key and that are also eligible for selection. The bottom list (Table Partitioning Key) displays the columns that compose your key.

Follow these steps:

1. Enter **S** in the Column Name Selection list next to each column to be included in your table's partitioning key, and press Enter.

   Note: You can select a block of lines at one time, using the S*nn* command (where *nn* specifies the number of columns) or a pair of SS commands. For more information, see the online Help.

   If no key columns were previously defined and the list at the bottom of the screen was empty, the columns you selected will have been automatically inserted; otherwise, a message will display indicating that your selection(s) are pending.

2. Specify the appropriate insertion point in the Table Partitioning Key area as follows, and press Enter:

   - Specify **A** on a line after which you want the columns to be inserted.
   - Specify **B** on a line before which you want the columns to be inserted.

   Columns are inserted according to your specification.

   Note: The default ORDER for selected columns is always ASC (ascending.) If you want the column to be defined as descending, enter **DESC** in its ORDER field.

3. (Optional) After you have selected all of the columns for your key, modify the arrangement of your key columns or delete unwanted key columns from the second list using the M (Move) and D (Delete) line commands. The line commands that are available for use in the second list are: A, B, Dnn, Mnn, Snn, DD, MM, and SS. For more information, see online help.

4. Press the END key to process your work.

   The changes to the key columns are saved, and the screen from which you came appears again.

You can enter the L command on the primary command line to choose the display properties of the Column Name Selection List. By default, the Column Name Selection list displays only the columns that are eligible to be selected as key columns for your index. Entering L toggles the Column Selection List display mode so that ineligible columns (although not selectable) as well as columns that are already part of the key are displayed.

Note: For more information, see the online help.

You can use the HEADER command to toggle the header portions of both lists ON and OFF to provide additional space on the screen for both lists.

Setting the Hide option, located in the upper right corner of the Column Name Selection List, to Y will hide the Column Name Selection List to provide the most space on the screen for the Table Partitioning Key List, making it easier to manage larger and more complex compound keys.

Modify the Partitions and Limit Key Values of a TCP Table

You manage your table's partitions and limit values using the Table Partitioning & Limit Key Values screen. This screen allows you to individually manage all of your table's partitions and limit values using one convenient scrollable list. From this screen, you can:

- Copy, repeat and exchange limit values amongst partitions.
- Add, copy, move, repeat, delete, and rotate partitions.
- Compare any partition and limit value changes with the original partition organization and limit value settings.

To access the Table Partitioning & Limit Key Values screen, enter the **LIMITS** command on the Table Create, Table Template, or Table Alter screen.

Note: If the table is using ICP, the table will be converted to use TCP, deriving its partitioning key columns and limit values from the table's clustering index. If the table is not using ICP, it must have a defined partitioning key before the LIMITS command is allowed.

The limit values for each column in the partitioning key are arranged horizontally across the screen. The partitions are arranged vertically, with one row for each partition in your table. You can scroll up and down through the partitions. If all columns in your table's partitioning key do not fit on the screen at one time, you may scroll the display left and right to see the other key columns.

Follow these steps:

1. Modify the partitions and limit values as needed, and press Enter.

   You can directly enter limit values into any column for any partition on the screen. You can also use commands to facilitate the setting, moving, repeating, and clearing of values and partitions, depending on the maintenance mode to which the screen is set.

   For a complete list of line commands and primary commands that you can use to modify partitions and limit values, see the online help.

   Note: If you want to discard all changes that you made up to the point of the last explicit or implicit save, enter **RESTORE**. (For more information about the SAVE and RESTORE commands, see the online help.)

   If you want to undo changes that you made to a partition or limit key, enter the **U** (undo) command beside the partition. All unprotected limit values are reset to their original value. When COMPARE is ON, you can also recover a partition that was deleted.

   Note: For more information about how to protect and unprotect columns using the PRO (protect) and UNP (unprotect) commands, see the online help.

2. When you are done making changes, enter END.

   Your changes are implicitly saved and you are returned to Table Create, Alter, or Template screen.

   Note: If you make changes that you do not want to save, enter **CANCEL** instead. All of your changes, up to the point of the last implicit or explicit save, will be discarded, and you will be returned to the previous screen.

More information:

Manage Key Columns for a TCP Table (see page 356)

Rotate Partitions

If you want to logically rotate the first partition so that it is the last partition, you can use the ROTATE command.

Note: The ROTATE command is not permitted when creating or templating a table or when the table's tablespace was not originally partitioned.

Follow these steps:

1. Enter **ROTATE** in the command line.

   The first partition is logically rotated so that it is the last partition and appears as the last partition in the list with its limit values set to blanks.

2. Enter, at minimum, a limit value for the rotated partition's first key column.

   Note: If partition values are ascending, the new partitioning key for the rotated partition must be higher than the current high key limit. If partition values are descending, the new partitioning key must be lower than the current low key limit.

Change Maintenance Modes when Modifying Partitions and Limit Key Values

While modifying partitions and limit key values, you might need to change maintenance modes, depending on the modifications you want to make. For example, you might want to permit limit value changes but prevent modifications to partitions.

You can use any of the following modes when modifying partitions and limit key values:

MAXVALUES

Allows you to directly manage MAXVALUE and MINVALUE settings, only, for any key column in any partition without having to use commands.

Note: With the exception of the ROTATE command, you *cannot* manipulate partitions in this mode.

PARTITIONS

Allows you to manage partitions (such as adding, copying, moving, repeating, deleting, and rotating partitions) in addition to managing their limit values.

VALUES ONLY

Allows you to manipulate limit values for partitions (such as setting, clearing, repeating, copying, re-arranging, or exchanging limit values between partitions) independent of the logical arrangement or ordering of the partitions.

Note: With the exception of the ROTATE command, you *cannot* manipulate partitions in this mode.

To change the maintenance mode, enter **P** (for PARTITIONS mode), **V** (for VALUES ONLY mode), or **M** (for MAXVALUES mode) in the command line.

The initial maintenance mode that is used when the Table Partitioning & Limit Key Values screen displays is PARTITIONS.

Note: To change the default mode to VALUES ONLY, enter **SAVESETTINGS** in the command line when in VALUES ONLY mode. The next time the screen appears, it will initially display in VALUES ONLY mode. Only PARTITIONS and VALUES ONLY may be saved as default modes. It is not allowed for MAXVALUES mode.

Auxiliary Tables

An auxiliary table is a table used to store LOB data for a LOB column belonging to a regular (base) table.

The Auxiliary Table screens let you alter, create and template DB2 auxiliary tables. These screens appear when you specify **A** (auxiliary) in the Table Type field on the Table Create, Alter, or Template screens.

The Aux Table screens let you alter, create, and template DB2 auxiliary tables. As with regular tables, you can:

- Enter selection criteria to bring up lists of objects to choose from.
- Apply user defaults to the table definition using the Apply Def command.
- Display the original and current definitions of an object using the COMPARE command.

In addition, you can display a list of LOB columns in the base table, by specifying selection criteria in the Base Table field. From this list, you can select the LOB column for your table.

When altering a table object of any type, you cannot switch from an auxiliary table to a standard table and vice-versa. However, you can change the type on a create or template.

Note: For auxiliary tables, you must specify a LOB tablespace that resides in the same database as that of the base table in the Tablespace field.

The Base Table, Creator, Column Name, and Part fields are unique to auxiliary tables and contain the LOB column information for the auxiliary table as follows:

Base Table

Indicates the name of the table that contains the LOB column for the auxiliary table. To select from a list, enter selection criteria.

Creator

Identifies the creator of the base table that contains the LOB column for the auxiliary table.

Column Name

Specifies the name of the LOB column in the base table for which the auxiliary table is to store LOB data.

Part

Specifies the partition of the base table for which the auxiliary table is to store the specified column's LOB data.

TS Parts

Indicates whether the tablespace is partitioned. A value is required when the tablespace is partitioned. Do not specify a partition when the tablespace is not partitioned. This field is optional when the specified base table or its tablespace is not found.

Do not specify a partition number for an auxiliary table that already exists for the same partition, base table and LOB column.

Note: For information about the fields, press F1 (Help).

Materialized Query Tables (MQTs)

A *materialized query table* (MQT) contains data that is derived from one or more source tables specified by a fullselect. A source table is a base table, view, table expression, or user-defined table function.

Note: A unique index *cannot* be created on an MQT.

MQTs can contain frequently requested information. MQTs are often used in data warehousing applications to improve SQL efficiency. However, keep the following points in mind:

- To ensure that data is current, MQTs must be periodically refreshed (which can be expensive).
- MQTs use storage.

Create an MQT

You can create MQTs.

Follow these steps:

1. Perform the following actions on the CA RC/Update Main Menu:
   - Type **C** in the Option field.
   - Type **T** in the Object field.
   - Type a name in the Item Name field.
   - (Optional) Type values for one or more of the remaining fields to add to the object definition.

   Note: For complete information about the fields on the CA RC/Update Main Menu, see the online help.

   Press Enter.

   The Table Create panel appears.
2. Enter **MQT** for Table Type.

   The panel is renamed *MQT Create*, and the column information is replaced by an empty, boilerplate fullselect statement:

   ```
   ( @ ) AS ( SELECT @ FROM )
   ```

   @ (first occurrence)

   Holds the place of the names assigned to the columns in the table.

   @ (second occurrence)

   Holds the names of the columns belonging to the objects that are the subject of the query.

   Note: You can include @ symbols anywhere in the SELECT statement to hold the place of column names. If there is no @ symbol, and you select columns, the columns are placed before FROM in the first SELECT statement.

   FROM

   Precedes the names of existing or predefined tables or views (if you choose to specify them).

3. Make any of the following entries as needed:

   a. Specify values in place of the first and second occurrences of @.

   b. Specify the name of a table or view after FROM, ensuring that it contains any columns specified in the second @ occurrence.

   Press Enter.
   One or more lists of columns appear in a separate scrollable area at the bottom of the panel. These lists contain names of selectable columns belonging to the tables or views being queried by the fullselect. The number of lists depends on how many valid tables or views are specified after the FROM clause in the fullselect.

4. (Optional) Type **S** next to columns that you want to select, and then press Enter.

   Note: All columns in the column selection list have assigned numbers. In your SELECT statement, you can use an abbreviation by specifying the column number preceded by a colon. The column abbreviations are automatically expanded to their full column names.

   All selected columns are inserted into your fullselect text at *each* @ placeholder.

5. Make other changes as needed to the values for your MQT, and then press the END key.

   The CA RC/Update Creation Confirmation panel appears with the generated DDL for creating your object.

6. Perform *one* of the following actions:

   - Press Enter to execute the DDL online through the Batch Processor.
   - Enter **E** in the command line to edit the generated DDL.
   - Press the END key to return to the create panel.

   Note: Depending on the CA RC/Update Profile settings for your user ID, you might also have the option of rebuilding the DDL with RC/Alter so you can specify utilities and extended DDL generation and analysis options.

   Your session continues, depending on your choice.

Alter an MQT

You can alter MQTs.

Follow these steps:

1. Perform the following actions on the CA RC/Update Main Menu:
   - Type **A** in the Option field.
   - Type **T** in the Object field.
   - (Optional) Type values for one or more of the remaining fields to add to the object definition.

   Note: For complete information about the fields on the CA RC/Update Main Menu, see the online help.

   Press Enter.

   The MQT Alter panel appears.
2. Make any edits to the fullselect statement by doing one or both of the following:
   - Enter changes directly into the statement.
   - Invoke a list of column information, and then select columns to add to the fullselect statement.

   Note: If the result set for the fullselect statement changes, some restrictions (see page 374) might apply.

   All columns in the column selection list have assigned numbers. In your SELECT statement, you can use an abbreviation by specifying the column number preceded by a colon. The column abbreviations are automatically expanded to their full column names.

   Changes are applied as dictated. All selected columns are automatically inserted into your table's fullselect text at *each* @ placeholder.
3. Make other changes as needed to the values for your MQT, and then press the END key.

   The Alter Confirmation panel or Alteration Analysis panel (for RC/Alter) appears, depending on whether changes can be made without dropping and re-creating the object.

4. Continue the process as follows, depending on the panel presented:
   - Respond on the Alter Confirmation panel by performing *one* of the following actions:
     - Press Enter to execute the DDL online through the Batch Processor.
     - Enter **E** in the command line to edit the generated DDL.
     - Press the END key to return to the alter panel.
   - Respond on the Alteration Analysis panel by typing **O** or **B** for Execution Mode, completing other fields as needed, then pressing Enter.

     Note: Based on your selections on the Alteration Analysis panel, you can invoke the Analysis Options panel, Recovery Options panel, or Batch JCL Specification panel (if batch mode is specified).

   Your session continues, depending on your choice.

More information:

How to Manage and Manipulate SQL Text (see page 373)
Refresh an MQT (see page 370)
Fullselect Restrictions (see page 374)

Template an MQT

You can create an MQT using an existing table as a template. This process is known as *templating* an MQT.

Note: Dependent objects are *not* included in the template operation.

Follow these steps:

1. Perform the following actions on the CA RC/Update Main Menu:
   - Type **T** in the Option field of the main header.
   - Type **T** in the Object field of the main header.
   - Type a name in the Item Name field to specify the MQT you want to template.

     Note: If you do not specify a name, a selection panel appears when you press Enter.
   - (Optional) Type values for one or more of the remaining header fields.

   Note: For complete information about all fields, see the online help.

   Press Enter.

   The MQT Template panel appears.

2. Complete fields as follows:

   Table

   Specifies a name for your MQT.

   Creator

   Specifies the name of a creator.

   Press Enter.

   The values are applied.

3. Specify any additional changes for the MQT, and then press the END key.

   A confirmation panel appears for confirming that you want to execute DDL to create your MQT.

4. Perform one of the following actions:

   - Press Enter to execute the DDL online through the Batch Processor.
   - Enter **E** in the command line to edit the generated DDL.
   - Press the END key to return to the MQT Template panel.

   Note: Depending on the CA RC/Update Profile settings for your user ID, you might also have the option of rebuilding the DDL with RC/Alter so you can specify utilities and extended DDL generation and analysis options.

   Your session continues, depending on your choice.

More Information:

Reconfigure SQL Text into a More Readable Format (see page 373)
Enable Unrestricted Text Entry in the SQL Text Area (see page 373)
Separate SQL Text into Individual Lines of SQL Tokens (see page 374)
Refresh an MQT (see page 370)

Convert a Base Table to an MQT

You can use CA RC/Update to alter a table by converting the table to be an MQT. All changes that need to be propagated to dependent objects are incorporated into the alterations.

Important! During the conversion, all objects and attributes that violate MQT restrictions will be dropped. If the table's column definition is different from the result set of the fullselect statement that you will use to create the MQT, you must alter the column definitions to match the fullselect result. If you do not, certain restriction might cause problems during the alteration.

Follow these steps:

1. Perform the following actions on the CA RC/Update Main Menu:
   - Type **A** in the Option field of the main header.
   - Type **T** in the Object field of the main header.
   - Specify a table name in the Item Name field.
   - Specify a creator name in the Creator field.

   Note: If you do not specify a table name or creator name, a selection list appears, from which you can choose.

   Press Enter.

   The Table Alter panel appears.
2. Enter **MQT** in the Table Type field.

   The panel is renamed *MQT Alter*, and a fullselect statement appears at the bottom of the panel.
3. Change any other table definition fields as needed, and then press the END key.

   The CA RC/Update Alteration Analysis panel appears, where you must analyze your requested changes.
4. Complete the fields of the Alteration Analysis panel to set up an online analysis or batch analysis, and then press Enter.

   CA RC/Update performs an alteration analysis, creating the necessary commands and DDL for the alteration. When the analysis is complete, you can submit the job to perform the alteration and display an analysis report, showing details of the processing. Your session continues, depending on your choice.

More Information:

Isolation Level Considerations for MQTs (see page 371)

Convert an MQT to a Base Table

One reason you might want to change an MQT into a base table is to perform table operations that are restricted for an MQT (for example, adding partitions, altering table columns, renaming the table, or creating a unique index).

Follow these steps:

1. Perform the following actions on the CA RC/Update Main Menu:
   - Type **A** in the Option field.
   - Type **T** in the Object field.
   - (Optional) Type values for one or more of the remaining fields to add to the object definition.

   Note: For complete information about the fields on the CA RC/Update Main Menu, see the online help.

   Press Enter.

   The MQT Alter panel appears.
2. Complete fields as needed, type **REGULAR** for Table Type, and then press Enter.

   The values are applied as your table's attributes.
3. Press the END key.

   The RC/Update Alter Confirmation panel appears with the generated DDL for creating your object.
4. Perform *one* of the following actions:
   - Press Enter to execute the DDL online through the Batch Processor.
   - Enter **E** in the command line to edit the generated DDL.
   - Press the END key to return to the alter panel.

   Note: Depending on the CA RC/Update Profile settings for your user ID, you might also have the option of rebuilding the DDL with RC/Alter so you can specify utilities and extended DDL generation and analysis options.

   Your session continues, depending on your choice.

Refresh an MQT

You can use one of the following methods to refresh the data within one or more MQTs:

- Issue the REFRESH line command in a table report in CA RC/Query.

  Note: For more information about using the REFRESH command, see the online help within CA RC/Query.

- Execute a REFRESH SQL statement through interactive SQL (ISQL).

Note: You can also set the REFRESH MQT control option to Y on the CA RC/Update Analysis Options panel. This option determines whether to refresh MQTs that are dropped and re-created when RC/Alter is used to make a change.

This section discusses the ISQL method.

Follow these steps:

1. Enter **ISQL** in the command line on any CA RC/Update panel.

   The SQL Editor panel appears.

2. Specify the following statement:

   ```
   REFRESH TABLE creator.mqt_name;
   ```

   creator.mqt_name

   Specifies the name of the MQT you want to refresh.

   Note: You can enter REFRESH statements on multiple lines on the SQL Editor panel to refresh multiple MQTs.

   Press the END key.

   The ISQL Online SQL Execution panel appears, showing the SQL that will be executed.

3. Make the following entries:

   a. Type **S** in the Option field.

   b. Type **C** in the Commit or Rollback field (to commit the SQL that is executed).

   c. (Optional) Complete other fields on the panel as needed.

   Press Enter.

   The SQL execution is processed according to your specifications.

Note: For more information about ISQL, see the *CA Database Management Solutionsfor DB2 for z/OS Value Pack Reference Guide*.

More Information:

Control Options (see page 273)
Create an MQT (see page 363)
Alter an MQT (see page 365)
Template an MQT (see page 366)

Isolation Level Considerations for MQTs

MQT access can be affected by the isolation level that was in effect when the MQT was created. The isolation level controls the type and duration of locks on DB2 objects. When you create an MQT or convert a base table to an MQT, the ISOLATION column in SYSIBM.SYSVIEWS is set to the isolation level being used when the DDL is executed to create the MQT.

An MQT is only usable in query rewrite when the isolation level from SYSIBM.SYSVIEWS is equal to or higher than the isolation level of the dynamic query being considered for rewrite. Thus, to ensure successful queries against an MQT, you can use the Batch Processor .ISOLEVEL command to set the isolation level to a higher value.

Note: For more information about the .ISOLEVEL command and how the Batch Processor uses the isolation level, see the *CA Database Management Solutions for DB2 for z/OS Batch Reference Guide*. For general information about isolation levels, see the IBM DB2 *SQL Reference* guide.

During analysis, CA RC/Migrator may determine that an MQT needs to be created or re-created. To ensure that the MQT is usable in query rewrite, CA RC/Migrator inserts an .ISOLEVEL command in the input stream to set the appropriate isolation level. The isolation level will be one of the following values (in descending order):

- For DDL import, the value specified from an input stream .ISOLEVEL statement
- The isolation level specified on the MQT Create or MQT Alter panel
- The isolation level from the DB2 catalog for the dropped MQT
- The default CA RC/Migrator value for newly created MQTs

More Information:

Convert a Base Table to an MQT (see page 368)
Create an MQT (see page 363)

Control the Display of Columns from Queried Tables or Views

By default, a list appears at the bottom of the MQT Create, Alter, or Template panel, showing selectable column names belonging to the tables or views being queried by the fullselect. (The name of the queried table or view appears after the FROM clause in the fullselect.)

You can control the display of the list of selectable column names.

Follow these steps:

1. Enter **COLS** in the command line on the MQT Create, Alter, or Template panel.

   The list is no longer shown.

2. (Optional) Enter **COLS** again in the command line.

   The list appears.

Insert All Columns into Your Fullselect Automatically

When you use the ALL command on the MQT Create, Alter, or Template panel, all columns in the column selection list are inserted into the object's SQL editable area.

Note: The column selection list does *not* need to be displayed for you to insert all columns. However, PARSE must be ON (meaning tokens that are split between lines are identified and the split is fixed).

To insert all columns into your fullselect automatically, enter **ALL** in the command line on the MQT Create, Alter, or Template panel.

All table/view columns are inserted into the SQL as follows:

- If @ placeholders exist in the SQL, columns are inserted after the placeholders.
- If no @ placeholders exist, the columns are inserted following the last specified column.
- If no @ placeholders exist, and no columns are specified in the SQL, columns are inserted into their syntactically correct defaulted positions, according to DB2 rules if possible.

How to Manage and Manipulate SQL Text

On the MQT Create, Alter, and Template panels, you can manage and manipulate SQL text by doing any of the following:

- Reconfigure SQL text into a more readable format (see page 373).
- Enable unrestricted text entry SQL in the text area (see page 373).
- Separate SQL text into individual lines of SQL tokens (see page 374).

Reconfigure SQL Text into a More Readable Format

You can use the FO command to format the current editable or displayed SQL text into a more readable format with indentation.

Note: Entering this command causes the requested formatting to be applied only once.

To reconfigure SQL text into a more readable format, enter **FO** in the command line on an MQT Create, Alter, or Template panel.

Text is reconfigured.

More Information:

Create an MQT (see page 363)
Alter an MQT (see page 365)
Template an MQT (see page 366)

Enable Unrestricted Text Entry in the SQL Text Area

The TE command enables you to type text freely into the SQL text area. When you reach the end of a line in which you are typing, the cursor automatically skips to the beginning of the next line. This permits you to quickly enter text without regard to cursor position.

Note: The TE feature is OFF by default.

Follow these steps:

1. Enter **TE** in the command line on the MQT Create, Alter, or Template panel.

   The line command area is removed, and free text entry is now allowed.

2. (Optional) Type text as needed, and then press Enter.

   The Text Entry feature is turned off, and you are placed back into the standard text entry mode. If tokens are split across lines, and PARSE is ON, the split will be fixed when you press Enter.

More Information:

Create an MQT (see page 363)
Alter an MQT (see page 365)
Template an MQT (see page 366)

Separate SQL Text into Individual Lines of SQL Tokens

The WS command causes the current editable or displayed SQL text to be broken up into individual SQL tokens, which are placed on separate lines. You can then use standard ISPF line commands to more easily delete, move, and copy columns.

Note: Entering the WS command causes the requested formatting to be applied only to the current editable or displayed text. The current PARSE setting has no influence on the behavior of this command.

Follow these steps:

1. Enter **WS** in the command line on the MQT Create, Alter, or Template panel.

   Any existing SQL text is broken up and displayed on separate lines.

2. (Optional) Make changes, and then enter **WW** in the command line.

   The tokens are reconfigured into uniform text lines.

More Information:

Create an MQT (see page 363)
Alter an MQT (see page 365)
Template an MQT (see page 366)

Fullselect Restrictions

If you use a fullselect while altering an MQT, converting a table to an MQT, or converting an MQT to a table, the fullselect result set must match the table/MQT column definition. Otherwise, the following restrictions apply during alteration analysis:

- Restrictions regarding drop/recreation DDL generation
- Restriction regarding ALTER TABLE DDL generation

More information:

Drop/Recreation DDL Generation Restrictions for Tables or MQTs (see page 375)
ALTER TABLE DDL Generation Restrictions for Tables or MQTs (see page 376)

Drop/Recreation DDL Generation Restrictions for Tables or MQTs

When DDL is generated to drop and recreate a table or MQT, using a CREATE TABLE with a fullselect, the fullselect result set must match the table/MQT column definition; otherwise, the following issues might occur:

- The limit key values for a TCP MQT might not be formatted properly in the generated DDL.

  You can manually edit the limit key values in the generated DDL to properly format them.

- The Unload and Load utility control statements for data conversions might be generated incorrectly when the MQT is created as user-maintained, causing the data conversion to fail.

  To avoid this, use the MQT REFRESH analysis option instead of the Data Unload analysis options. This will cause a REFRESH TABLE DDL statement to be generated instead of the Unload and Load utility statements.

- Column name changes in the fullselect result set will not be propagated to dependent objects (for example, views and indexes).

  To avoid this, complete the steps of the following alternate method regarding the MQT's Table Type.

To prevent these issues, do one of the following to ensure that the table's column definitions match the MQT fullselect result set:

- If you are altering a table, alter the column definitions before changing the Table Type to **MQT**.
- If you are altering a MQT, perform the following steps before changing the fullselect statement:
  1. Change the Table Type to **Regular**.
  2. Alter the column definitions.
  3. Change the Table Type back to **MQT**.
  4. Modify the fullselect to match the altered column definitions.

More information:

ALTER TABLE DDL Generation Restrictions for Tables or MQTs (see page 376)

ALTER TABLE DDL Generation Restrictions for Tables or MQTs

If an ALTER TABLE DDL statement is generated (to convert a table to an MQT), and the table column definitions do not match the MQT's fullselect result set, execution of the statement will fail. To avoid this, force a drop/create of the table by performing one of the following:

- Supply the optional column name list followed by the AS keyword in the fullselect statement.
- Alter the table's column definitions to match the MQT fullselect result set before changing the Table Type to MQT.

CA RC/Update alteration analysis drops the table and creates the MQT.

More information:

Drop/Recreation DDL Generation Restrictions for Tables or MQTs (see page 375)

Temporal Table Support

You can create, template, and alter temporal tables.

If you are implementing temporal table support on a DB2 subsystem running DB2 10 or above, review the following information:

- Support is provided for temporal tables that exist in explicitly created tablespaces. For temporal tables that exist in implicitly created tablespaces, SQL errors might occur.

  The following table types are provided:

  - Temporal
  - History (for system-period or bi-temporal tables)

    Note: History tables are not displayed in the list of objects to template, alter, or drop.

    To create the history table, use *either* of the following options:

    - Template the base table, delete the PERIOD clause and AS column clauses: AS ROW BEGIN, AS ROW END, and AS TRANSACTION STARTID, and then ALTER TABLE ADD VERSION.
    - Flip the table type to REGULAR, and then ALTER TABLE ADD VERSION.

  To add or drop versioning, use ISQL.

- When altering a temporal table, you can:
  - Add or change a default attribute of a column defined AS ROW BEGIN, AS ROW END, or AS TRANSACTION START ID.
  - Add a PERIOD definition.
- When creating, altering, or templating a table, you can add the BUSINESS_TIME WITHOUT OVERLAPS clause to the definition of a unique constraint using the B option on the Unique Constraints Management panel.

  Note: The table must be a temporal table with a BUSINESS_TIME period defined. The constraint cannot explicitly specify a column of the BUSINESS_TIME period.
- When creating, altering, or templating an index, you can specifically include or exclude the BUSINESS_TIME WITHOUT OVERLAPS clause in the index key. To enable this functionality, use the BTWOO field on the Index Create, Alter, and Template panels.

Example: How to Create System Time and Business Time Temporal Tables

This example provides high-level information for creating, templating, and altering temporal tables. For a system time temporal table, the BEGIN, END, and transaction ID columns must be TIMESTAMP(12). The default indicators are B, E, and I, respectively. For a business time temporal table, you can pick all timestamp formats or date, size of 6.

Follow these steps:

1. To create temporal tables:
   - Specify the Table Type as T (for temporal)
   - Define a business time period table with begin and end columns, a column type (date or all timestamp formats), and a size of 6.
   - Define system period time tables with begin and end columns, and a column indicator. Use a column type of timestamp and a size of 12.

   Save and generate DDL.
2. Create a history table by templating the previously created system period time table, and making the following changes:
   - The Table Type to Regular. The B, E, and I attributes go away.
   - The table name.
   - (Optional) Blank out the tablespace name.

   Save and generate DDL.
3. To add versioning, use ISQL (ALTER TABLE ADD VERSION).
4. To alter these tables, change the business table column type from date to timestamp, and analyze the changes. The drop and recreate are generated created for the table. You can also add new columns.

Identity Columns

An *identity column* is one of the following:

- Any column that is defined with a data type of DECIMAL, INTEGER, SMALLINT, or BIGINT
- A distinct type that is sourced on the DECIMAL, INTEGER, SMALLINT, or BIGINT data types and whose column default indicator on the Table Alter, Create, or Template panel is set to A or D

Note: For more information about identity columns, see the IBM *SQL Reference* or *Administrator Guide*.

Create Identity Columns

You can create identity columns.

Follow these steps:

1. Access the Table Create, Table Alter, or Table Template panel (see page 343).

   The Table Create, Table Alter, or Table Template panel appears.

2. Insert a column (see page 348), specifying one of the following for COLUMN TYPE:

   - BIGINT
   - SMALLINT
   - INTEGER
   - DECIMAL

   Press Enter.

   The product inserts the column.

3. Enter one of the following values in the D field (which represents the default indicator attribute):

   A

   Specifies Generated Always as Identity.

   D

   Specifies Generate by Default as Identity.

   The identity column is now created.

View or Update Identity Column Attributes

You can use the V (Value) command to view or update identity column attributes. The Value command lets you do the following:

- Specify a default value for a newly inserted column of an existing table.
- Cast a default for a column, thus causing the DEFAULT WITH clause to be inserted into your DDL (when the column's default option is C).
- Specify the identity column attributes when the column's data type is BIGINT, DECIMAL, INTEGER, or SMALLINT, and the default option is A or D.

For non-identity columns, a maximum of 254 characters can be specified, depending on the data type. Other requirements are as follows:

- When the default indicator for the column is Y or N (or is blank), the column must be a newly inserted column, in which case the default value is used only when the table is unloaded.
- When the default indicator for the column is C, the column could be old or new, in which case the default value is used internally by DB2 for all new rows that are inserted and old rows that are updated.

Follow these steps:

1. Access the Table Create, Table Alter, or Table Template panel.

   The Table Create, Table Alter, or Table Template panel appears.

2. Enter **V** in the Cmd field next to an identity column.

   The Identity Column Value Specification panel appears.

3. Complete the fields as follows:

   Start With

   Specifies the starting value of the identity column.

   This can be any positive or negative value, including zero, which could be assigned to the column, based on its data type. The default is the value specified in the Minvalue field for an ascending sequence or the value specified in the Maxvalue field for a descending sequence.

   Valid values (based on data type) include the following:

   - For BIGINT, the smallest value would be -9223372036854775808, and the largest value would be 9223372036854775807.
   - For DECIMAL, valid values include any positive or negative value within the scope of the decimal number's defined precision that does not contain non-zero digits to the right of the decimal point.

     For example, if the precision is 6, the smallest value would be -999999, and the largest value would be 999999.
   - For INTEGER, the smallest value would be -2147483648, and the largest value would be 2147483647.
   - For SMALLINT, the smallest value would be -32768, and the largest value would be 32767.

   Note: Start With is not necessarily the value that a sequence would cycle to after reaching the minimum or maximum value of the sequence. Start With can be used to start a sequence outside the range that is used for cycles. The range used for cycles is defined by the Minvalue and Maxvalue fields.

Increment By

Specifies the interval between consecutive values of the identity column.

The value can be any positive or negative value that is not zero. Additional limitations are as follows , and does not exceed the value of a:

- The value cannot exceed the value of a small integer constant for an identity column whose data type is SMALLINT.
- The value cannot exceed the value of a large integer constant for an identity column whose data type is DECIMAL or INTEGER.

The default is 1.

Valid values (based on data type) include the following:

- For DECIMAL, valid values include any positive or negative value that is not zero, does not exceed the value of a large integer constant, and that does not contain non-zero digits to the right of the decimal point.

  For example, if the precision is 15, the smallest value would be -2147483648, and the largest value would be 2147483647.
- For INTEGER, the smallest value would be -2147483648, and the largest value would be 2147483647.
- For SMALLINT, the smallest value would be -32768, and the largest value would be 32767.

Note: If the value is positive, the sequence of values for the identity column ascends. If the value is negative, the sequence of values descends.

Cache

Specifies a value regarding keeping some preallocated values in memory and improve performance when inserting rows into a table.

integer

Specifies the number of values of the identity column sequence that DB2 is to preallocate and keep in memory. During a system failure, all cached identity column values that are yet to be assigned are lost, and thus, will never be used. Therefore, this integer value also represents the maximum number of values for the identity column that could be lost during a system failure. The value can be from 2 up to the largest value that can be represented by a large integer constant. The default is 20.

NO

Specifies that values for the identity column are not preallocated. In a data sharing environment, use NO to guarantee that identity values are generated in the order in which they are requested.

Minvalue

Specifies the minimum value that is generated for the identity column.

This value can be any positive or negative value (including zero) that could be assigned to the column based on its data type (and must also be less than the value specified in the Maxvalue field).

If this field is blank, the default for an ascending sequence is the value specified in the Start With field (or 1, if the Start With value is also blank). For a descending sequence, the default is the minimum value of the data type.

Maxvalue

Specifies the maximum value that is generated for the identity column.

This value can be any positive or negative value (including zero) that could be assigned to the column based on its data type, and must also be greater than the value specified in the Minvalue field.

If this field is blank, the default for an ascending sequence is the maximum value of the data type. For a descending sequence, the default is the value specified in the Start With field, or -1 if the Start With value is also blank.

Cycle

Specifies whether the identity column should continue to generate values after reaching the minimum or maximum value (determined by the Minvalue and Maxvalue fields) of the sequence:

YES

Specifies that values continue to be generated for the column after the minimum or maximum value has been reached. After an ascending sequence reaches the maximum, it generates its minimum value. After a descending sequence reaches its minimum, it generates its maximum value. The minimum and maximum values (specified in the Minvalue and Maxvalue fields) determine the range of the cycle.

NO

Specifies that values will not be generated for the identity column once the minimum or maximum value for the sequence has been reached. This is the default.

Press Enter to save your changes.

Changes are saved.

More information:

Access Table Functions (see page 343)
Determining the Default Value (see page 383)

Determining the Default Value

When the Start With, Minvalue, or Maxvalue fields are blanked out on the Identity Column Value Specifications screen at the time that Enter, End, or PF3 is pressed, a default value is automatically determined and substituted in place of the blanks. The default values that would otherwise be determined by DB2 at the time the DDL is executed are instead determined for you and displayed on the screen in place of the blanks so you can see what these values will be before strategy analysis and the generation of your DDL.

Note: Depending on the product you are using, these values might not be available.

To determine the default value, the following processing occurs:

- The Increment By field is inspected first. If this field is blank, or contains any valid positive number for the data type, the sequence is assumed to be ascending; otherwise, it is descending.
- When the Start With field is blanks:
  - For an ascending sequence, the Minvalue field value is substituted in place of the blanks. However, if Minvalue is also blank, 1 is used.
  - For a descending sequence, the Maxvalue field value is used. However, if Maxvalue is also blank, the maximum value of the data type (and precision, if DECIMAL) is used.
- When the Minvalue field is blanks:
  - For an ascending sequence, the Start With field value is substituted in place of the blanks. However, if Start With is also blank, 1 is used.
  - For a descending sequence, the minimum value of the data type (and precision, if DECIMAL) is used.
- When the Maxvalue field is blanks:
  - For an ascending sequence, the maximum value of the data type (and precision, if DECIMAL) is used.
  - For a descending sequence, the Start With field value is used. However, if Start With is also blank, -1 is used.
- All fields are defaulted on newly inserted or defined identity columns.

Note: For additional information about the determination of default values for identity column attributes, see the IBM DB2 *SQL Reference* guide.

Table Check Constraints List Screen

On the Table Check Constraints List screen, you can name sets of values that define constraints for a table by entering U in the Check Const field of the Table Create, Alter, or Template screen.

Note: If you are creating, altering, or templating an MQT, and you have specified Initially Deferred in the Data field, the Check Const field is unavailable. DB2 does not support check constraints for MQTs. Initially Deferred is the default value (unless overridden by object definition defaults).

The current table name appears in the Check Constraints for Table (display only) field. The constraints appear in the Constraint Name column.

To specify the action you want to take for the current line, specify a value in the S column as follows:

(blank)

Creates DDL to apply this check constraint to the current table. All check constraints in the list are active if the Check Const field is set to Y on the Table Create, Alter, or Template screen. This is the default.

C

Creates a new check constraint (valid on first line only). When you specify the check constraint name and press Enter, an SQL Editor session is opened where the check constraint text is entered.

S

Edits the check constraint. An edit session is opened where the check constraint text is entered.

D

Deletes the check constraint. If the constraint already exists for the table, it is dropped when the alteration DDL is generated. If not, it is marked as deleted in the display, and no DDL is generated. During the current edit session, a check constraint marked as deleted can be restored using the U command.

U

Undeletes a check constraint that was marked with a D. Not valid after exiting the Table Check Constraints List screen.

R

Resets the text for this check constraint to the value on the Table Check Constraints List screen at the beginning of the edit session. Valid only for pre-existing check constraint names.

SQL Editor Screen

The SQL Editor screen is where you enter and save the DDL to be associated with a check constraint name on the Table Check Constraints List screen. The DDL you enter here is inserted as a CHECK clause to create or alter the table.

Enter SQL Editor line commands in the line command area. In the text area enter the DDL to associate with the check constraint name being edited. Press F3 (end) to save the text and return to the Table Check Constraints List screen.

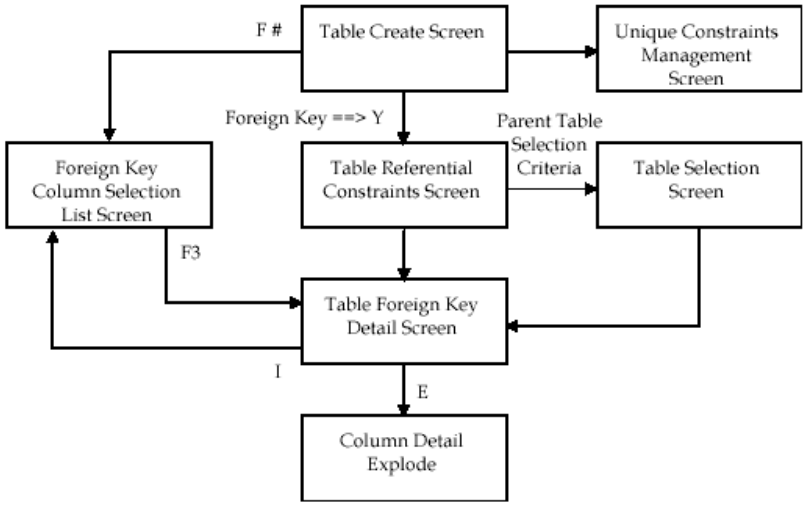
Note: For more information about the SQL Editor, see the *CA Database Management Solutions for DB2 for z/OS General Facilities Reference Guide*.

Referential Integrity for Tables

Referential Integrity is the state in which all values of all foreign keys are valid. DB2 helps ensure referential integrity between your tables when you define referential constraints.

Referential Integrity Screen Flow

The following illustration shows all the table referential integrity screens and how to invoke them:



Referential Integrity Fields on the Table Screens

This section describes the fields on the Table Create, Table Alter, and Table Template screens that pertain to referential integrity.

Forgn Key

Create or update foreign key definitions.

N

Specifies that no foreign key definition exists.

Y

Specifies that a foreign key has been defined. Enter Y to create a foreign key. The Table Referential Constraints screen appears.

U

Specifies to update an existing foreign key definition. The Table Referential Constraints screen appears.

Note: If you are creating, altering, or templating an MQT, and you have specified Initially Deferred in the Data field, the Forgn Key field is unavailable. An MQT cannot be a parent or child in a referential integrity relationship. Initially Deferred is the default value (unless overridden by object definition defaults).

Column CMD

Create a primary or foreign key definition on a column by entering a referential integrity line command.

F*n*

Defines a column as part of the foreign key, where *n* is the number representing the position of the column within the foreign key for a composite key. The Foreign Key Detail screen appears.

P*n*

Defines a column as part of the primary key, where *n* is the number representing the position of the column within the primary key for a composite key. The PK column is updated to reflect the change. The order of the number is important.

Table Screen Main CMD

Enter **UC** or **UniqueConst** to invoke the Unique Constraints Management Screen.

Table Screen Column Headings

The column headings on the Table Screen are as follows:

PK

Displays the column's key sequence number (if the column is defined as part of the primary key). Enter a new number to update the primary key value.

UK

Displays Y if the column is defined as part of a unique constraint in the Unique Constraint Management Screen.

FK

Displays Y if the column is defined as part of a foreign key.

Unique Constraints

A *Unique Constraint* is a rule that prevents duplicate values in one or more columns in a table. Unique Constraints are Primary and Unique Keys.

Note: An MQT *cannot* have unique constraints.

Create a Primary Key

From a Table screen, enter **Pn** in the CMD area, where *n* is the number representing the position of the column within the primary key for a composite key.

Create a Unique Constraint

The Unique Constraints Management screens let you create, alter, or delete primary and unique key constraints from a DB2 table. It also lets you insert, rearrange, or delete a DB2 Table's primary key or unique constraint's columns. You access these screens by entering **UC** or **UniqueConst**.

Follow these steps:

1. Insert a line at the CMD line.
2. Enter a constraint name.
3. Enter a constraint type. Valid values are as follows:
   - U (Unique Key)
   - P (Primary Key)
4. Enter **S** or **E** at the command line to select the constraint for editing.

   Your constraint name will be displayed at the top right of the Unique Constraints Management screen.

5. Enter **I** in the command line to begin inserting the columns that you want included in your constraint.

   Alternatively, you can enter ***** (an asterisk) under the Column field to display the list of available columns for selection.

6. Press PF3 (End) to return to the Create/Alter Table screen.

   Note: If your table has Unique Constraints, a Y appears on the Table Screen below the UK option for each of the columns that apply.

Update a Unique Constraint

You can update a unique constraint.

Note: This procedure applies to both Primary Keys and Unique Keys.

Follow these steps:

1. Enter **UC** or **UniqueConst** in the command line.

   The Unique Constraints Management screen appears. This screen displays existing constraints for the table. It also lets you delete or create new unique constraints.

2. Press PF3 (End).

   The unique constraint definition is finished, and you are returned to the Table screen.

More information:

Create a Unique Constraint (see page 387)

Valid Line Commands

The following list describes the line commands you can use on the Unique Constraint Management screen:

D*nn*

Deletes *nn* constraints, including this one. If *nn* is omitted, 1 is assumed.

E

Places the display into Edit Mode = COLUMN for this constraint so you can edit the constraint's columns. Press PF3 (End) when you are done and the display automatically returns to Edit Mode = CONSTRAINT.

I*nn*

Inserts *nn* blank constraints immediately after this constraint. If *nn* is omitted, 1 is assumed.

R*nn*

Repeats this constraint *nn* times. Creates *nn* iterations of this constraint and inserts them immediately after this constraint. If *nn* is omitted, 1 is assumed.

RES

Resets this constraint back to its old definition. When this command is entered for a newly inserted constraint that has no old definition, the constraint is removed. When COMPARE is ON and this command is entered for a newly deleted constraint, the deleted constraint is reinserted into the list.

S

Selects this constraint for edit. The S line command behaves the same way as the E line command.

SRC

Sets this constraint to its Source definition. This command is valid only for CA RC/Migrator comparisons.

TRG

Sets this constraint to its Target definition. This command is valid only for CA RC/Migrator comparisons.

U*nn*

Undoes *nn* constraints (including this one) and returns them to their old definitions. If *nn* is omitted, 1 is assumed. This command performs the same function as the RES command.

DD

Uses paired DD commands to specify a block of constraints to delete.

RR

Uses paired RR commands to specify a block of constraints to repeat.

UU

Uses paired UU commands to specify a block of constraints to undo.

Referential Constraints

A *referential constraint* is the rule that the non-null values of a foreign key are valid only if they also appear as values of its parent key (Primary or Unique Key). The table that contains the parent key is also called the parent table of the referential constraint, and the table that contains the foreign key is a dependent of that table.

Create a Foreign Key

Foreign keys are attributes of one table that have matching values in primary keys in another table, allowing for relationships between tables. You can use this procedure to create a foreign key.

Follow these steps:

1. Access the Table Create, Table Template, or Table Alter panel.

   The table panel appears.

2. Do *one* of the following:

   - Enter **Fn** in the CMD area, where *n* is the number that represents the position of the column within the foreign key.
   - Enter **Y** in the Foreign Key field (to display the Table Referential Constraints panel), and then type **C** in the option field and enter the relation name, parent table information, ENFORCED rule, and the delete rule.

     Note: If you are creating, altering, or templating an MQT, and you have specified Initially Deferred in the Data field, the Forgn Key field is unavailable. An MQT cannot be a parent or child in a referential integrity relationship. Initially Deferred is the default value (unless overridden by object definition defaults).

   The Foreign Key Detail panel appears.

3. Enter **I** to insert a column.

   The Column Selection panel appears.

4. Select the columns that form the foreign key, and press PF3 (End).

   The foreign key definition is finished, and the table panel reappears.

More information:

Access Table Functions (see page 343)
Display and Update Referential Rules for a Table (see page 391)

Update a Foreign Key

You can update a foreign key.

Follow these steps:

1. Enter **U** in the Foreign Key field on a table screen.

   Note: If you are creating, altering, or templating an MQT, and you have specified Initially Deferred in the Data field, the Forgn Key field is unavailable. An MQT cannot be a parent or child in a referential integrity relationship. Initially Deferred is the default value (unless overridden by object definition defaults).

   The Table Referential Constraints screen appears.

2. Select a rule to be updated.

   The Foreign Key Detail screen appears.

3. Move, insert, or delete columns in the foreign key definition, then press PF3 (End).

   The foreign key definition is completed, and you are returned to the table screen.

Display and Update Referential Rules for a Table

You can display referential rules that have been defined for a table. With existing rules displayed, you can review existing foreign key relationships, update relationships, and generate new foreign keys.

Follow these steps:

1. Access the Table Create, Table Template, or Table Alter panel.

   The table panel appears.

2. Enter **Y** or **U** in the Foreign Key field.

   Note: If you are creating, altering, or templating an MQT, and you have specified Initially Deferred in the Data field, the Forgn Key field is unavailable. An MQT cannot be a parent or child in a referential integrity relationship. Initially Deferred is the default value (unless overridden by object definition defaults).

   The Table Referential Constraints panel appears, listing the foreign keys (by relation rule name) that have been defined on the table.

3. (Optional) Create a foreign key constraint by typing **C** in the first line in the O (Option) column and then completing the following fields:

   Relation Name

   Specifies the referential rule name. If you are creating a new rule, enter a name (maximum of eight characters) in this field. If no name is entered, a rule name is generated based on the first column in the foreign key. The generated name uses the first character of the first foreign key column, followed by seven random characters (A-Z, 0-9).

   Parent Table Creator

   Specifies the ID of the parent table's creator. This field accepts SQL selection criteria.

   Parent Table Name

   Specifies the name of the parent table. This field accepts SQL selection criteria.

   Note: If selection criteria are entered in the Table Name and Creator fields, a Table Selection panel appears, letting you select from a list.

   Delete Rule

   Specifies a delete rule:

   - Restrict—If there is data in the child key, the parent key cannot be deleted. This is the default.
   - Cascade—If the parent is deleted, the child is deleted, regardless of data.
   - Null—If columns in the child are nullable, the values are set to null when the parent is deleted.
   - No Action (Default)—If the parent row is deleted, no action is taken in the child.

   EN (ENFORCED)

   Specifies whether the referential constraint is enforced by DB2 during normal operations (such as insert, update, or delete operations):

   Y

   Uses ENFORCED processing for the foreign key. This is the default.

   N

   Does *not* use ENFORCED processing for the foreign key. A value of N specifies NOT ENFORCED processing.

   Press Enter, and then complete the specifications (see page 394) on the Table Foreign Key Detail panel to return to the Table Referential Constraints panel.

   The Table Referential Constraints panel appears.

4. (Optional) Enter *one* of the following commands in the O (Option) column to manipulate an existing foreign key constraint:

   S

   Selects a foreign key for updating. If you select a foreign key, the Foreign Key Detail panel appears for that foreign key.

   If you are selecting an existing foreign key, the following display-only fields provide additional information about the key:

   Column Name

   For existing relations, displays the names of the columns used in the key.

   Column Type

   For existing relations, displays the data types of the columns used in the key.

   D

   Deletes the foreign key.

   The product processes the foreign key constraint as directed.

More information:

Access Table Functions (see page 343)

Foreign Key Specifications

You can define foreign key specifications using the Table Foreign Key Detail panel. Columns can be added or deleted, and existing columns can be changed. All fields (with the exception of the CMD field) are protected. The panel contains two scrollable areas.

The first scrollable area shows those columns currently contained in the foreign key (identified by foreign key number, column name, column type, size, null, field procedure, ENFORCED setting, *and* parent column. Use the PF7 and PF8 keys to scroll this area. User-defined distinct types are also supported.

The second scrollable area shows *index* information for the parent table identified by index name, coltype, size, null, fldproc, and primary key. Use the PF10 and PF11 keys to scroll this area. Enter **S** next to an index to pair the index columns to the foreign key columns. If there is no column in the foreign key for the pairing, the column is added to both the foreign key and the base table. Any new columns are added to the end of the table.

Note: You can change the parent table by entering a new name and creator information in the Parent Table Name and Creator fields. If selection criteria are specified, a table selection screen appears. If a new table is specified, the new column information appears at the bottom of the panel. Tables created in batch mode are considered invalid.

You press PF3 (End) to save the foreign key definition.

Valid Line Commands

Modification of the foreign key definition is accomplished using various line commands. In the first scrollable area, the following line commands are valid:

 # or n

Specifies the parent table columns number next to the current table. This changes the columns definition in the table, but does not change the column name. The option helps ensure that column definitions in the foreign key match those in the primary key.

I

Inserts a column. This displays the Column selection panel, which lists all columns in the current table.

M, MM, or M*nn*

Moves the column or column block within the foreign key to a location marked by A (AFTER this line) or B (BEFORE this line). The move command only affects the column's position within the foreign key.

D, DD, or D*nn*

Deletes the column or column block from the foreign key.

U or UU

Undeletes a column or column block from the table or foreign key.

If multiple commands are entered, all are processed in turn according to the foreign key number order.

Foreign Key Column Selection List

The Foreign Key Column Selection List displays when you specify **I** next to a column in the first scrollable area of the Foreign Key Detail screen and press Enter.

To select columns to add to the foreign key rule, enter an **S** or a number in the S (selection) column to denote the column's position in relation to the other selected columns or within the foreign key.

Note: LOB and ROWID columns appear, but cannot be selected.

When all columns for the key have been selected, press PF3 (End) to save the selections and return to the Foreign Key Detail screen.

Column Line Commands

The column definition section of the table screen supports standard ISPF line commands for defining and editing your table's columns. Some of the line and block commands supported are insert (I), copy (C), delete (D), repeat (R), and move (M). Special line commands, P# and F#, are provided to define referential rules.

E (explode)

Column Explode Detail lets you perform all attribute editing for a column on one screen.

T (type)

Select a data type for the column from a list of column types.

U (undo)

Removes changes you have made to a column.

Res (reset)

Reset the value of the column to its original value.

The table option offers some special line commands to aid in defining column information. These options are discussed at this time because they apply to three table options: Create, Alter, and Template.

Column Explode Detail (E)

The Column Explode Detail screen presents all the available attributes for a column on a single screen. To display the Column Explode Detail Screen, enter **E** next to each column with which you want to work. The Column Explode option does not need to be used to edit a column's attributes. The benefit of the Column Explode Detail screen is that all attributes are on the screen. If more than one column has been selected, each column is processed individually, in the order of their appearance in the table.

From this screen, two new primary commands are available:

TYPE

Lets you select a data type for the columns from a list of column types.

VALUE

Lets you specify a default value for a new column.

The most current or new data for each column is always displayed, unless specified otherwise. When current data appears, NEW appears in the Data Type field. If any column data is changed, the data associated with the column at the beginning of the session is considered OLD.

Note: Newly inserted columns do not have old data. Therefore, OLD cannot be entered in this field for a newly inserted column.

Enter information in these fields exactly as in the Table Create, Template, or Alter screen (see Table Create). Invoke the Column Type Screen by entering an asterisk (*) or an invalid data type in the Column Type field.

When finished with the Column Explode screen, press the F3 (End) key to return to the Table Screen. Any changes made on the Explode screen are shown on the Table Screen. If changes were made on the Explode screen that should be canceled, press F3 (End) to exit the screen and enter the RES (reset) line command next to the column on the table screen. This will reset the column back to its previous definition.

If more than one column has been selected, the next column automatically appears when F3 (End) is presented. Once all selected columns have been processed, pressing F3 (End) displays the table screen.

Column Type Selection (T)

The Column Type Selection panel makes it possible to select a data type for a column by choosing the data type from a list of valid column types. If changing the column type, the current value appears. To select a Column Type for a column, enter **T** in the line command area. More than one column can be selected simultaneously. Each column will be processed individually, starting with the first selected column. The others will be processed in turn, in the order of their appearance in the table. Enter a question mark (**?**) or any other invalid data type in the Column Type Field to bring up the Column Type Selection panel.

Detailed information about the previous column data types can be found in the Table Create.

If more than one column has been selected, a column type screen for the column next in the table will be displayed. If this is the last column being processed, the Table Create, Template, or Alter screen displays.

To proceed to the next step without changing the data type, press the F3 (End) key without making a data type selection.

Reset or Undo

If changes are made to a column and you want to return the column to its original definition, enter the U (UNDO) or RES (RESET) line command next to the column name. This will retrieve the original definition of the column before any changes were made.

To undelete a column that has been deleted, enter the C (COMPARE) command to invoke the compare screen. Find the deleted column and enter the U or RES command on the Column Deleted line.

Note: For inserted columns, a Column Inserted line appears. If the RES command is entered on an inserted column, it is removed.

Insert and Repeat Considerations

Line commands are very powerful tools for changing the columns in a table. Columns can be inserted, deleted, repeated or moved using the ISPF-like line commands in CA products. Two common situations need more clarification.

Insert a New First Column

Because the **I** (INSERT) line command inserts a new line for a new column following the line on which it is entered, inserting a new column as the first column in a table requires multiple steps.

Follow these steps:

1. Enter the I (INSERT) line command for any of the columns in the table.
2. Specify the new column name and attributes on the newly inserted line.
3. Move the new column to the first position:
4. Type the M (move) line command on the line defining the new column.
5. Type the B (BEFORE) line command on the line defining the first column.
6. Press Enter to perform the move.

This is the only way to insert a new first column in a table. If done incorrectly, data could be lost.

Specifying Default Values for New Columns (V)

The Column Default Value screen allows specification of a default value for a new column.

More information:

Column Default Value Screen (see page 347)

Repeating a Column

The **R** (REPEAT) line commands inserts a new column after the original column, with this new column initially showing the same column name and attributes as the original column. The data remains with the original column. Because DB2 will not allow two columns with the same name, the name of one of them must be changed.

If the name of the second (new) column is changed, the data remains with the original column, and the new column contains null values until populated.

If the name of the first (original) column is changed, so that the second (new) column retains the name of the original column, it is treated as a rename of the original column. The data remains with the first (original), but now renamed, column, and the second (new) column contains null values until populated.

The R (REPEAT) line command can be avoided. Use the I (INSERT) line command, where there is no confusion over which is the new line.

After altering a table, always issue the COMPARE command to verify the alterations.

Toggle and Scrolling Commands

The following types of commands can be entered to make working with tables easier:

- Toggle commands (see page 399)
- Scrolling commands (see page 400)

Toggle Commands

This section describes the commands you can use to toggle the display:

HEADER or H

Toggles the screen header on and off. This is particularly useful in split screen mode.

COMPARE or C

Enables comparison of the current version of the table with the table as it existed at the beginning of the session. Toggle this information on and off by entering C in the command line.

Scrolling Commands

The following commands control the scrolling of the column definition section on the table screen.

Note: For more information about the scrolling commands, see the *CA Database Management Solutions for DB2 for z/OS General Facilities Reference Guide*.

EXCLUDE

Excludes an attribute from the column list on the Table Screen.

SHOW

Displays the column attribute again that was excluded. This can be helpful to limit the column attributes displayed on the screen.

FIND

Finds a string within the column list. FIND can be column specific or global.

RFIND

Repeats the previous find. RFIND can be column specific or global.

FREEZE

Lets users freeze attribute columns for horizontal scrolling. The CMD, ### (column number), and COLUMN NAME attributes are automatically frozen.

MELT

Removes column freezes.

UNFREEZE

Removes column freezes.

SCROLLB

Sets the horizontal scrolling mode to BYTE.

SCROLLC

Sets the horizontal scrolling mode to COLUMN (the default).

QPRINT

Prints the table's column list.

PPRINT

Prints the columns displayed on the current page.

SORT

Is not functional for the column list.

DDL Execution

When F3 (End) is pressed from the Create, Alter, Template or Drop screen, a confirmation screen appears before the DDL is executed. Accept, edit, or reject the DDL to be used to complete the request.

Note: For more information, see the “Using CA RC/Update” chapter.

If your changes are complex, require dropping and recreating objects (and their dependencies), or the current CA RC/Update Operation Mode is set to A (RC/Alter), you can perform a complete analysis of your changes that includes rebuilding the DDL and optionally including utility statements and extended DDL generation and analysis options.

Confirmation Options

A Confirmation screen appears before the DDL is executed.

Press Enter to execute the displayed DDL (online mode), write it to a data set (batch mode), or pass to RC/Alter and CA RC/Migrator to control DDL statement generation (RC/Alter mode). During the online DDL execution, the Audit Message File appears. This screen displays completion messages and return codes. If there are errors, all statements are rolled back. Use the scrolling keys to review all messages. Press F3 (End) or Enter to return to the previous screen.

Enter **E** (Edit) to edit the generated DDL statements using the CA RC/Migrator DDL Edit Facility.

To refrain from processing the DDL, press F3 (End). The previous screen reappears, where changes can be made to the request. Enter the CANCEL command to cancel the option and return to the Main Menu or process the next items for Alter and Template options.

Chapter 19: Indexes

This section contains the following topics:

Features for Maintaining Indexes

You can use the index object options to create, template, alter, and drop indexes. The index option includes the features:

- DB2 Object Selection Help. When defining an index, the object names of the index's table and database must be specified. Help is provided for entering DB2 object names. Instead of entering a specific object name, selection criteria can be entered.
- Column Selection Help. Specify the columns for the index by selecting them from a list. All columns from the selected table are displayed. Specify the columns and their order within the index on a single screen.
- Limit Key Help. When creating a partitioned index, the limit must be specified for at least the first column in the key for each partition. This lets you specify the limit key value on a separate screen.
- Help for conversion from index-controlled partitioning to table-controlled partitioning. You may have DB2 convert a table object that is using index-controlled partitioning to use table-controlled partitioning. To do so, you can alter the table object's clustering index to be NOT CLUSTER or create a partitioned index or data-partitioned secondary index (DPSI) using ALTER or CREATE statements.

  Note: For information about table-controlled partitioning, see the IBM document *DB2 Universal Database for z/OS Administration Guide*.

  Storage Group and VCAT Help. Help is also provided for entering VCAT and storage group names. Instead of entering a specific name, selection criteria can be entered to receive a list of objects meeting the criteria.
- Space Calculation Assistance. A sophisticated space calculation facility is available to assist in calculating space allocations for the indexspace. If the indexspace is partitioned, space calculations can be made on each partition. Once complete, the space information is passed back to the indexspace screen.
- Space Locator. The Space Locator facility helps locate a volume or STOGROUP with adequate free space for indexspaces.
- VSAM Define Screen. To specify VSAM data sets rather than use a storage group, the data set name and type must be specified and any required passwords supplied. A VSAM Define screen is provided to assist in defining VSAM data sets.
- Alteration Support. DB2 permits only certain changes to indexspaces.
  If the requested changes are not supported by DB2, RC/Alter is invoked to drop and re-create the indexspace. All dependents, data, and authorizations are automatically restored and all changes are automatically propagated to any dependent object types. This extremely powerful feature is transparent to the end user. The user makes the request for the change, and the change is made using the appropriate methods.

Access Index Functions

Index functions can be accessed by requesting to create, alter, template, or drop an index.

To access index functions, perform one of the following:

- Enter **C** in the Option field and **I** in the Object field of the main header to create an index.
- Enter **A**, **T**, or **D** in the Option field and **I** in the Object field of the main header to alter, template, or drop an index. Enter an index name in the Item Name field and a user ID in the Creator field to get a more specific selection list. Select the indexes you want to alter, template or drop.

The appropriate index screen appears.

Index Selection Panel

When altering, dropping, or creating (by template) an index, a selection panel is presented for selecting the indexes if not already specified. Make the index selection list more specific by specifying selection criteria for the Item (index) Name and Creator ID prompts in the header. Use the EQF facility to further specify the indexes displayed.

Note: For more information about EQF, see the *General Facilities Reference Guide*.

The selection panel displays most of the fields available in the SYSIBM.SYSINDEXES index to make selection of the indexes easier. All selection panels support the standard scrolling commands.

Note: For more information about these fields, see the online help.

Create Index Option

The Create Index option lets you create an index (and implicitly, indexspace). Indexes can be created for existing tables or for tables that have not been created. When the Index Create screen appears, an index name and creator must be specified. The name and creator of the table for which the index is to be created must then be specified. If you have specified a name for your index, and a valid table name and creator are entered on the Index Create screen, the Column selection panel for that table appears. If search criteria are entered, a table selection list appears.

Once a table is selected, a list of columns is presented for that table. Select the columns and their order in the index. If creating an index for a table that has not been created, columns cannot be selected from a list; it must be entered manually. Once the information is complete, review the DDL and decide whether to process the create, edit the DDL, or cancel the create.

If a table has been created in the current session (batch mode), but the DDL has not been executed, an index can be created based on any of these tables. The table information is retained, enabling tables and index creation in the same batch job. Tables created in batch mode in the current session are listed at the top of the Table selection panel and CREATE appears in the Database column.

Note: The Item Name and Creator fields of the header pertain to the index, not to its base table.

Enter information for tables on which indexes should be defined in the Table Name and Creator fields of the Index Create screen. If a valid table name and creator ID are entered, the column selection panel for that table appears. If valid selection criteria is entered in the Table Name and Creator fields, an Index Table selection panel appears.

An index can be partitioned or non-partitioned. If the index refers to a table stored on a partitioned tablespace, the index must also be partitioned. Non-partitioned indexes can be clustered.

More information:

Index Selection Panel (see page 405)

Create a Non-Partitioned DB2 Index

You can create a non-partitioned DB2 index.

Follow these steps:

1. Access the Index Create screen.
2. Enter data in the following fields to define some basic information:
   - Index Name (and its corresponding Creator field)
   - Table Name (and its corresponding Creator field)

   Other fields contain default specifications.

   Note: Set defaults through the DEF command for all fields except Index Name, Table Name, and Partitioned. For more information, see the “Object Definition Defaults” chapter. For information about the fields, press F1 (Help).

   Press Enter.
3. The Index Column Selection & Key Maintenance screen appears.
4. Select the columns that compose the index key.
5. Enter storage related information about the indexspace in the Partition Information fields.

   Because this is a non-partitioned index, only enter information on the first line. A VCAT or storage group must be entered to override the storage group default (SYSDEFT).

   These fields are optional. Because the indexspace is not partitioned, the only valid line commands are V (VSAM Define), S (Space Calculation), and U (Undo any changes, set back to old definition).

   Note: You can set defaults through the DEF command for all fields except Erase. For more information, see the “Object Definition Defaults” chapter.

   Press F3 (End) to process the creation.

Create a Partitioned DB2 Index

A partitioned index is used whenever the table to which it refers is stored on a partitioned tablespace. A partitioned index must also be a clustering index. You can use this procedure to create a partitioned DB2 index.

Follow these steps:

1. Access the Index Create screen.
2. Enter data in the following fields to define some basic information:
   - Index Name (and its corresponding Creator field)
   - Table Name (and its corresponding Creator field)

   Other fields contain default specifications.

   Note: Set defaults through the DEF command for all fields except Index Name and Table Name. For more information, see the "Object Definition Defaults" chapter. For information about the fields, press F1 (Help).

   Press Enter.

   The Index Column Selection & Key Maintenance screen appears.
3. Select the columns that compose the index key, then press Enter.
4. Enter storage related information about each partition in the Partition Information fields.

   You must enter a VCAT or storage group if you do not want to accept the storage group default (SYSDEFT).

   Note: Set defaults through the DEF command for all fields except Part and Erase. For more information, see the "Object Definition Defaults" chapter.

   Press F3 (End).

   The Index Partitions Limit Key Values screen appears.
5. Enter a limit key value for the first partition, then press Enter.

   The information is processed, and the next partition is presented.
6. Press F3 (End) (after information has been entered for all partitions).

   The creation is processed.

More information:

Index Partitions Limit Key Values Screen (see page 417)
Index Column Selection & Key Maintenance (see page 413)

Maintain Comments in an Index

Whenever you create, alter, or template an index, you can also create and maintain comments for the index.

Follow these steps:

Perform one of the following:

- Create a comment for an index by entering **Y** in the Comment field.

  The Index Comment panel appears.
- View or update an index's existing comment by entering **U** in the Comment field.

  The Index Comment panel appears.
- Delete an index's comment by entering **N** in the Comment field.

  Any existing comments are removed.

Comments can be up to 254 characters long. Adding a comment inserts the COMMENT ON INDEX statement into the generated DDL.

Index Create Screen

The Index Create screen permits entry of information about the index and the (implicitly created) indexspace.

Note: For information about the fields, press F1 (Help).

Changing Partition Information for an Indexspace

If you specify partition information and the indexspace is changed to non-partitioned, all lines except the first are deleted. If the indexspace is returned to a partitioned indexspace, the old partition values are returned.

More information:

Alter Index Option (see page 415)

Manipulating Partition Information

Note: You can use standard ISPF line commands to create a number of partitions quickly (copy, delete, move, and insert). After copying a partition, change the information necessary to complete its definition.

To manipulate the partition information on the Index Create, Index Alter, or Index Template, enter a standard ISPF line command or a line command pertaining to a partition (for partitioned indexspaces) or indexspace (for non-partitioned indexspaces).

The following commands are supported for partitioned and non-partitioned indexspaces except as noted:

A

Places the partition to be copied or moved *after* this line. This command applies to partitioned indexspaces only.

B

Places the partition to be copied or moved *before* this line. This command applies to partitioned indexspaces only.

C, C*nn*, or CC

Copies this partition, number of partitions, or block of partitions to a location marked by A or B. These commands apply to partitioned indexspaces only.

D, D*nn*, or DD

Deletes this partition, number of partitions, or block of partitions. These commands apply to partitioned indexspaces only.

G

Allows you to view or enter Group Bufferpool Cache information for this partition. When specified, the Group Bufferpool Cache Setting screen appears.

I or I*nn*

Inserts a partition or number of partitions after this partition. These commands apply to partitioned indexspaces only.

L

Edits the limit key values for this partition. When specified, the Index Partitions Limit Keys Values screen appears, which displays all columns for the partition horizontally across the screen. To enter limit key values for a specific partition, enter LIMITS on the command line, and press Enter.

Note: Only the partition on which you entered the command is unprotected. All other partitions are automatically protected. To unprotect all other columns on all other partitions, enter the UNPROTECT * command on the primary command line.

LC

Generates and browse IDCAMS LISTCAT for VSAM data set.

Note: For more information, see the IBM IDCAMS documentation.

M, M*nn*, or MM

Moves this partition, number of partitions, or block of partitions to a location marked by A or B. These commands apply to partitioned indexspaces only.

R, R*nn*, or RR

Repeats this partition, number of partitions, or block of partitions after this partition. These commands apply to partitioned indexspaces only.

RES

Resets the partition to its old definition.

S

Performs space calculations for this partition. When specified, the Space Calculation screen appears.

SO

Performs space calculations for this partition. When specified, the Options screen appears, which lets you change the space calculator options before going to the space calculator.

SRC

Sets all values for this partition to its source definition values. On a partitioned index, the limit values for all columns in the partition are also reset. This command is valid for compares only.

TRG

Sets all values for this partition to its target definition values. On a partitioned index, the limit values for all columns in the partition are also reset. This command is valid for compares only.

U, U*nn*, or UU

Undoes this partition, number of partitions, or block of partitions and return them back to their old definitions. This command performs the same function as the RES command.

V

Edits the VSAM cluster definition for this partition.

More information:

Index Partitions Limit Key Values Screen (see page 417)
Define a VSAM Cluster (see page 424)

Group Bufferpool Cache Setting for Indexspace Partitions

When you issue the G line command next to an indexspace partition on an index create, alter, or template panel, the Group Bufferpool Cache Setting panel appears.

To specify what pages of the indexspace are written to the group buffer pool in a data-sharing environment, enter one of the following values in the GBP Cache for partition field:

A

Caches both changed and unchanged pages in the group buffer pool.

C

Caches only changed pages in the group buffer pool. This is the default.

N

Does not cache any pages in the group buffer pool. DB2 uses the group buffer pool only for cross-invalidation. If you specify N, the tablespace or partition must not be in a rebuild pending status and must be in the dropped state.

Note: In a non-data sharing environment, you can specify this information, but it is ignored.

Index Table Selection Panel

The Index Table selection panel appears when creating a new index, or when selection criteria are entered in the Table Name field on the Index Create, Index Alter or Index Template screen. This screen lets you select the tables on which indexes should be created.

Tables created as part of the strategy appear at the top of the screen. For newly created tables, *CREATE* appears in the Database column. To view referential integrity relationships in the Status field, use the M (Max) scroll command (enter M in the command line and press F20 to scroll all the way right. This field indicates whether the table has a primary index already created.

Select the tables to work with by entering **S** next to the names. If you are creating an index, the Index Column Selection & Key Maintenance screen appears; otherwise the Index Template or Index Alter screen appears.

Index Column Selection & Key Maintenance

You can select, delete, move, and modify key columns and index keys quickly and easily from the Index Column Selection & Key Maintenance screen.

This screen appears when you:

- Enter **K**, **KA**, or **KB** line commands next to a key column on the Index Create or Index Alter screen. The A (after) or B (before) command is inserted automatically for you on this screen on the corresponding line from the Index Create or Index Alter screen. Select the columns that you want to insert after this key column from the Column Name Selection List.
- Enter the **KEY** primary command on the Index Create or Index Alter screen.
- Alter, Create, or template on the Index Create or Index Alter screens, and after you press Enter it is determined that key columns are defined for the index.

By default, the Column Name Selection list only displays the columns that are eligible to be selected as key-columns for your index. When you select a column, it is removed from this list and inserted into the Index Key list at the bottom of the screen.

Note: To display ineligible columns (although not selectable) as well as columns that are already part of the key, enter the **L** primary command to toggle the Column Selection List display mode.

More information:

Select Columns (see page 414)

Select Columns

You can select a column to work with on the Index Column Selection and Key Maintenance screen.

Follow these steps:

1. Enter one of the following in the **CMD** column under the Column Name Selection list:

   Snn

   Selects *nn* column names (including this one) and moves them to the location marked by A or B in the Index Key list. If *nn* is omitted, 1 is assumed.

   SS

   Indicates to use block select. Use paired SS commands to delimit a block of column names to move to the location marked by A or B in the Index Key list.

   Note: When using these commands, only eligible columns will be included in the selection. Columns that are displayed but are not selectable (marked by three dashes) are not included in the selection or the count specified by S*nn*.

2. Enter **A** (after) or **B** (before) in the CMD column under the Index Key list next to the Index Key where you want to place the column. All entries are put in the index in ascending order by default.

   If S and numbers are used in the same column selection, the columns selected with S will come first in the index key, in the order of their column number. The numbered keys will follow the S keys, in the order of the entered numbers.

   To insert blank key columns, specify **I** (insert) in the CMD column under the Index Key list.

   Note: For other valid line commands, see the online help.

   Press Enter.

Processing Considerations for Index Creation

The CREATE INDEX statement cannot be executed while a DB2 utility has control of the tablespace that contains the identified table. If the named table contains data, CREATE INDEX creates the index entries for it. If the table does not yet contain data, CREATE INDEX creates a description of the index; the index entries are created when the table is loaded.

Confirm the Index Creation

When all information has been entered, you can confirm the index creation.

Follow these steps:

1. Press F3 (End) to process the creation.

   A confirmation screen appears.

2. Accept, edit, or reject the DDL that will be used to create the index.

Template Index Option

The Template Index option lets you create a new index using an existing index as a template. In fact, a template session is actually a create session with an additional step: an index can be selected as a starting point. The other fields can be changed, but the name of the templated index must be changed.

The Index Template screen displays the values used to create the template index. Change the fields for the new index according to the procedures outlined in Create Index Option. Once changes have been made, press F3 (end) to process the template.

Note: To view the IDCAMS System Services LISTC Report and view space amounts for user-defined data sets, enter LC in the CMD field for the data set. For more information about the specifics of this report, see the IBM IDCAMS documentation.

Alter Index Option

The Alter Index option lets you alter any characteristic of the Index, even those to which DB2 does not allow changes. You can also make changes not allowed by DB2 by dropping and re-creating the index. The DB2 ALTER INDEX command is fully supported. The following information can be altered in DB2:

- Buffer Pool
- Close data set
- Free Page
- Percent Free
- Using (data set)
- Primary Quantity
- Secondary Quantity
- Erase Rule

- Padded
- Cluster

To alter an index, access the Index Alter screen and make changes according to the procedures outlined in Create Index Option. After you make changes, press F3 (end) to process the alter.

Processing Considerations for Index Alterations

When altering an index, take the following into consideration.

- An index cannot be altered while a DB2 utility has control of the index or its associated tablespace.
- A change to the CLOSE rule is effective immediately.
- A change to the buffer pool assignment has no effect on the index until the next time its data sets are opened.
- A change to FREEPAGE or PCTFREE has no effect until index entries are stored by a load operation or the index is reorganized.
- If altering the number of partitions, be sure to change the partitions in the tablespace first. DB2 does not allow changes to the index before the tablespace.

Online Schema Support Regarding Index Alterations

The product provides the following support for online schema evolution (extended ALTER functionality introduced in DB2 V8):

- Adding columns to an index using ALTER INDEX with the ADD COLUMN clause (instead of dropping and re-creating the index).
- Altering the PADDED attribute by either of the following in the analysis output:
  - ALTER INDEX *creator.indexname* PADDED
  - ALTER INDEX *creator.indexname* NOT PADDED
- Altering the CLUSTER attribute by way of the ALTER INDEX with the CLUSTER or NOT CLUSTER clause (instead of dropping and re-creating the index).

If you execute alterations that use ADD COLUMN, PADDED, or NOT PADDED, the index is left in Rebuild Pending (RBDP) status. If the REBUILD INDEX analysis option is set to Y, REBUILD INDEX statements are included in the analysis output DDL file. Execution of the ALTER INDEX NOT CLUSTER statement can lead to table-controlled partitioning if the index is the partitioning index for a table that uses index-controlled partitioning.

Convert a Non-Partitioned Index to a Partitioned Index

You can change an index from non-partitioned to partitioned.

Follow these steps:

1. Access the Index Alter screen.
2. For Partitioned, specify YES.
3. For Cluster, specify YES.
4. Use the R, Rn, or RR commands to repeat the first partition (in Partition Information section of screen) to give as many partitions as needed.
5. Change the partition information fields as necessary (PRIQTY, SECQTY, Frpage, and so on).
6. Specify Limit Keys for each partition.
7. Press F3 (End) to process the alter.

   A Confirmation screen appears. Accept, edit, or reject the DDL that is used to alter the indexes.

More information:

Index Partitions Limit Key Values Screen (see page 417)
Create Index Option (see page 406)

Index Partitions Limit Key Values Screen

The Index Partitions Limit Key Values screen contains a scrollable list of partitions with all key columns displayed horizontally across each row in the list. From this screen, you can modify, copy, repeat, move, and exchange limit values for your index between partitions.

Depending on how you accessed this screen, the header information displays a set of instructions for you to modify or enter limit key values.

- Enter limit key values for index partition number *nnn* below. A constant must be present for the first index key column of the partition. Unspecified columns will be defaulted by DB2. Press F3 after you have completed your changes.

  Note: Appears when you enter the **LIMITS** command from the Index Create, Index Alter, or Index Template screens.

- Enter limit key values. At minimum, a constant must be present for the first index key column of each partition. Unspecified columns are defaulted by DB2. Press F3 after you have completed your changes.

  Note: Appears when you are alter, create, or template an index and when you press F3 it is determined that no limit values exist for any of the index's columns for any partitions.

More information:

Modify Limit Key Values (see page 419)
Specify Limit Key Values (see page 421)

Limit Key Requirements

Note the following limit key requirements:

- Each limit key must have the same data type as its corresponding column.
- The precision and scale of a decimal limit key cannot be greater than the column's.
- The highest value of the key in any partition must be lower than the highest value of the key in the next partition.
- The highest value of the key in the last partition is always the highest possible value of the key.
- If changes are made to the key after defining limit keys, the limit key values are deleted, and must be redefined. A message appears accordingly.

Modify Limit Key Values

A constant must be present for the first index key column of each partition. Unspecified columns will be defaulted by DB2. Press F3 after you have completed your changes.

Follow these steps:

1. Enter **L** next to a partition on the Index Create, Index Alter, or Index Template screen.

   The Index Partitions Limit Key Values screen appears.

   Note: This screen can use MAXVALUES mode or VALUES ONLY mode. In MAXVALUES mode, you can manage MAXVALUE settings for key columns. In VALUES ONLY mode, you can manage the limit values for partitions, regardless of partition arrangement. For complete information about these modes (including changing modes), see the online help.

2. Enter values as follows in the CMD column for the partition's key columns. These columns are displayed, from left to right, in the order in which they were defined for the index. Each column type is type sensitive.

   A

   Replaces or exchanges the limit key values for the partitions after this line with the limit key values of the partitions selected with the copy (C, Cnn, CC) or exchange (X, X*nn*, XX) commands.

   B

   Replaces or exchanges the limit key values for the partitions before this line with the limit key values of the partitions selected with the copy (C, Cnn, CC) or exchange (X, *Xnn*, XX) commands.

   C, Cnnn, or CC

   Copies the limit key, number of limit keys, or block of limit key values to the location marked by A or B.

   PR, PR*nn*, or PPR

   Protects the limit key values from being directly or indirectly altered by user input or by certain commands. This command is similar to the PROTECT primary command, with the exception that it works on a row-by-row basis instead of a columnar basis. These settings stay in effect for the duration of the immediate editing session only and are discarded upon leaving the Index Partitions Limit Key Values screen.

   Note: For more information, see the online help.

R, R*nnn*, or RR

Repeats the limit key values of this partition into the partition after this partition.

CMAX

Clears all unprotected occurrences of MAXVALUE for this partition (replacing with blanks).

RES

Resets the limit key values of this partition to its old definition.

SMAX

Sets all unprotected limit values for this partition to MAXVALUE.

SMAB

Sets all unprotected blank limit values for this partition to MAXVALUE only.

SMAF

Sets only the first unprotected blank limit value for this partition to MAXVALUE.

SRC

Sets the limit key values of this partition to its source definition values. This command is valid for compares only.

TRG

Sets the limit key values of this partition to its target definition values. This command is valid for compares only.

UN, UN*nn*, or UUN

Unprotects the limit key values for the specified partitions so that they can be directly or indirectly altered by user input or through the use of certain commands. This line command is similar to the UNPROTECT primary command, with the exception that it works on a row-by-row basis instead of a columnar basis. These settings stay in effect for the duration of the immediate editing session only and are discarded upon leaving the Index Partitions Limit Key Values screen.

Note: For more information, see the online help

X, X*nnn*, or XX

Exchanges the limit key values for the specified partitions with the limit key values of the partitioned marked by A or B, or another block of identical size marked by another X, Xnn, or XX command.

3. Validate the partition key columns content by pressing Enter, F3, or End.

Specify Limit Key Values

When it is determined that no limit values exist for any of the index's columns for any partitions or you enter the LIMITS command on the Index Create, Index Alter, or Index Template screen, you can specify limit key values.

Follow these steps:

1. Do one of the following on the Index Partitions Limit Key Values screen:
   - Enter limit key values for index partition number *nnn* below.
   - Enter limit key values.

   Note: Minimally, a constant must be present for the first index key column of the partition. Unspecified columns will be defaulted by DB2. Press F3 after you have completed your changes.
2. Press F3 after you have completed your changes.

Protecting Limit Values

You can optionally protect limit values for one or more columns across all partitions with the PROTECT command. You can also protect limit values for all columns in a specific partition with the P line command so that they cannot be inadvertently modified when using certain data modifying commands, such as copy, exchange, and so on.

Automatic Partitions Management

Automatic partitions management is controlled using the AUTOPARTS primary command. The AUTOPARTS (or AP) command lets you automatically manage the displayed list of partitions on the Index Create, Index Alter, and Index Template screens such that the number of partitions displayed in the partition information fields always matches the number of partitions in the tablespace of the table for your index.

The AUTOPARTS option is only applicable for partitioned indexes, where:

- The tablespace of the table, for which the index is being altered, created, or templated, is known.
- The tablespace is partitioned.
- The number of parts in the tablespace is also known.

Note: By default, this option is off; however, you can enable it and save the settings so that is default setting is enabled or on.

AUTOPARTS Command Syntax

The AUTOPARTS command syntax follows:

```
AUTOPARTS ON|OFF|SAVE or AP ON|OFF|SAVE
```

ON

Enables automatic partitions management. The number of partitions in the tablespace of the table for the index determines the number of partitions that are required for your index. The number of partitions displayed on the screen is always equal to the number of partitions in the table's tablespace.

OFF

Disables automatic partitions management. You, not the tablespace of the table for the index, determine the number of partitions required for your index.

SAVE

Saves the current automatic partitions management setting. The saved setting is used as the default setting the next time you create, alter, or template a partitioned index. The saved setting is also remembered across multiple sessions.

Note: To display the current command setting, specify AUTOPARTS alone without any options.

Usage Notes

When automatic partitions management is enabled, a list of partitions that matches the number of partitions in the tablespace of the table for your index appears on the Index Create, Index Alter, or Index Template screens automatically.

When AUTOPARTS ON is specified, automatic partitions management is immediately enabled only when the index is partitioned and the tablespace for the table is known and is also partitioned. Otherwise, this option is temporarily disabled until these criteria are met.

Note: When AUTOPARTS is ON and blank partitions are interspersed through the list, you can automatically remove the blank partitions by specifying the AUTOPARTS OFF command followed by the AUTOPARTS ON command. The first command displays all blank and non-blank partitions, and then because the automatic partitions management option is off for a moment, all the blank partitions are automatically deleted the next time you press the Enter key, or before executing the second command, which turns the auto parts option back on.

Calculate Space Requirements for Indexes

The Space Calculator calculates the primary and secondary space quantities needed for your indexspaces. This eliminates time consuming calculations.

You can create a new stand-alone indexspace strategy that saves space requirements for an index. You can also see the affects of different sub-page and free space decisions on the index pages and levels.

Note: For more information about the Space Calculator, see the *General Facilities Reference Guide*.

Follow these steps:

1. Do *one* of the following:
   - Type **SS** (Space Calculator) in the Option field on the CA Database Management Solutions for DB2 for z/OS main menu and press Enter.
   - Type **SSC** from any product and press Enter.

   The SpaceCalc Strategy Select panel appears.
2. Do *one* of the following:
   - Type **CI**, the name of the strategy, the name of the indexspace, and the index name on the create line to create a new stand-alone indexspace strategy.
   - Press Enter to see a list of existing strategies and then type **E** next to a strategy you want to edit.

   Press Enter.

   The Index Strategy panel appears.

   Note: This panel has the same fields as the Index Alter, Index Create, and Index Template panels that can be accessed from CA RC/Migrator and CA RC/Update.
3. Select an index, specify a VCAT or storage group name in the VCAT or Stogroup fields, and press Enter.

   The Index Space Calculation panel appears and shows the index name, number of partitions, and key length. The control parameters default to the values last used on the Index Create and Index Alter panels.
4. Edit the index information fields and control parameters as needed and press Enter.

   Note: All fields must contain a value before the Space Calculator can calculate the results.

   The Summary and Detailed Space Statistics are calculated and updated to reflect your changes on the Index Space Calculation panel. The scrolling position remains when you press Enter so that you can make changes, press Enter, and immediately see the effects on all fields.

5. Press F3 (End) when you have completed your calculations.

   The Index Space Calculation panel is refreshed and shows the calculated space statistics.

   If you accessed space calculation from CA RC/Migrator or CA RC/Update, you are returned to the Index Create, Index Template, or Index Alter panel. The calculations are updated on these panels and the information is used for the selected indexspace or partition.

6. Type **S** and an eight-character strategy name or object link in the SSTRAT field to save the strategy and press Enter.

   Note: You can enter the name of a new strategy or reference an existing (previously saved) strategy.

   The SpaceCalc Strategy Save panel appears.

7. Press Enter to save the strategy with the specified attributes.

   The action code of the SSTRAT field is reset to N and you are returned to the Space Calculation panel. If you made changes to the statistics and saved them, an object-linked space strategy is created.

More information:

Locate Free Space (see page 335)

Define a VSAM Cluster

Define the VSAM cluster by accessing the VSAM Define screen. Use the Space Calculation screen before defining the VSAM data sets, as the space statistics calculated are saved and brought to this screen. If the Volume Space Locator was used, any VOLSER selections made will be used to define the VSAM cluster.

Follow these steps:

1. Enter the VSAM line command (**V**) in the line command area next to the partition or index whose VSAM cluster to define.

   The VSAM Define screen appears.

   Note: For information about the fields, press F1 (Help).

2. Enter all necessary information, then press F3 (End) to process the information.

   You are returned to the index Create, Template, or Alter screen.

Auxiliary Indexes

If a column is defined as LOB in a table, it requires its own auxiliary tablespace and index.

When you select an auxiliary table, the appropriate Auxiliary Index screen (create, alter, or template) appears. An informational message appears indicating that the index you are creating is on an auxiliary table.

Note: Because indexes on an auxiliary table are implicitly unique, the Unique Rule field is always set to YES and cannot be changed.

The Aux Index screens let you alter, create, and template DB2 indexes on auxiliary tables. An index's primary purpose is to provide more efficient access to a table.

As with indexes on regular tables, you can:

- Apply user defaults to the index definition using the Apply Def command.
- Display the original and current definitions of an object using the Compare command.
- Enter selection criteria in many fields to bring up lists of objects to choose from.

  Note: For more information about these fields, see the online help.
- Create an explicit VSAM data set for the index through the VSAM definition facility.
- Estimate the space required by an index using the space calculation feature. You can also see the affects of different sub-page and free space decisions on the index pages and levels.

BUSINESS_TIME WITHOUT OVERLAPS Clause

You can create indexes on application temporal and bi-temporal tables with the BUSINESS_TIME WITHOUT OVERLAPS clause included. When the table for the index is a temporal table defined with a BUSINESS_TIME period, set the BTWOO option to YES. The BUSINESS_TIME WITHOUT OVERLAPS clause is inserted as the last item in the index key automatically.

When the index is defined using the Index on Expressions panels, enter BUSINESS_TIME WITHOUT OVERLAPS as the last key-expression for your index.

Note: When BUSINESS_TIME WITHOUT OVERLAPS is included, all key-expressions must specify simple column names.

Data-Partitioned Secondary Indexes (DPSIs)

On a partitioned tablespace, you can implement a data partitioned secondary index (DPSI) that clusters the data within each partition. The use of a DPSI can reduce or eliminate contention between jobs that target different partitions of a tablespace.

Note: Because it is a partitioned index, a DPSI can be created only on a partitioned tablespace. For information about DPSI restrictions and for general information about secondary indexes, see the IBM *DB2 Universal Database for z/OS Administration Guide*.

Create a DPSI

A DPSI can only be created for a table that is defined in a partitioned tablespace that is using TCP or ICP. If the table is using ICP, creating a DPSI for the table will cause DB2 to implicitly convert the table so that it uses TCP. To create a data-partitioned secondary index:

Follow these steps:

1. Go to the Index Create screen.
2. Enter a name and creator for your index in the Index Name and Creator fields if not already done so. Do not press Enter yet.
3. Verify that the Partition option is set to **YES**. Do not press Enter yet.
4. Enter a valid table for your index in the Table Name and Creator fields. *(The table must* be *defined in a partitioned tablespace and must be using TCP or ICP.)* Press Enter.

   The Index Column Selection & Key Maintenance screen for that table should display.

   Note: If selection criteria are entered and more than one table matches the selection criteria, the Table selection panel displays with a list of all matching table objects from which you need to select the table for your index; otherwise, the table is auto-selected and the Index Column Selection & Key Maintenance screen displays.

5. Select the columns and their order for the index key. When you are done, enter **END**.

   You are returned to the Index Create screen. The columns that you selected for your index's key should display in the Key-Column List at the top of the screen.

   Important! It is crucial that your key is not a matching or superset of the table's current partitioning or clustering index key; otherwise, a type-P index will be assumed instead of a type-D.

   If the table is using TCP, the PS column in the Column Name Selection list will show the partitioning sequence number of the column within the table's current partitioning key.

   If the table is using ICP, PS will be filled in *after* selecting your key columns and will display only in the Index Key list containing the key you are constructing, thus showing you the relationship of the DPSI key columns with that of the table's (yet to be created) partitioning key after being converted to use TCP by DB2 when the generated DDL is finally executed and the DPSI created.

6. Verify that the Partitioned option is set to **YES** and that the number of partitions correctly displays for the table's tablespace to the immediate right of the Partitioned field. The Partition List at the bottom of the screen should display an initial list of 10 empty (defaulted) partitions.

   Note: If AutoParts is ON, the Partition List will initially display a number of partition lines equal to the number of partitions in the table's tablespace.

7. Verify that the index type is **D**.

   If the index type is not D, even after verifying that the Partitioned option to YES, then it is likely that you created a key that is either a matching or superset of the table's partitioning key (if the table is using TCP) or the table's type-2 clustering index (if the table is using ICP.) Make corrections directly on the Index Create screen in the Key-Column List; or, enter the **KEY** command to display the Index Column Selection & Key Maintenance screen and make the corrections there.

8. Enter the indexspace information for at least one partition in the Partition List at the bottom of the screen. You may enter selection criteria in VCAT or STOGROUP to display a selection list of available VCAT or STOGROUP names to choose from for each indexspace partition. You may also use some common ISPF commands (such as A, B, C, D, I, M, and R) to copy, delete, insert, move, and repeat partitions. To see all available commands and descriptions of each, see the online help and tutorials.

   Notes: VCAT or STOGROUP must be specified for the other fields to retain their values; otherwise, they will automatically reset to their default values each time Enter is pressed.

   You need only enter indexspace information for one partition if all partitions for your DPSI are to be defined the same. If you define more than one partition, the first most common definition will be used for the global using block in the generated DDL.

   Unless overridden using Object Definition Defaults, if you leave all partitions blank without specifying either VCAT or STOGROUP, one screen display partition will be automatically defined for you using the following defaults which will then be used for the global using block in the generated DDL:

   - VCAT will be set to blanks.
   - STOGROUP will be set to SYSDEFLT.
   - PRIQTY will be set to DEFAULT.
   - SECQTY will be set to DEFAULT.
   - ERASE will be set to NO.
   - FRPAGE will be set to 0.
   - %FR will be set to 10.

9. (Optional) Modify any other fields on the screen as needed. When you are done, enter **END**.

   The CA RC/Update Creation Confirmation screen displays with the generated DDL for creating your index.

10. Review the generated DDL displayed on the CA RC/Update Creation Confirmation screen. From this screen, you may immediately execute the DDL online through the Batch Processor, edit the generated DDL, CANCEL and return to the Index Create screen; or, depending on the CA RC/Update Profile settings for your user ID, build the DDL with RC/Alter so you can specify utilities and extended DDL generation and analysis options.

More information:

Automatic Partitions Management (see page 421)
Access Index Functions (see page 405)

Create and Manage an Index on Expression

An index on expression provides a means to improve the processing of queries. The expressions that you specify transform key values so that they are not exactly the same as values in the table columns. An index on a general expression can enhance query performance by providing a more efficient index for the optimizer to use when selecting an access path.

Note: For general restrictions and limitations when using an index on expression, see the IBM *DB2 9 SQL Reference Guide*.

You can create and manage an index on expression. During the index creation or alteration, you can update the keys that contain scalar expressions for the index. Each expression must be unique (and duplicate expressions are not allowed). You can also drop the index if necessary.

Note: An index on expression cannot be partitioned. Additionally, you cannot convert an auxiliary index into an index on expression.

Follow these steps:

1. Access the index create, alter, or template panel, and enter **E** (EXPRESSIONS) in the command line.

   The Index on Expression version of the panel appears.

   Important! At least one KEY-EXPRESSION field must define a scalar expression; otherwise, DB2 creates a simple index on table columns.

2. (Optional) Enter any of the following commands in the command line (to toggle the feature between active and inactive):

   CAPS

   Activates or deactivates automatic uppercase. When CAPS is active, all alphabetic characters are automatically converted to uppercase.

NULLS

Activates or deactivates treatment of space at the beginning or end of lines as null. When NULLS is active, space at the beginning or end of the line is treated as null. When NULLS is inactive, space at the beginning or end of the line is treated as blank spaces.

PARSE

Activates or deactivates the automatic parsing of expressions. When PARSE is active, expressions are automatically parsed for @ symbols when you press Enter or PF3 (End). The symbols serve as markers for where you want to substitute column names. When a symbol is detected, a selection list automatically appears, from which you can select a column name to substitute for the symbol. This selection ability is useful so that you do not have to remember the names of the columns in the table.

The product indicates the status of each command at the top of the panel. For example, if all of these commands are active, CAPS NULL PARSE appears in the top right-hand corner of the panel. The command name is visible only if active.

3. Edit expressions as needed, or use line commands in the Cmd field to define or manipulate expressions.

   For example, you can use line commands to insert a new key expression or replicate an existing expression. You can also include @ symbols as placeholders in your expressions (to be replaced later by table columns that you select).

   Expressions change according to your customizations. At any time, you can issue the RES line command next to an existing expression to reset the expression to its old definition.

   Note: For complete information about available line commands, see the online help.

4. (Optional) Enter the following command next to any expression to replace @ symbols with table column names:

   P*nn*

   Parses *nn* expressions for @ symbols and lets you select table column names to replace the symbols. If you do not specify a range (*nn*), 1 is assumed, and a new selection list appears for each detected @ symbol. You can specify a block form of the command (PP/PP).

   Note: If PARSE is active, the product ignores the P*nn* command. With PARSE active, a single selection list is invoked when you press Enter of the END key (to handle all expressions at the same time). To activate or disable PARSE, use the PARSE primary command.

   Symbol replacement occurs as you make selections.

5. Press the END key when you are finished, and press Enter to confirm that you want to perform the requested activity.

   The product processes the request.

More information:

Access Index Functions (see page 405)
PARSE Command—Enable or Disable Search for @ Symbols in Expressions (see page 432)

Convert an Index on Expression to an Index without Expressions

You can convert an index that contains expressions into an index that contains no expressions in its definition. This is useful if changes to your database or table column data have made the existing expressions no longer applicable.

Note: Toggling between the index types does *not* discard any defined features that are specific to one type. For example, toggling from a key column index to an index on expression does not discard the defined key columns. If you change the index back to a key column index, the key columns reappear.

Follow these steps:

1. Access the index create, alter, or template panel, and enter **E** (EXPRESSIONS) in the command line.

   The regular index version of the panel appears.

2. (Optional) Specify values for Partitioned and Cluster.

   If you do not make an update, these fields retain their values from the index on expression definition.

   Values change according to your edits.

Convert an Index without Expressions to an Index on Expression

You can convert an index that contains no expressions into an index that contains expressions in its definition. An index on a general expression can enhance query performance by providing a more efficient index for the optimizer to use when selecting an access path.

Note: An index on expression cannot be partitioned. Additionally, you cannot convert an auxiliary index into an index on expression.

To convert an index without expressions to an index on expression, access the index create, alter, or template panel, and enter the following in the command line:

E

Specifies to create an index on expression (or change an index without expressions to an index on expression). The E entry is the short form of the EXPRESSIONS primary command. The command toggles the index between formats if you enter it multiple times.

The Index on Expression version of the panel appears.

PARSE Command—Enable or Disable Search for @ Symbols in Expressions

The PARSE command activates or disables the automatic parsing of the editable area of all key expressions. Entering PARSE in the command line of an index on expression panel toggles automatic parsing.

When automatic parsing (PARSE) is active, the product parses all expressions for @ symbols when you press Enter or PF3 (End) on any index on expression panel. The @ symbols are markers for where you want to substitute column names. When the product detects a @ symbol, a column name selection list automatically appears, from which you can select the column names to substitute.

Note: Any @ signs that are enclosed in quotation marks or embedded in a string are *not* recognized.

When PARSE is active, parsing occurs *before* the processing of REPEAT, COPY, and MOVE line commands (R, RR/RR, C, CC/CC, M, and MM/MM). When PARSE is inactive, no automatic parsing occurs, and @ symbols remain unaltered in the expression. You can use the SAVESETTINGS (SS) primary command to make the current PARSE setting the default setting for each time you return to the panel.

The product indicates the status of the PARSE feature at the top of the panel. The name (PARSE) is visible only if active.

Note: For precise control over parsing individual expressions, disable the PARSE feature, and enter the P line command beside the specific expressions that you want to parse for @ symbols. To disable PARSE, enter the PARSE primary command, and ensure that *PARSE* no longer appears at the top of the panel. For more information about the P line command, see the online help.

This command has the following format:

```
PARSE [ON|OFF]
```

ON

Activates the PARSE feature.

OFF

Disables the PARSE feature.

Example: Replace @ Symbols with EMPLOYEE Column Name

This example substitutes column name EMPLOYEE in place of the @ symbol (where permissible) in the following expressions:

```
UPPER(@,' ')
UPPER( @ , ' ' )
LOWER( MAIL_NO@ , ' ') CONCAT '@' CONCAT COMPANY CONCAT '.COM'
```

If you use the PARSE primary command (to activate the PARSE feature), press Enter or PF3 (End), and choose the EMPLOYEE column, the product updates the expressions as follows:

```
UPPER(EMPLOYEE,' ')
UPPER( EMPLOYEE , ' ' )
LOWER( MAIL_NO@ , ' ') CONCAT '@' CONCAT COMPANY CONCAT '.COM'
```

In the first and second lines, substitution is successful. Only the @ symbol is affected, and no spaces are inserted or taken away from the original expression. In the third line, no substitution occurs, because the @ symbol is either embedded in a string or enclosed in quotation marks.

Example: Active and Inactive PARSE Status

This example shows text at the top of an index on expression panel, indicating that PARSE is active:

```
Index on Exp Alter ---- CAPS NULLS PARSE
```

This example shows text at the top of an index on expression panel, indicating that PARSE is *not* active:

```
Index on Exp Alter ---- CAPS NULLS -----
```

EXPRESSIONS Command—Change Index Type Between Expressions and Key Columns

The EXPRESSIONS command toggles the index type between an index defined on scalar expressions and an index defined on key columns. When you toggle between types, panel options change to accommodate the type.

When you enter the command for a key column index, the list of key columns changes to a list of key expressions.

Note: Toggling between the index types does *not* discard any defined features that are specific to one type. For example, toggling from a key column index to an index on expression does not discard the defined key columns. If you change the index back to a key column index, the key columns reappear.

This command has the following format:

```
EXPRESSIONS [ON|OFF]
```

ON

Sets the index type to *on scalar expressions*.

OFF

Sets the index type to *simple on key columns*.

More information:

Convert an Index on Expression to an Index without Expressions (see page 431)
Convert an Index without Expressions to an Index on Expression (see page 432)

Index Drop Considerations

Under the rules of DB2, an indexspace cannot be dropped if the index is defined on a table in a partitioned tablespace. The partitioned tablespace is automatically dropped before dropping the index.

Note: For information about how to drop an object with the various drop options (S, SO, R, RO), see the “Using CA RC/Update” chapter.

If the index selected to be dropped contains a unique constraint the CA RC/Update Default Drop Analysis screen appears. From this screen you can:

- Launch CA RC/Migrator Analysis to generate the drop DDL.
- Specify the name of the data set to which the drop DDL is written.
- Specify batch, online, or RC/Alter processing.

DDL Execution

When F3 (End) is pressed from the Create, Alter, Template or Drop screen, a confirmation screen appears before the DDL is executed. Accept, edit, or reject the DDL to be used to complete the request.

Note: For more information, see the “Using CA RC/Update” chapter.

If your changes are complex, require dropping and recreating objects (and their dependencies), or the current CA RC/Update Operation Mode is set to A (RC/Alter), you can perform a complete analysis of your changes that includes rebuilding the DDL and optionally including utility statements and extended DDL generation and analysis options.

Chapter 20: Referential Integrity

This section contains the following topics:

Introduction to Referential Integrity

Tables or relations are made up of rows. Rows are made up of column entries. A column entry or the concatenation of more than one column entry can be called a key. Every table has inherently what are called candidate keys. A candidate key is a key that is unique to the table. For example, there is a table called EDIT_DEMO that comprises the following columns:

- NAME
- DESCRIPTION
- YEARS_ALIVE
- MINUTES_ALIVE
- SECONDS_ALIVE
- DATE_BORN
- TIME_BORN
- TIMESTAMP

Entries in the column DESCRIPTION will be unique. As a result, the DESCRIPTION COLUMN is a candidate key. For the purpose of this example, NAME cannot be a candidate key (there may be individuals with the same name). However, the concatenation of NAME and DATE_BORN is a candidate key, as no person shares a name and a birth date.

Select either one of the two candidate keys (DESCRIPTION and the concatenation of NAME and DATE_BORN) to be the primary key. The candidate key selected can be random. Choose the most important candidate key. In this example, the concatenation of NAME and DATE_BORN is probably the more logical choice for primary key. There is another Table called DEMO2, which contains the following columns.

- NAME
- SSN
- DATE_BORN
- SALARY

In this table, the only single column candidate key is SSN. Duplicate entries exists in NAME, DATE_BORN, and SALARY. (However, the concatenation of the columns NAME and DATE_BORN is also a candidate key.)

If the people in EDIT_DEMO are the same people in DEMO2, some information, such as NAME and DATE_BORN, should be consistent between the tables. Referential Integrity enables definition of the relationship between a primary key in one table (EDIT_DEMO) and a key in another table (DEMO2), called the foreign key. The foreign key must match the primary key's type and length exactly. Also, if a concatenation is used, the order of columns concatenated must be the same. The column names do not have to be the same.

Defining the relationship between two tables using primary and foreign keys is called making referential rules. Referential rules help ensure referential integrity. Once the relationship between the keys has been defined, DB2 will help ensure that changes to data are consistent between parent and child tables. (The table whose primary key is being used for reference is called the Parent Table. The table that calls upon this reference through a foreign key is called the Child Table. The parent and child are not necessarily different.) The Referential Integrity option permits assignment of primary and foreign keys and to define the relationship between the data in those keys.

Overview of Product Referential Integrity Features

The Referential Integrity functions provide a method for completing the task of creating unique constraints. (This can also be created as part of the Table Create or Table Alter process). Some of these Referential Integrity features include the following.

- Table selection panels. Table selection panels permit table specification for unique constraint creation without having to remember the creator ID and table name.
- Column Selection. The Create Referential Rules screen provides the user with column selection for easy selection of columns for unique constraints or primary keys. The order of the columns within a concatenation is specified simply by entering the column's order number.

- Unique Constraints Specification Help. When creating a foreign key, the user is assisted by the Foreign Specification screen. This screen prompts the user for parent table information. The user can specify a table explicitly or request a table selection panel. The user is also prompted for a rulename and delete rule information.
- Referential Rules selection panels. When requesting to drop referential rules, the user can select from a list of referential rules. The list of rules can be made more specific by entering primary table name and creator information or by using EQF.
- Model Command. The Model command permits you to change the display of the Referential Rules selection panel. Toggled on, the MO (model) command displays table name and creator information only once for each table. This assists in identifying quickly the referential rules created for each table.

Preliminaries

Before creating a referential rule using the RI option, you need to do the following:

- Set up a primary key (see page 439)
- Set up a foreign key (see page 439)

How to Set Up a Primary Key

To set up a primary key, do the following:

1. Create the Parent Table.
2. Make sure the columns used as a key is NOT NULL.
3. Create a unique index on the primary key specification for the Parent Table.

   Note: You cannot create a unique index on a materialized query table (MQT).

How to Set Up a Foreign Key

To set up a foreign key, do the following:

1. Create the Parent Table.
2. Create the Child Table.
3. Assign a Primary Key Specification to the Parent Table.

Accessing Referential Integrity Functions

Referential Integrity functions can be accessed by requesting to create or drop referential integrity rules. To create a set of referential integrity rules, enter C in the Option field and RI in the Object field of the main header and press Enter. The Create Referential Rules screen appears. To drop a set of referential integrity rules, enter **D** in the Option field and **RI** in the Object field of the main header and press Enter. The Drop Referential Rules screen appears.

Create Referential Rules

You can create and update referential rules from the Table Create, Template, and Alter options; and the RI option. This chapter addresses creating referential rules through the RI option.

The RI option treats referential rules as objects, rather than as attributes of a table. Therefore, referential rules are defined after the table is defined. Creating rules in this manner is much easier from the perspective of a user, because there are no worries about self-referencing problems and cyclical tables.

The Create Referential Integrity option performs all the SQL ALTER TABLE statements that refer to primary and foreign keys. When creating primary or foreign keys, select the table upon which keys are to be created from a Table selection panel. This selection list can be made more specific by entering selection criteria in the header fields or by using EQF.

After selecting the table to use, the Create Referential Rules screen appears, permitting specification of the columns to use for primary and foreign keys. When creating foreign keys, the Foreign Key Specification screen assists in specifying information about the parent table. A table selection list can also be requested at this time.

After all necessary information to create the keys has been supplied, the Confirmation screen appears. Accept, edit, or reject the DDL to be used to alter the table.

If the user accepts the DDL and is operating in Online mode, the DDL is executed online. If operating in Batch mode, the DDL is written to the specified data set.

Create Referential Rules Table Selection Panel

To create referential integrity for a table, enter **C** in the options field, and **RI** in the object field. The name (or portion of a name) of the table for which referential integrity should be created can be entered as well. Table Creator ID information can be included. EQF can also be employed at this point. Press Enter after the information has been entered.

The Create Referential Rules Table selection panel appears.

Note: For information about the fields, press F1 (Help).

Scroll right to display the following fields of the Table selection panel that are especially helpful when choosing tables upon which to create rules:

PARENTS

The number of tables referenced as parent tables by this table (TABLE NAME).

CHILDREN

The number of tables to which this table is a parent table. These tables are the children of this table.

KEYCOLS

The number of columns used in key definitions.

Scroll right (F11) to see the following column:

STATUS

This column indicates whether the table has a primary index already created.

Select the Table for which rules are to be created by entering **S** next to the name and pressing Enter. The Create Referential Rules Screen appears.

Note: Primary Keys and Foreign Keys can be created in the same session.

Creating Primary Keys

To create primary key rules for the table being displayed, enter **Pn** in the command line area, where *n* is the number that represents the position of the column within the primary key for a composite key.

If one column is the sole column for the primary key, enter **P1** next to that column. If the concatenation of two columns is being used to create the primary key specification, enter **P1** next to the first column and **P2** next to the second. The order of the number IS important, especially when later creating foreign keys.

Once primary key specifications have been entered, press F3 (End). A Confirmation Screen will appear. Accept, edit, or reject DDL to be used to alter the table.

Primary Key Rules

When creating Primary keys, keep the following rules in mind.

- There can be only one primary key defined on a table. The key must be unique and have an index.
- During an Edit session, a primary key value cannot be updated if there are any dependent rows tied to the primary key value. RC/Edit protects the primary key columns.
- When INSERTing a row into a dependent table, all foreign keys must match primary keys in the parent tables (unless one of the foreign key columns is NULL, in which the whole foreign key is considered NULL).

In the case of a self-referencing table, a row can be inserted into a dependent table where the primary key does not exist for the specified foreign key. The required primary key value must be in the same row as the foreign key value. In a DEPARTMENT table that contains a ADMINDEPT foreign key that references the primary key column DEPT in the same table, a row can be inserted with the DEPT column equal to the ADMINDEPT column.

Creating Foreign Keys

To create foreign key rules for the table being displayed, enter **Fn** on the command line, where *n* is the number that represents the position of the column within the foreign key for a composite key. If the foreign key is a composite key, the order of the columns in the foreign key specification must be the same as the order specified in the primary key specification of the parent table. Once Fn has been entered next to all columns that will make up the foreign key, press F3 (End). The Foreign Key Specification Screen appears.

Note: This version does not support the NO ACTION delete rule when creating or altering tables with referential constraints. If a table is altered that has NO ACTION defined as a delete rule, it is translated to RESTRICT. RESTRICT appears instead of NO ACTION and is used in its generated DDL. We do not provide notification of referential constraints defined with the NO ACTION delete rule.

To use the NO ACTION delete rule, an alternate method for applying the delete rule exists. To define a referential constraint with the NO ACTION delete rule, use RC/Alter to define the delete rule as RESTRICT. After the DDL is generated, use the EDIT command to manually edit the DDL to change the RESTRICT delete rule to NO ACTION. Note, however, that CA does not recommend this action because undesirable or unpredictable results may occur.

Once foreign key specification information has been entered, press Enter to process the information. The Create Confirmation screen appears. Accept, edit, or reject the DDL to be used to alter the table.

Foreign Key Rules

When creating foreign keys, keep the following foreign key rules in mind.

- Only one foreign key can be created for a table during a session. To create more foreign keys on one table, begin the Create Referential Integrity session again.
- If the Tablespace that contains the foreign key table has ever contained data, creating foreign keys puts the Tablespace in a CHECK PENDING mode. This means that the Tablespace is locked until the check utility is run.
- To add a foreign key, the user must have both normal authorities on the dependent table and the ALTER privilege on the parent table.
- The columns referenced in the foreign key statement must already exist in the parent table. Also, they must be in the same order. The names and NULL attributes can be different.

The foreign key can contain columns not in the primary key (but these must follow the primary key columns).

- Two foreign keys (on the same table) cannot reference the same primary key.
- To increase performance, an index should be created on a foreign key. If the columns of the foreign key are the same as the primary key (for that table, not the parent table), then the primary index will be shared by the foreign key.
- When ADDing a foreign key (ALTER statement), all plans that reference the table are invalidated. If the delete rule for the foreign key is CASCADE or SET NULL, then all plans that see the parent table, and all plans that refer to tables from which deletes cascade to the parent table, are invalidated.
- If a dependent table can be accessed from the same parent by more than one path, then all DELETE rules must be the same for that dependent. This is an important rule because it prevents DB2 from creating DEADLOCK conditions.
- If a referential table cycle is defined, one of the delete rules in the table cycle cannot be CASCADE. If only two tables define the cycle, the delete rule cannot be CASCADE. Otherwise, a finite loop could be created on a row deletion which would delete all rows in both tables.
- If there is data in the table, creating foreign keys puts the tablespace in a CHECK PENDING mode. (A table in a non-segmented tablespace is considered populated if it has ever contained records, even if all were deleted.) This means that the tablespace is locked until the check utility is run. The check utility must be run to unlock the tablespace.

Drop

The Drop Referential Integrity option permits dropping of referential rules. All DB2 ALTER TABLE statements necessary to drop primary and foreign keys are supported.

Primary Keys

If a primary key definition is dropped, the UNIQUE INDEX remains active, and all DEPENDENT RULES are also dropped.

Foreign Keys

When dropping a foreign key, remember that the user must have *both* the normal authorities on the dependent table and the ALTER privilege on the parent table. Additionally, when dropping a foreign key, no plans are invalidated.

Drop Referential Integrity Screen

The Drop Referential Integrity option enables a primary or foreign key to be dropped from a table. To drop a key, enter **D** in the option field and **RI** in the object field. Enter the name (or portion of a name) of the primary key table in the Item field or the name of the primary key table creator in the Creator field to narrow the displayed selection list. Use the Extended Query Facility by entering **Y** in the Where field.

After the information has been entered, press Enter.

After F has been entered next to the foreign keys to be dropped, and P has been entered next to the primary keys to be dropped, press Enter. A Drop confirmation screen will appear. Accept, edit, or reject the drop.

Commands

The Header and Compare commands can make working with aliases easier. Following is a brief description of how to use these commands:

COMPARE

Compare is useful when you are templating or altering an alias. The C (compare) command can be entered in the command line to see a summary of all changes made to the alias. The Compare screen shows the old and new versions of the alias. Use the RESET command to reset the attributes to their original status.

HEADER

The H (header) command toggles the header on and off. Enter **H** in the command line to toggle the header.

Column Manipulation Commands

The display of the Drop Referential Rules screen can be customized through the use of column manipulation commands. FREEZE (UNFREEZE, MELT), and EXCLUDE columns to make selecting keys easier.

Note: For more information about primary commands, see the online help.

FREEZE

The FREEZE command fixes a specified column in place on the display during scrolling. To freeze a column, enter **FR** followed by the name of the column. Freeze multiple columns by entering multiple FR (freeze) commands.

UNFREEZE

The UNFREEZE command, like the Freeze command, works only on the column named. To unfreeze multiple columns, enter multiple Unfreeze commands.

MELT

To unfreeze all columns, use the MELT command. The MELT command unfreezes all frozen columns (except the permanently frozen columns).

EXCLUDE

The EXCLUDE command excludes the specified column from display. To exclude a column enter **X** (exclude) followed by the column name. To show excluded columns, enter the **SHOW** command.

DDL Execution

When F3 (End) is pressed from the Create, Alter, Template or Drop screen, a confirmation screen appears before the DDL is executed. Accept, edit, or reject the DDL to be used to complete the request.

Note: For more information, see the “Using CA RC/Update” chapter.

If your changes are complex, require dropping and recreating objects (and their dependencies), or the current CA RC/Update Operation Mode is set to A (RC/Alter), you can perform a complete analysis of your changes that includes rebuilding the DDL and optionally including utility statements and extended DDL generation and analysis options.

Chapter 21: View

This section contains the following topics:

Features for Maintaining Views

The View object options fully support the DB2 CREATE and DROP VIEW statements, and changes to a view.

View option functionality is as follows:

- Table and Column Selection Help. Instead of specifying specific table and columns names, you can select a table from a table selection list. Enter selection criteria to make the selection list more specific. Once a table has been selected, all columns for that table are displayed in a separate scrollable column area. Select columns from this list for the view.
- Column Area. All columns from every table referenced are displayed in a separate scrollable area. Select columns to include in the view from this list. After selecting the columns, appropriate SQL is added to include the column in the view.
- SELECT Statement Boilerplate. A boilerplate SELECT statement is included to assist in writing the view SQL. This boilerplate includes placeholders for column names. Select the columns and they are included appropriately in the select statement.
- Interactive SELECT statements. As SELECT statements are entered, the results are returned in a table selection panel or the column area. Select the items to include, and the SQL that will be used to create the view is updated.
- Alter Option. Although DB2 does not provide an ALTER VIEW statement, you can change any characteristic of a view with this option. The changes are made by dropping and recreating the view.

Accessing View Functions

View functions can be accessed by requesting to create, alter, template, or drop a view:

- To create a view, enter **C** in the Option field and **V** in the Object field of the main header, and press Enter.
- To alter, template, or drop a view, enter **A**, **T**, or **D** in the Option field and **V** in the Object field of the main header and press Enter.

A view name can also be entered in the Item Name field or a user ID in the Creator field to get a more specific selection list. From the selection list, select the views to alter, template, or drop. The appropriate view screen appears.

Selection Panels

When altering, dropping, or creating (by template) a view, a selection panel is presented to select the view if not already specified. Make the view selection list more specific by specifying selection criteria for the Item (view) Name and Creator ID prompts in the header.

Use the EQF facility to further specify the views displayed, and use scrolling commands to scroll through information.

Note: For more information about EQF and scrolling commands, see the *CA Database Management Solutions for DB2 for z/OS General Facilities Reference Guide*.

The View Selection screen displays all the fields available in the SYSIBM.SYSTABLES table. This information is available to make selection of the views easier.

To choose items from the listing, type **S** next to the items to select. Press Enter to register the selection.

Note: For information about the fields, press F1 (Help).

Text Command

The **T** (text) command displays the text of the view while the selection panel appears. While viewing a View selection panel, enter **T** (text) in the command line. The text of the first view (and any other views that fit) appears.

Note: For information about the fields, press F1 (Help).

Scroll down to display more view text. For faster display, move the view of the text to display to the top of the selection list by moving the cursor to that view, and scrolling down with scroll type DATA before entering the **T** (text) command. The **T** (text) command works as a toggle; enter **T** to turn text off.

Editing or Browsing View Data

Edit or browse data in views by using RC/Edit.

Note: For more information, see the “RC/Edit” and “RC/Browse” chapters.

View Commands

The Header (see page 286) and Compare (see page 449) commands can make working with views easier.

Note: For more information about primary commands, see the online help.

Compare

Compare is useful when you are templating or altering a view. The C (compare) command can be entered in the command line to see a summary of all changes made to the view. The Compare screen shows the old and new versions of the view. Use the RESET command to reset changed attributes to their original status.

Header

The header is the first part of the analysis output file and is included by default. To exclude the header, N must be specified in the Header prompt of the Analysis Options screen. The analysis header is especially helpful when reviewing output generated several days earlier. The analysis output header indicates the strategy used to generate the output, the strategy definition, and the analysis options.

The information in the header is also displayed during execution. Batch Processor comments appear as part of the execution output; however, they are truncated to 72 characters. Part of a sample header for an object alteration analysis is shown in the following example. Notice that all the lines are Batch Processor (PBP) comments. PBP comments must have two hyphens (--) as the first two characters of the line:

```
--STRATEGY INFORMATION:
--STRATEGY ==> RCUPDATE  DESCRIPTION  ===> RC/UPDATE ALTERATION
--CREATOR  ==> USER1     SHARE OPTION ===> N (U,Y,N,X,L)  SRC SSID ===> D81A
--
--
--ALTERED OBJECTS:
--  OBJECT TYPE         NAME              CREATOR
--  TABLE               EDIT_DEMO         USER02
--
--ANALYSIS OPTIONS:
```

The header is broken down into three parts: Strategy Information, Altered Objects, and Analysis Options.

Create

The Create View option permits the creation of a view through a series of easy to use screens. These screens include a Table selection panel and Column Selection help. A typical Create View session starts with a strategy screen, moves to the Create View screen, and can include a Table selection panel.

View Create Screen

The View Create screen provides an easy way to supply the information necessary for view creation.

CAPS

The status of the CAPS feature.

Note: For more information, see the *CA Database Management Solutions for DB2 for z/OS General Facilities Reference Guide*.

NULLS

The status of the NULLS feature.

Note: For more information, see the *CA Database Management Solutions for DB2 for z/OS General Facilities Reference Guide*.

View Name and Select Statement

The default SELECT statement used to create the view.

Check Option

The Check option enables you to add the DB2 WITH CHECK OPTION statement to the view syntax. This option indicates all inserts and updates against the view are to be checked against the view definition (the search condition of the WHERE clause) and rejected if the inserted or updated row does not conform to that definition. If the view does not have a WHERE clause this is ignored.

CMD (Command)

Enter ISPF line commands here to insert or update the SELECT statement.

Note: For more information about line commands, see the *CA Database Management Solutions for DB2 for z/OS General Facilities Reference Guide*.

Enter the View Name and Creator ID in the appropriate fields. If this information was supplied in the Item Name and Creator fields of the header, the information is carried to this screen.

SELECT Statement

The syntax for creating a view requires a SELECT statement that begins AS SELECT. The boilerplate statement provided with the Create View screen is

```
( @ ) AS SELECT @ FROM
```

The at sign (@) serves as a placeholder for column names selected from the list of columns in the separate column scrollable area.

The SELECT statement provided as a boilerplate for all SELECT statements includes two at signs. The first (before the SELECT) will hold the place of the name assigned to the columns in the view. The second (after the SELECT) holds the names of the columns within the table. Include @ signs anywhere in the SELECT statement to hold the place of column names. If there is no @ sign and columns are selected, the columns will be placed in the SELECT statement before the FROM clause of the first SELECT.

You can access a table selection list by entering table selection criteria with PARSE on. PARSE on is the default. The asterisk (*) is not valid.

For example, to get a listing of all tables created by USER1, enter this SELECT statement:

```
( @ ) AS SELECT @ FROM USER1.%
```

A table selection panel will appear. Tables created in batch mode in the current session are listed at the top of the Table selection panel and CREATE appears in the Database column. They are selected like any other table. Select the table from which columns should be selected. The table name enters the SQL statement and the column names appear in the column scrollable area. (When tables are selected, the column listing at the bottom also adjusts.)

Column Area

Columns are displayed in a separate scrollable area at the bottom of the screen. The Column Area displays the column names for any tables that appear in a FROM clause in the SELECT statement. The Column Area is scrolled using the ISPF left and right commands (F10 and F11).

Note: For information about the fields, press F1 (Help).

Scrolling Data

Use F7 and F8 to scroll the data in the SQL area. Use the F10 and F11 to scroll the data in the column scrollable area.

Selecting Columns

Select columns from the column scrollable area by entering **S** next to each column to select. The column names will be placed in the SELECT statement wherever a place holder (@) appears, in accordance with the rules outlined in the previous section.

If a column is added to the SELECT portion of the statement using the @ place holder, the FROM TABLE portion of the statement must still be updated.

Note: If a column is selected without using a place holder, the column name will be added to the column list with the SELECT column and FROM table portion of the statement updated automatically.

Using Correlation Letters

If correlation letters were included in the SELECT statements, the correlation letter can be used as an abbreviation for the table name in subsequent SQL. CA RC/Update will also refer to table names using correlation letters, if specified.

Using Column Numbers

Every column displayed in the Column Area is assigned a number. Use this number in the SELECT statement as a column abbreviation. Use the column number preceded by a colon. When Enter is pressed, the column abbreviation is expanded to the full column name.

ALL Command

The ALL command can be used to SELECT all the columns of a table.

Confirming the Creation

The Creation Confirmation screen enables users to accept, edit, or reject the DDL to be used to create the object.

Template

The template option allows creation of a new view using an existing view as a template. A template session is actually a create session with the additional step of selecting a view as a starting point. The name of the templated view must be changed. The other fields can be changed. The view's dependent objects are not included in the template operation.

The View Template screen displays the values that were used to create the template view. The fields for this screen are similar to those on the View Create screen. Change the fields for the new view according to the procedures outlined in the Create section. When you have made your changes, press F3 (End) to process the template.

Note: Use the ALL command to select all the columns of a table.

A confirmation screen will appear. Accept, edit, or reject the DDL to be used to create the views.

More information:

ALL Command (see page 453)

Alter

The Alter View option permits changes to any characteristic of a view. DB2 does not provide an ALTER VIEW statement, so the view is dropped and re-created. The screen flow for an Alter session is similar to the Template session.

The View Alter screen is similar to the View Template screen. When the screen appears, all columns from every table referenced are displayed in the column scrollable area. Change fields as necessary, according to the procedure outlined in the Create. When finished, press F3 (End) to process the alter. There is no DB2 ALTER VIEW statement. All view alterations are made by dropping and recreating the view.

Note: Use the ALL command to select all the columns of a table.

The Alter Confirmation screen appears. Accept, edit or reject the DDL to be used to alter the view.

ALL Command

The ALL command can be used to SELECT all the columns of a table.

Create Commands

There are many commands to assist with a View Creation session. A brief description is presented here.

Note: For more information about these commands, see the *CA Database Management Solutions for DB2 for z/OS General Facilities Reference Guide*.

CAPS

Works as a toggle, turning CAPS lock on and off.

COLS ON or OFF

Controls the display of a second scrollable area at the bottom of the screen called the Column Area.

FORMAT

Lets you format the view text into a more readable format with indentation.

TEXT

Is intended for fast entry of text using fast typing techniques. When the TE command is entered, the CMD area is removed and automatic cursor skip will occur upon reaching the end of the text line. This permits quick entry of SQL statement text without regard to cursor position. When Enter is pressed, TE is removed and the user is placed back into standard text entry. If words are split between lines, and PARSE is on, the split will be fixed when Enter is pressed.

WORD WRAP

Turns word wrap on and all lines are restructured to fit as much on a line as possible. Enter the WW command to reflow fragmented statement text into a uniform text lines.

WORD SPLIT

Separates the SQL statement text into tokens and places each token on a separate line. After entering the WS command and performing any edits, enter the WW (Word Wrap) command to reflow the tokens into uniform text lines.

NULLS

Works as a toggle, turning on and off the NULLS feature. When NULLS is on, space at the beginning or end of the line is considered null. When NULLS is off, space at the beginning or end of the line is considered blank spaces. By default, NULLS is ON.

PARSE ON or OFF

Turns on and off the parsing feature of the editor. When PARSE is on, the entered SQL statement is parsed looking for special characters.

View Drop Considerations

When a view is dropped, DB2 invalidates application plans dependent on the view and revokes the privileges of users authorized to use it.

Note: For information about how to drop an object with the various drop options (S, SO, R, RO), see the “Using CA RC/Update” chapter.

DDL Execution

When F3 (End) is pressed from the Create, Alter, Template or Drop screen, a confirmation screen appears before the DDL is executed. Accept, edit, or reject the DDL to be used to complete the request.

Note: For more information, see the “Using CA RC/Update” chapter.

If your changes are complex, require dropping and recreating objects (and their dependencies), or the current CA RC/Update Operation Mode is set to A (RC/Alter), you can perform a complete analysis of your changes that includes rebuilding the DDL and optionally including utility statements and extended DDL generation and analysis options.

Chapter 22: Sequence

This section contains the following topics:

Overview of Sequences

A sequence is a user-defined object that generates an ordered arrangement of numeric values, in ascending or descending order according to the user's specification. Using sequences, DB2 users can request and specify a unique value for a given column (presumably a key column).

You can create, template, alter, and drop a sequence.

Create a Sequence

You can create sequence objects.

Follow these steps:

1. Perform the following actions on the CA RC/Update Main Menu:
   - Type **C** in the Option field.
   - Specify **SQ** in the Object field.
   - (Optional) Type values for one or more of the remaining fields to add to the object definition.

   Note: For complete information about the fields on the CA RC/Update Main Menu, see the online help.

   Press Enter.

   The Sequence Create panel appears.

2. Complete the fields, and then press Enter.

   The values are applied as your new sequence's attributes.

   Press the END key.

   The RC/Update Creation Confirmation panel appears with the generated DDL for creating your object.

3. Perform *one* of the following actions:

   - Press Enter to execute the DDL online through the Batch Processor.
   - Enter **E** in the command line to edit the generated DDL.
   - Press the END key to return to the create panel.

   Note: Depending on the CA RC/Update Profile settings for your user ID, you might also have the option of rebuilding the DDL with RC/Alter so you can specify utilities and extended DDL generation and analysis options.

   Your session continues, depending on your choice.

Template a Sequence

You can create a sequence by using an existing sequence as a template. This process is known as *templating* a sequence.

Follow these steps:

1. Perform the following actions on the CA RC/Update Main Menu:

   - Type **T** in the Option field of the main header.
   - Type **SQ** in the Object field of the main header.
   - Type a name in the Item Name field to specify the sequence you want to template.

     Note: If you do not specify a name, a selection panel appears when you press Enter.
   - (Optional) Type values for one or more of the remaining header fields.

   Note: For complete information about all fields, see the online help.

   Press Enter.

   The Sequence Template panel appears.

2. Complete header fields on the Sequence Template panel as follows:

Sequence

Specifies a name for your sequence.

Schema (first occurrence)

Specifies the name of a sequence creator.

Schema (second occurrence)

Specifies the high-level qualifier of the data type for the sequence. The value in this field, along with the value in the AS Type field, specifies the data type to be used for the sequence. When the value for AS Type is ambiguous or invalid, the value for this Schema field is also used as part of the criteria by which the column type selection list is built. This field is scrollable and expandable.

AS Type

Specifies the sequence's data type, which can be any exact numeric data type (SMALLINT, BIGINT, INTEGER, or DECIMAL) or a user-defined distinct type for which the source type is an exact numeric data type with a scale of 0. This field is scrollable and expandable. Selection criteria are also accepted.

Comment

Specifies whether comments are defined for the sequence. You can also enter comments.

N

Indicates that no comments exist for the sequence. This is the default.

Note: If Y is displayed in this field, and you type over the Y with N, the existing comment is deleted.

Y

Indicates that a comment is defined for the sequence.

U

Indicates that you want to view, update, or delete the sequence's comment.

Precision

Specifies the precision for the sequence. Precision depends on the data type of the sequence. You can enter a specific value for Precision only if the specified data type is one of the following:

- A user-defined type (from which a defined precision can be picked up)

  Note: When the sequence's data type is SMALLINT, INTEGER, or a distinct type that is sourced on SMALLINT, BIGINT, INTEGER or DECIMAL, you cannot edit the Precision field. Instead, the field displays a precision that is inherent to (or defined for) the specified type.

- DECIMAL

If the data type is not user-defined or DECIMAL, Precision is set to 10 (for an INTEGER type) or 5 (for a SMALLINT type).

Press Enter.

The values are applied as your templated sequence's attributes.

3. Specify any changes to information for the Sequence Dynamics, and then press the END key.

   The RC/Update Creation Confirmation panel appears for confirming that you want to execute DDL to create your sequence.

4. Perform *one* of the following actions:

   - Press Enter to execute the DDL online through the Batch Processor.
   - Enter **E** in the command line to edit the generated DDL.
   - Press the END key to return to the Sequence Template panel.

   Note: Depending on the CA RC/Update Profile settings for your user ID, you might also have the option of rebuilding the DDL with RC/Alter so you can specify utilities and extended DDL generation and analysis options.

   Your session continues, depending on your choice.

Alter a Sequence

You can alter a sequence. All changes that need to be propagated to dependent objects are incorporated into alterations.

Follow these steps:

1. Perform the following actions on the CA RC/Update Main Menu:
   - Type **A** in the Option field of the main header.
   - Type **SQ** in the Object field of the main header.
   - Specify a sequence name in the Item Name field.
   - Specify a creator name in the Creator field.

   Note: If you do not specify a sequence name or creator name, a selection list appears, from which you can choose.

   Press Enter.

   The Sequence Alter panel appears.
2. Change any Sequence Dynamics definition fields as needed, and then press the END key.

   The Alter Confirmation panel or Alteration Analysis panel (for RC/Alter) appears, depending on whether changes can be made without dropping and re-creating the object.
3. Continue the process as follows, depending on the panel presented:
   - Respond on the Alter Confirmation panel by performing *one* of the following actions:
     - Press Enter to execute the DDL online through the Batch Processor.
     - Enter **E** in the command line to edit the generated DDL.
     - Press the END key to return to the alter panel.
   - Respond on the Alteration Analysis panel by typing **O** or **B** for Execution Mode, completing other fields as needed, then pressing Enter.

     Note: Based on your selections on the Alteration Analysis panel, you can invoke the Analysis Options panel, Recovery Options panel, or Batch JCL Specification panel (if batch mode is specified).

   Your session continues, depending on your choice.

Assign a Data Type to a Sequence

Using the Type command, you can select and assign a data type for a sequence. You can choose a built-in type or user-defined distinct type (UDT).

The initial list that is presented is built using the current Type Schema that is specified for the sequence, with all remaining selection criteria (Item Name, Source Schema, and Source Type) set to asterisk (*).

Follow these steps:

1. Enter **TYPE** in the command line of a sequence create, alter, or template panel.

   The Sequence Data Type Selection panel appears.

2. (Optional) Enter additional selection criteria for any of the following fields to customize the listed information:

   - Item Name
   - Schema
   - Src Schema
   - Src Type

   The listed information changes according to your specifications.

3. Enter **S** next to the name of the data type that you want to assign.

   The sequence create, alter, or template panel appears, with your new data type assigned to the sequence.

How to Prevent Duplicate Sequences After a Drop-and-Recreate

If an alteration to a sequence causes the object to be dropped and recreated, the default start value for the sequence (START WITH) is set to its default. The recreated sequence also does not use the current value for MAXASSIGNEDVAL (which specifies the last possible assigned value for the sequence). These improper settings cause duplicate sequence objects.

To resolve this issue, do the following:

1. Install the RMRSQPE stored procedure.

   RMRSQPE fetches the MAXASSIGNEDVAL or RESTARTWITH values that exist for the sequence *before* it is dropped. After the sequence is recreated, the procedure restarts numbering for the sequence object based on the previous MAXASSIGNEDVAL value

2. Define model utilities to call the procedure.

3. Perform the alteration again.

Note: For information about installing RMRSQPE and defining the necessary model utilities, see the *Implementation Guide*.

Example: Invoke a Stored Procedure to Reset the Start Value for a Recreated Sequence

This user scenario describes how a database administrator can use the RMRSQPE stored procedure to restart numbering for an altered sequence that has been dropped and recreated. By using RMRSQPE, the sequence object resumes with the proper value, and you avoid encountering duplicate sequences. The scenario is as follows:

1. You create an alteration strategy to alter sequence object SEQ1 by changing the schema name. This alteration requires a drop/create for the object, at which point START WITH (the start value for the sequence) is set to its default. This leads to duplicate sequences.

2. To circumvent the issue, you install RMRSQPE on each subsystem (as needed) and create an CA RC/Migrator model with additional steps that call RMRSQPE.

3. Now, when the sequence object is recreated after an alteration, RMRSQPE restarts numbering for the sequence object from the previous MAXASSIGNEDVAL. Therefore, the sequence object resumes properly.

Chapter 23: Stored Procedures

This section contains the following topics:

Overview of Stored Procedures

A *stored procedure* is a routine that can be called to perform operations. A stored procedure can be either of the following:

SQL procedure

Contains only SQL statements and can be a *native* SQL procedure or *external* SQL procedure.

External procedure

Contains host language statements (and might contain SQL statements).

With stored procedures, common code can be called from several programs. Host languages can call procedures that exist on the local system. Additionally, SQL can call a procedure on a remote system, which means you can use SQL procedures to enhance performance of distributed applications.

Language and Fenced Options

When you define a stored procedure, the following fields on the create, alter, or template panel determine what options are made available for completing the definition:

- Language (which determines the language in which the procedure will be written)
- Fenced (which determines whether the procedure runs in an external address space)

The header portion of the panel is dynamically built and allows modification only to attributes that are applicable, based on the settings of Language and Fenced. As a result, you do not have to be concerned with attributes that are not applicable to the type of procedure with which you are working. When you change the setting of Language or Fenced, these other available attributes change accordingly.

Note: For complete information about Language, Fenced, and all other fields, see the online help.

Parameter Declaration List

The Parameter Declaration List is a scrollable region on the procedure create, alter, and template panels. You can use this list to manage parameter information, including the following:

- Number of parameters for your stored procedure
- Data type of each parameter
- Name of each parameter
- Inclusion of temporary transition tables

More information:

SQL Body and Parameter Declaration List Display (see page 478)
SQL Body (see page 467)

Convert Distinct Types to Their Respective Source Types

For distinct types specified by the DATA TYPE and SCHEMA fields in all parameter declarations, you can automatically convert the types to the built-in type upon which each distinct type is sourced.

To convert distinct types to their respective source types, enter **CD** in the command line of any procedure create, alter, or template panel.

Note: This command is valid *only* when the Parameter Declaration List is displayed. Additionally, if no distinct types or parameters are defined for the procedure, the command has no effect on the procedure.

SQL Body

The SQL body specifies the statements that define the body of an SQL procedure. An SQL body is required to generate valid DDL for an SQL procedure.

You can use standard ISPF editor commands to copy, replicate, delete, and move lines to manage the SQL procedure's SQL text.

Note: If the Parameter Declaration List is showing (or enabled) instead of the SQL body, you need to issue the SQL primary command to display the SQL body. If the SQL body is not visible because there is not enough room, you can issue the HEADER primary command (to turn off the header content) or issue the COMPARE primary command (to toggle the compare display OFF if it is ON).

More information:

SQL Body and Parameter Declaration List Display (see page 478)
Parameter Declaration List (see page 466)

Format the SQL Text for Better Readability

You can use the FO command to format the current editable or displayed SQL text into a more readable format (with indentation).

Note: Entering this command causes the requested formatting to be applied only once to the current text. Additionally, the current PARSE setting has no influence on the behavior of this command.

To format the SQL text for better readability, enter **FO** in the command line.

The product indents and formats the text in the SQL body into a more readable format.

Limitations when Implementing Stored Procedures

Certain limitations apply when regarding implementing stored procedures.

Limitations regarding RC/Alter are as follows:

- RC/Update Mode A is not supported for stored procedures.
- A response of A to the DDL Execution Confirmation window is not supported for stored procedures.
- The product does not support some changes that require an object drop/create due to an RC/Alter analysis being required in order to generate the necessary utility, drop, and create statements. This includes some changes where object interdependencies with other objects require an RC/Alter analysis to determine whether changes to other objects are needed. When a drop/create can be performed without an RC/Alter analysis, a DROP will be generated, followed by a CREATE; otherwise, no DDL is generated.

Stored procedure create, alter, and template limitations are as follows:

- Stored procedures created with a CREATE or ALTER statement that is longer than 32767 bytes are not eligible for alteration or templating in CA RC/Update.
- The SQL Body is not extracted from a CREATE or ALTER procedure statement for an alter or template.
- Stored procedures cannot be created with an SQL Body that exceeds 32767 bytes.
- Altering Language for an existing stored procedure will generate a drop/create.
- Altering the TT option from NO to YES in the Parameter Declaration List is not supported and will cause bad DDL to be generated.
- When creating or templating stored procedures, specifying YES for the TT option in the Parameter Declaration List is not supported and will cause bad DDL to be generated.

Drop limitations are as follows:

- Drop Impact reporting is not supported.
- You cannot drop stored procedures that have dependent objects.
- Drop with Recovery is not supported for stored procedures, meaning that dropped stored procedures cannot be automatically recovered.

The following option limitations exist:

- Comment is not supported and if set to Y will not generate COMMENT ON PROCEDURE.
- Add Ver is not supported.

- Active is not supported.
- Decimal does not reflect the value stored in the DB2 catalog.
- FOR UPDATE CLAUSE does not reflect the value stored in the DB2 catalog.
- Date Format does not reflect the value stored in the DB2 catalog.
- Time Format does not reflect the value stored in the DB2 catalog.
- TT in the Parameter Declarations list cannot be altered from NO to YES during an alteration; otherwise, bad DDL will be generated.

Miscellaneous support limitations are as follows:

- The Table Like clause is not currently supported.
- The TT line command (Table Like) is disabled in the Parameter Declaration List.
- ADD, REPLACE, and REGENERATE version is not currently supported for a native SQL stored procedure.
- You cannot drop/create a native SQL stored procedure having multiple versions.
- Altering any parameter declaration attribute on an existing native SQL stored procedure causes a drop/create.

Create a Stored Procedure

You can create a stored procedure.

Follow these steps:

1. Complete fields as follows on the CA RC/Update Main Menu:
   - Type **C** in the Option field.
   - Type **PR** in the Object field.
   - Type a name in the Item Name field.
   - (Optional) Type values for one or more of the remaining fields to add to the object definition.

   Note: For complete information about the fields on the CA RC/Update Main Menu, see the online help.

   Press Enter.

   The Procedure Create panel appears.

2. Complete the following fields:

Language

Specifies the language interface convention to which the procedure body is written.

Note: All programs must be designed to run in the server's environment. Additionally, Assembler, C, COBOL, and PL/I must be designed to run in IBM's Language Environment.

ASSEMBLE

Writes the stored procedure in Assembler. Unless overridden by object definition defaults, this is the default.

C

Writes the stored procedure in C or C++.

COBOL

Writes the stored procedure in COBOL (including the OO-COBOL language extensions).

JAVA

Writes the stored procedure in Java byte code and executes the procedure in the Java Virtual Machine.

PLI

Writes the stored procedure in PL/I.

REXX

Writes the stored procedure in REXX. Specifying REXX sets Program Type to MAIN and restricts the allowable values for Parm Style to GENERAL or NULLS.

SQL

Writes the stored procedure in DB2 SQL procedural language. Specifying SQL sets DBINFO to NO and hides the Parm Style field.

Fenced

Specifies whether the procedure runs in an external address space.

Note: Unless Language is SQL, this field unconditionally defaults to YES and *cannot* receive keyed input.

YES

Specifies that the procedure runs in an external address space (and when Language is SQL, specifies that the SQL procedure program is an MVS load module with an external name). Unless overridden by object definition defaults and Language is SQL, this is the default.

For non-SQL procedures (when Language is other than SQL), the procedure must always run in an external address space, in which case this option is unconditionally set to YES (regardless of its object definition defaults setting).

NO

Specifies that the procedure does not run in an external address space. NO is allowed only when Language is SQL, thus defining your SQL procedure as native; otherwise, the SQL procedure is external.

You may specify NO even when External Name is not blanks, which removes External Name from the panel and treats it as implicitly blank. Changing Fenced back to YES restores External Name to the panel, showing its previous value.

3. Complete the procedure definition as follows:
   - Complete the remaining fields on the panel, depending on your Language and Fenced selections.

     Note: For complete information about all fields, see the online help.
   - Complete any information in the Parameter Declaration List (see page 466) or SQL body (see page 467) area.

   Press the END key.

   A confirmation panel appears.
4. Press Enter, and then enter **Y** in the execution confirmation pop-up to begin the activity.

   The product processes the request.

More information:

Language and Fenced Options (see page 466)
Parameter Declaration List (see page 466)
SQL Body (see page 467)

Alter a Stored Procedure

You can alter a stored procedure.

Follow these steps:

1. Complete fields as follows on the CA RC/Update Main Menu:
   - Type **A** in the Option field.
   - Type **PR** in the Object field.
   - Type a name in the Item Name field.
   - (Optional) Type values for one or more of the remaining fields to add to the object definition.

   Note: For complete information about the fields on the CA RC/Update Main Menu, see the online help.

   Press Enter.

   The Procedure Alter panel appears.

2. Complete the following fields:

   Language

   Specifies the language interface convention to which the procedure body is written.

   Note: All programs must be designed to run in the server's environment. Additionally, Assembler, C, COBOL, and PL/I must be designed to run in IBM's Language Environment.

   ASSEMBLE

   Writes the stored procedure in Assembler. Unless overridden by object definition defaults, this is the default.

   C

   Writes the stored procedure in C or C++.

   COBOL

   Writes the stored procedure in COBOL (including the OO-COBOL language extensions).

   JAVA

   Writes the stored procedure in Java byte code and executes the procedure in the Java Virtual Machine.

PLI

Writes the stored procedure in PL/I.

REXX

Writes the stored procedure in REXX. Specifying REXX sets Program Type to MAIN and restricts the allowable values for Parm Style to GENERAL or NULLS.

SQL

Writes the stored procedure in DB2 SQL procedural language. Specifying SQL sets DBINFO to NO and hides the Parm Style field.

Fenced

Specifies whether the procedure runs in an external address space.

Note: Unless Language is SQL, this field unconditionally defaults to YES and *cannot* receive keyed input.

YES

Specifies that the procedure runs in an external address space (and when Language is SQL, specifies that the SQL procedure program is an MVS load module with an external name). Unless overridden by object definition defaults and Language is SQL, this is the default.

For non-SQL procedures (when Language is other than SQL), the procedure must always run in an external address space, in which case this option is unconditionally set to YES (regardless of its object definition defaults setting).

NO

Specifies that the procedure does not run in an external address space. NO is allowed only when Language is SQL, thus defining your SQL procedure as native; otherwise, the SQL procedure is external.

You may specify NO even when External Name is not blanks, which removes External Name from the panel and treats it as implicitly blank. Changing Fenced back to YES restores External Name to the panel, showing its previous value.

3. Complete the procedure definition as follows:

   - Complete the remaining fields on the panel, depending on your Language and Fenced selections.

     Note: For complete information about all fields, see the online help.

   - Complete any information in the Parameter Declaration List (see page 466) or SQL body (see page 467) area.

   Press the END key.

   A confirmation panel appears.

4. Press Enter, and then enter **Y** in the execution confirmation pop-up to begin the activity.

   The product processes the request.

More information:

Language and Fenced Options (see page 466)
Parameter Declaration List (see page 466)
SQL Body (see page 467)

Template a Stored Procedure

You can template a stored procedure.

Follow these steps:

1. Complete fields as follows on the CA RC/Update Main Menu:

   - Type **T** in the Option field.
   - Type **PR** in the Object field.
   - Type a name in the Item Name field.
   - (Optional) Type values for one or more of the remaining fields to add to the object definition.

   Note: For complete information about the fields on the CA RC/Update Main Menu, see the online help.

   Press Enter.

   The Procedure Template panel appears.

2. Complete the following fields:

Language

Specifies the language interface convention to which the procedure body is written.

Note: All programs must be designed to run in the server's environment. Additionally, Assembler, C, COBOL, and PL/I must be designed to run in IBM's Language Environment.

ASSEMBLE

Writes the stored procedure in Assembler. Unless overridden by object definition defaults, this is the default.

C

Writes the stored procedure in C or C++.

COBOL

Writes the stored procedure in COBOL (including the OO-COBOL language extensions).

JAVA

Writes the stored procedure in Java byte code and executes the procedure in the Java Virtual Machine.

PLI

Writes the stored procedure in PL/I.

REXX

Writes the stored procedure in REXX. Specifying REXX sets Program Type to MAIN and restricts the allowable values for Parm Style to GENERAL or NULLS.

SQL

Writes the stored procedure in DB2 SQL procedural language. Specifying SQL sets DBINFO to NO and hides the Parm Style field.

Fenced

Specifies whether the procedure runs in an external address space.

Note: Unless Language is SQL, this field unconditionally defaults to YES and *cannot* receive keyed input.

YES

Specifies that the procedure runs in an external address space (and when Language is SQL, specifies that the SQL procedure program is an MVS load module with an external name). Unless overridden by object definition defaults and Language is SQL, this is the default.

For non-SQL procedures (when Language is other than SQL), the procedure must always run in an external address space, in which case this option is unconditionally set to YES (regardless of its object definition defaults setting).

NO

Specifies that the procedure does not run in an external address space. NO is allowed only when Language is SQL, thus defining your SQL procedure as native; otherwise, the SQL procedure is external.

You may specify NO even when External Name is not blanks, which removes External Name from the panel and treats it as implicitly blank. Changing Fenced back to YES restores External Name to the panel, showing its previous value.

3. Complete the procedure definition as follows:
   - Complete the remaining fields on the panel, depending on your Language and Fenced selections.

     Note: For complete information about all fields, see the online help.
   - Complete any information in the Parameter Declaration List (see page 466) or SQL body (see page 467) area.

   Press the END key.

   A confirmation panel appears.
4. Press Enter, and then enter **Y** in the execution confirmation pop-up to begin the activity.

   The product processes the request.

More information:

Language and Fenced Options (see page 466)
Parameter Declaration List (see page 466)
SQL Body (see page 467)

Drop a Stored Procedure

You can drop a stored procedure.

Note: When you perform a drop of a native SQL stored procedure, you can drop all versions, or you can view and drop specific versions.

Follow these steps:

1. Complete fields as follows on the CA RC/Update Main Menu:
   - Type **D** in the Option field.
   - Type **PR** in the Object field.
   - (Optional) Type a name in the Item Name field.
   - (Optional) Type values for one or more of the remaining fields.

   Note: For complete information about the fields on the CA RC/Update Main Menu, see the online help. You can also generate drops from the CA RC/Update Drop panel or any create, alter, or template panel.

   The CA RC/Update Drop Procedure Selection panel appears, from which you can select one or more objects to drop.
2. (Optional) Perform one of the following to display versions for native SQL procedures in your list:
   - Enter **VERSIONS** in the command line to display versions of any native SQL procedures.
   - Enter **V** next to a specific native SQL procedure to expand its list of versions.

   Additional versions are listed separately.
3. Enter one of the following commands next to each procedure that you want to delete:

   S

   Specifies to perform a standard drop without options.

   SO

   Specifies to perform a standard drop with options. The product displays a Drop Options panel for each object selected with SO.

   R

   Specifies to perform a recoverable drop. This type of drop saves recovery information so that the object can be recovered and restored to its original state at a later time.

   RO

   Specifies to perform a recoverable drop with options. The product displays a Drop Recovery Options panel for each object selected with RO.

If you have specific versions under a native SQL stored procedure, you can then select the specific versions that you want to drop.

If you requested deletion for a native SQL Stored procedure with multiple versions, the Drop All Versions Confirmation window appears.

4. Enter a value in the Drop All Versions Confirmation window to determine whether to delete all versions:

   Y

   Drops all versions of the selected native SQL stored procedure. A DROP PROCEDURE statement will be generated for the drop.

   N

   Does not drop any versions for the selected item, bypassing the selection in its entirety. This is the default.

   The CA RC/Update Drop Impact List panel appears, listing the objects that will be dropped, based on your request.

SQL Body and Parameter Declaration List Display

When you work with stored procedures in CA RC/Update, only one of the following can be displayed on the panel at any given time:

- Parameter Declaration List, in which you manage the number of parameters for your stored procedure, the data type of each parameter, and the name of each parameter
- SQL body, in which you use standard ISPF editor commands to copy, replicate, delete, and move lines to manage an SQL procedure's SQL text

You cannot apply commands to an item that is not displayed. For example, if the Parameter Declaration List is active, you cannot enter a command that is applicable only to the SQL body.

You can issue the SQL or PARMS primary commands, respectively, to toggle the display between the Parameter Declaration List and SQL body.

Note: The current toggled setting can be saved, using the SAVESETTINGS (SS) command, so that it will be retained across multiple sessions and used as the default setting each time you start a new create/alter session for an SQL procedure.

Follow these steps:

1. Verify that the SQL body or Parameter Declaration List appears at the bottom of the panel.

   Note: If there is insufficient room for either section to be displayed, enter **HEADER** in the command line to turn off the header, at which point the Parameter Declaration List or SQL body will appear by itself.

   You are ready to make a replacement as needed on the panel.

2. Enter **SQL** or **PARMS** in the command line to toggle between the display of the SQL body and Parameter Declaration List.

   Toggling occurs as requested.

   Note: If the SQL body or Parameter Declaration List is not visible because there is not enough room on the panel, enter **HEADER** in the command line to toggle the header OFF.

Chapter 24: Synonym

This section contains the following topics:

Overview of Synonyms

In DB2, a synonym is an alternate name for a table or view. Synonyms are created by users who want to refer to tables by names that are easier to remember. These alternate names can also be used in applications to reference tables without tying the source code to the physical object.

Features for Maintaining Synonyms

You can use the synonym options to create, template, alter, and drop synonyms.

The Synonym option includes the following features.

- Table Selection Help. Instead of entering a specific table name, selection criteria can be entered to receive a list of tables that meeting the criteria.
- Create Synonyms for Multiple Users. Create synonyms for multiple users by listing the user's ID(s). SET CURRENT SQLID statements are generated for each user ID specified.

Accessing Synonym Functions

Synonym functions can be accessed by requesting to create, alter, template, or drop a synonym. To create a synonym, enter C in the Option field and S in the Object field of the main header and press Enter.

To alter, template, or drop a synonym, enter **A**, **T**, or **D** in the Option field and S in the Object field of the main header and press Enter. A synonym name can also be entered in the Item Name field or a user ID in the Creator field to get a more specific selection list. From the selection list, select the synonym(s) to alter, template, or drop. The appropriate Synonym screen appears.

Synonym Selection Panels

When altering, dropping, or creating (by template) a synonym, a selection panel is presented for selecting the synonyms. Make the synonym selection list more specific by specifying selection criteria for the Item (synonym) Name /or Creator ID prompts in the header.

Note: For more information about selection criteria, see the “Using CA RC/Update” chapter.

Use the EQF facility to further specify the synonyms displayed., and use scrolling commands to scroll through information.

Note: For more information about EQF and scrolling commands, see the *CA Database Management Solutions for DB2 for z/OS General Facilities Reference Guide*.

The Synonym Selection panel displays all the fields available in the SYSIBM.SYSSYNONYMS table. This information is available to make selection of the synonyms easier.

To choose items from the listing, type **S** next to the items to select. Press Enter to register the selection.

Note: For information about the fields, press F1 (Help).

Compare Command

Compare is useful when templating or altering a synonym. The C (COMPARE) command can be entered in the command line to see a summary of all changes made to the synonym. When working with a migration or alteration strategy, the Compare screen shows the old and new versions of the synonym. Use the RESET command to reset changed attributes to their original status.

When working with a compare strategy, the Compare screen shows the source, target, and current versions of the synonym. Use the SOURCE and TARGET commands to reset changed attributes to either the source or target values.

Create Synonym Option

The Create Synonym option enables the user to create a synonym for a table through a series of easy to use screens. The user does not have to remember the database and table name; a table selection list is provided. After selecting the table, the user assigns a synonym name and can specify other user IDs for synonym creation. A User ID selection panel is available to assists in the process.

Specify the IDs for which synonyms should be created on the Create screen and the necessary SET CURRENT SQLID statements are generated. To create a synonym for another user, that user's ID must be one of your secondary IDs.

Table Selection Panel

To create a synonym, first indicate the table(s) for which synonyms should be created. A Table selection panel will appear before the Synonym Create screen is presented. (Tables created as part of the current strategy appear at the top of the selection list.) To identify them as newly created tables, CA RC/Migrator displays *CREATE* in the Database field. Select tables by typing **S** next to the names and pressing Enter. Each table selected will be processed in turn, in the order of their appearance on the selection list.

Create Screen

The Alias Create screen provides an easy way to supply the information necessary for the alias creation.

Note: For information about the fields, press F1 (Help).

After information has been entered for all fields, press F3 (End) to process the alias creation. The Creation Confirmation screen appears.

Specifying Other User IDs

To specify other user IDs for synonym creation, the appropriate privileges must be held. The privileges required depend on how the user ID will be specified in the DDL. A SET SQLID statement will be issued for each user ID. All the users specified must be on your secondary ID list.

The USERID Selection List screen assists in designating multiple user IDs for synonym creation.

Select one or more user IDs by entering **S** next to each ID to select. Press Enter. Selected user IDs are queued. Use the **S** (Shrink) command to view the queue. Once the user IDs have been selected, press the F3 (End) key to process and return to the Synonym screen. Once all information has been entered on the Create screen, press F3 (End) to process the creation.

Template Synonym Option

Although the Template option does not copy the user ID list with the synonym selected, it is included for product consistency. Enter a user ID list for the templated synonym if needed.

The Synonym Template screen is similar to the Synonym Create screen. Change the information or enter user IDs according to the procedures outlined in Synonym Create. When all information has been entered, press F3 (End) to process the template.

Alter Synonym Option

Although DB2 does not permit synonym alterations, we do. Change the name of the synonym or the table it references, or add new user IDs. DB2 alters the synonym by dropping and recreating it under the user ID of the original creator. Like the Create and Template sessions, an Alter Synonym session allows selection of tables and user IDs. The screen flow for an alter session is similar to a template session.

The Synonym Alter screen allows changes to the values used to create the synonym. The Synonym Alter screen is similar to the Synonym Create screen.

Change the information or enter additional user IDs, according to the procedures outlined in Synonym Create. When all information has been entered, press F3 (End) to process the Alter.

Synonym Drop Considerations

According to DB2 rules, a synonym can be dropped only by its creator. However, we let you drop synonyms created by other users. To drop a synonym for another user, that user's ID must be one of your secondary IDs.

Chapter 25: Alias

This section contains the following topics:

Overview of Aliases

Like a DB2 synonym, a DB2 alias is an alternate name for a table or view. However, an alias can name remote (and local) tables or views, whereas synonyms can only name local tables or views. Synonyms and aliases also differ in that synonyms can be accessed only by their creators. Aliases allow flexibility and familiarity by letting all users access them. Aliases can be used to simplify the remote name of a table or view.

Features for Maintaining Aliases

You can use the alias options to simplify DB2 alias processes. You can create, modify, and drop alias with minimal effort. The alias option includes the following features.

- Table Selection Help. Instead of entering a specific table name, enter selection criteria (*, %, _) to receive a list of tables meeting the criteria.
- Alter Alias. You can alter existing aliases. With DB2, only comment and label fields can be altered. There is no ALTER ALIAS statement within DB2.
- Template Alias. This option lets you create a new alias based on an existing alias.
- Compare Feature. The COMPARE mode of alias lets you check present changes to the original alias to see exactly what has been modified.
- Drop Alias. This option lets you drop an existing alias.

Accessing Alias Functions

Alias functions can be accessed by requesting to create, alter, template, or drop an alias. To create an alias, enter **C** in the Option field and **A** in the Object field of the main header and press Enter.

To alter, template or drop an alias, enter **A**, **T** or **D** in the Option field and **A** in the Object field of the main header and press Enter. Enter an alias name in the Item Name field or a user ID in the Creator field to get a more specific selection list. From the selection list, select the aliases to alter, template, or drop.

Selection Panels

When altering, dropping, or creating (by template) a view, a selection panel is presented to select the view if not already specified. Make the view selection list more specific by specifying selection criteria for the Item (view) Name and Creator ID prompts in the header.

Use the EQF facility to further specify the views displayed, and use scrolling commands to scroll through information.

Note: For more information about EQF and scrolling commands, see the *CA Database Management Solutions for DB2 for z/OS General Facilities Reference Guide*.

The View Selection screen displays all the fields available in the SYSIBM.SYSTABLES table. This information is available to make selection of the views easier.

To choose items from the listing, type **S** next to the items to select. Press Enter to register the selection.

Note: For information about the fields, press F1 (Help).

Editing or Browsing Alias Data

Edit or browse data in aliases by using RC/Edit.

Note: For more information, see the “Defining Profile Variables” and “RC/Edit” chapters.

Alias Commands

The Header and Compare commands can make working with aliases easier. Following is a brief description of how to use these commands:

COMPARE

Is useful when you are templating or altering an alias. The C (compare) command can be entered in the command line to see a summary of all changes made to the alias. The Compare screen shows the old and new versions of the alias. Use the RESET command to reset the attributes to their original status.

HEADER

Toggles the header on and off. Enter H in the command line to toggle the header.

Create Alias

Create Alias permits creation of an alternative name for an existing table or view. The Alias Create screen has several blank fields. Select the table or view for the alias to reference. If selection criteria are entered in the Table Name or Creator fields, a Table/View selection panel appears. The Table/View selection panel lists the tables and views and their creators. Select one table or view from the selection panel.

After selecting the base table, complete the alias definition by filling in the remaining blank fields. A Confirmation screen asks for a decision: accept, edit, or reject the alias being created.

Create Screen

The Alias Create screen provides an easy way to supply the information necessary for the alias creation.

Note: For information about the fields, press F1 (Help).

After information has been entered for all fields, press F3 (End) to process the alias creation. The Creation Confirmation screen appears.

Confirming the Alias Creation

The Creation Confirmation screen shows the DDL to be used to create an alias. Use the scrolling keys (F7 and F8) to view all the DDL. Accept the DDL to be used to create the alias by pressing Enter. Edit the DDL by entering E for Edit in the command line. Reject the DDL by pressing F3 (End).

The DDL is executed if Enter is pressed. When the Batch Processor is complete, press F3 (End) to return to the Main Menu.

Template Alias

Template Alias allows creation of a new alias using an existing alias as a template. In fact, a template session is actually a create session with the additional step of being able to select an alias as a starting point. The name of the templated alias must be changed. The other fields can be changed.

The Alias Template screen displays the values used to create the template alias. Change the fields for the new alias according to the procedures outlined in Create Alias. Once the changes have been made, press F3 (End) to process the template.

If multiple aliases are chosen, the selected aliases are processed in the same order as they appear on the selection panel. After the processing of the first alias has been confirmed, press F3 (End) for the next templated alias.

After entering all the changes in the alias fields and pressing F3 (End), the Confirmation screen appears. Accept, edit, or reject the DDL to be used to drop and re-create the aliases.

The DDL is executed if Enter is pressed. When the Batch Processor is complete, press F3 (End) to return to the Main Menu.

Alter Alias

Alter Alias enables alteration of existing aliases. Use Alter Alias to alter an alias's name, location, label, or comment. DB2 does not have an ALTER ALIAS statement. For DB2 to make alias changes to fields other than the label and comments, the alias must be dropped and re-created. CA RC/Update makes changes by dropping and recreating the alias.

A table/view or alias listing can be accessed using selection criteria. The screen flow for an Alter session is similar to a Create session.

The Alias Alter screen displays the values used to create the alias. The fields for this screen are similar to those on the Alias Create screen. Change fields according to the procedure outlined in the Create Alias. Once changes have been made, press F3 (End) to process the alter. If several aliases were chosen, they are processed in the same order as they appear on the Selection panel. After the processing of the first alias has been confirmed, press F3 (End) for the next alias.

After making changes and pressing F3 (End), the Confirmation screen appears. Accept, edit, or reject the DDL to be used to drop and re-create the aliases.

Note: If more than one alias was selected, each alias is processed individually. Processing occurs in the same order as the aliases are listed on the Selection panel. After confirming the processing of the first alias, press F3 (End) for the Alter Alias screen with the next alias.

The DDL is executed if Enter is pressed. When the Batch Processor is complete, press F3 (End) to return to the Main Menu.

Drop Alias

According to DB2 rules, an alias can be dropped only by its creator or by a user with SYSADM or SYSCTRL authority. However, we let you drop aliases created by other users. To drop an alias for another user, that user's ID must be one of your secondary IDs.

Note: For more information about dropping an object, see the “Using CA RC/Update” chapter.

DDL Execution

When F3 (End) is pressed from the Create, Alter, Template or Drop screen, a confirmation screen appears before the DDL is executed. Accept, edit, or reject the DDL to be used to complete the request.

Note: For more information, see the “Using CA RC/Update” chapter.

If your changes are complex, require dropping and recreating objects (and their dependencies), or the current CA RC/Update Operation Mode is set to A (RC/Alter), you can perform a complete analysis of your changes that includes rebuilding the DDL and optionally including utility statements and extended DDL generation and analysis options.

Chapter 26: Join/Edit

This section contains the following topics:

Overview of Join/Edit

You can use Join/Edit to set up a query against several joined tables without the need for detailed knowledge of SQL. Queries can be made immediately or saved as DB2 views.

Note: When you create, alter, or template a join/edit, this does not necessary mean that you are editing the tables. It can also mean that you are browsing the tables.

Access Join/Edit

You can access Join/Edit as described in this section.

To access Join/Edit, specify **J** in the Object field of the Main Menu, specify **C**, **A**, or **T** in the Option field, and then press Enter.

Depending on the specified option, the Join Create, Join Alter, or Join Template screen appears.

Create a Join

You can create a join as described in this section.

Follow these steps:

1. From the main menu, perform the following actions:
   a. Type **J** in the Object field.
   b. Type **C** In the Option field.

   Note: You can also specify **A** to alter or **T** to template.

Press Enter.

The Join Create screen appears.

The status of the CAPS and NULLS commands is shown in the first line of this screen. The header fields (Option, Object, Mode, Item Name, and Creator) determine the execution mode, the object type, option, and the selection criteria for selecting the objects.

Note: The Where header field is not functional for this object type. For information about these fields, see the "Using CA RC/Update" chapter.

2. Select your tables. Enter the main table for the query in the Table field.

   The first part of this field is the table creator and is initially copied from the Creator field in the header. You can enter the full name or use the percent sign (%) to display a selection list. For example, USERID2.J% will produce a list of all tables created by USERID2 starting with J. On the previous screen, USERID1.jemptab is specified.

3. Set up the SELECT list by selecting the secondary tables that will be joined to the main table. As you select tables, their columns are displayed at the bottom of the screen.

   Note: The table names are prefixed by a table correlation character: A for the main table, B for the second table, and so on.

4. Specify a value in the DS field to indicate whether to display the column as follows:

   Y

   Includes the column in the displayed data.

   S

   Displays and sort the column.

   D

   Displays and sort the column in descending sequence. The sort order follows the display order.

   Any other value will cause the column not to be displayed. Where more than one sort field is specified, the sort sequence will be the same as the sequence in which the columns are displayed.

   Note: You can rearrange the columns with ISPF-like commands. The # field displays a sequence number for referencing display lines.

5. Specify a list of columns to be selected from the table for display in the SELECT field (see DS field above) or as the 'left hand' side of a JOIN or WHERE clause.

   You can enter values (preceded by the table correlation character and a period) directly in the SELECT, JOIN and WHERE fields (see following values), but it is easier to select them from the column lists. Fields that can be edited are indicated by underscore or hyphen characters.

   Note: For information about the fields, press F1 (Help).

   SELECT

   Specifies a list of columns to be selected from the table for display (see DS) or as the 'left-hand' side of a JOIN or WHERE clause.

   JOIN

   Specifies a list of columns appearing on the 'right-hand' side of a JOIN clause. For example, if in a given row the SELECT column is A.DEPT and the JOIN row is B.DEPT, the tables will be joined on those two columns. In other words, a JOIN statement in the following format will be generated:

   ```
   JOIN table B ON A.DEPT = B.DEPT
   ```

   Note: An equal sign is always implied.

   If two or more columns are specified in the JOIN column, they are linked by AND. If the JOIN cannot be satisfied in this way, consider defining the JOIN by using a WHERE clause instead. An example of a WHERE clause is as follows:

   ```
   WHERE A.DEPT=B.DEPT
   ```

   WHERE

   Identifies columns that appear on the 'right-hand' side of a WHERE clause. Note the following:

   - Literals can also be entered. For example, if SELECT contains A.DEPT and WHERE contains > '0001', only rows where the dept column has a value greater than '0001' will be selected. Literals can be continued to the next line in the WHERE column for a maximum size of 40 characters (including quotes if necessary). For a literal to be recognized as a continuation, the SELECT and JOIN columns must be blank.
   - The = column can be changed to any 1 or 2 character comparator that is valid in DB2 (>, =>, and so on).
   - LIKE should be specified as LK.
   - BETWEEN ... AND should be specified as BT...&.

- If more than one WHERE column is specified, an AND will be inserted between the comparisons by default. To override this default:

  Insert the word OR on a blank line between the columns (insert a new line if necessary); type over the word AND if it appears.

- If you import a view with a complicated WHERE clause into Join/Edit, opening and closing parentheses will each be displayed as a single entry in the WHERE list.

6. Set up a join by placing the column for the 'right-hand side' of the JOIN clause in the JOIN list next to the column in the SELECT list, which corresponds to the 'left-hand side' of the JOIN (add the 'left-hand' column if necessary).

   a. Display this column by blanking out the DS field.

   b. Type a value for the join type in the Type field as follows:

      I

      Omits rows if the 'left-hand' column or the 'right-hand' column in the JOIN clause does not have a counterpart value.

      O

      Includes rows without counterpart values.

      R

      Includes rows where the right-hand column is null.

      L

      Includes rows where the left-hand column is null.

   You can specify more than one pair of join columns, in which case they will be linked by 'AND'.

   A JOIN is always assumed to be on equality, that is ON A = B.

7. Set up the WHERE clause, as you did for the JOIN clause, by placing the 'right-hand' column in the WHERE list in the row where the corresponding 'left-hand' column appears.

   The first sub-field (2 characters wide) is where you place the comparison type. Valid values for DB2 are:

   `= < > <> <= >= !< !> != ^< ^> ^= IN LK`

   Note: The program will only check that this field is not blank. Further validation is left to DB2. LK is accepted as an abbreviation for LIKE. You can also type literal values in the WHERE column. If more than one WHERE clause is specified, they will be assumed to be joined by AND, unless the operator OR is placed on an empty line between the two columns in the WHERE list.

8. Enter one of the following values in the CMD line:

   - Enter **ALL** to select all columns from all tables displayed.
   - Enter **S** to SELECT an individual column and place it in the next available entry in the SELECT list.
   - Enter **S**, **J**, or **W** plus a row number to overwrite a displayed entry in the SELECT, JOIN or WHERE lists.

9. Specify a value in the Save field as follows:

   - Enter **Y** in the Save field to save the data as a view.

     Note: To edit the data, a view must be created and saved. If any of the tables in the join do not have a primary key defined, you can only browse, not edit after a save.

   - Enter **N** in the Save field to browse the data using SQL.

     This is the default. It is not necessary to save the join unless you want to reuse it in the future.

10. Specify **Y** in the Check Option field (to add the DB2 WITH CHECK OPTION to a view).

11. Press F3 (End) to create SQL ready to be processed by the Batch Processor.

    If you specified Y for Save, a CREATE VIEW syntax appears; otherwise, a direct SELECT appears. You can edit this SQL by entering ED or EDIT in the Command line.

12. Press F3 to exit.

13. Press Enter to process the SQL.

Avoiding Clutter from VIEW Objects

To prevent cluttering up the DB2 catalog with VIEW objects created once, use the parmlib join/edit creation of VIEW objects keyword. The keyword has the values:

N

Accepts only SAVE=N on the Join Create panel. This setting provides total avoidance of clutter.

E

Accepts SAVE=Y on the Join Create panel. This setting does not restrict clutter.

B

Accepts SAVE=Y on the Join Create panel but only creates a permanent VIEW if the name of the VIEW is TEMPJVIEW. Clutter is limited to one VIEW per creator.

Column Selection

To select all columns in a table, specify an asterisk (*) next to the table name. The columns will be added into blank SELECT rows, skipping over any non-blank entries.

Note: If two or more tables contain the same column name, do not select the second and succeeding column names. You will receive an sqlcode:

```
-612, columnname is a duplicate column name
```

To select an individual column, specify an S next to a column name to move it to the first blank SELECT row. You can also specify S*nn* to overwrite a displayed row, where *nn* is the number of the row displayed under #. J*nn* and W*nn* can also be used to move column names to specific JOIN and WHERE rows.

Note: You do not have to delete trailing hyphens in the SELECT, JOIN, or WHERE fields as they will be removed automatically.

Example

An example of what might appear after column selection is shown on the following screen:

```
ROPJCRC -- (CAPS ON)  ---- Join Create --- (NULLS ON)  ---
COMMAND ===>                                                SCROLL ===> PAGE

Option     ==> C                 Object    ==> J        Mode  ==> O ONLINE
Item Name  ==> J1                Creator   ==> USERID1  Where ==> N
SSID: D81B ------------------------------------------------------------ USERID1
View  ==> USERID1  >.                 > Save => N         Check Option ==> N
Table ==> USERID1  >.                  >


CMD # DS SELECT.............. JOIN.............TYPE WHERE.................
___ 01 Y A.LASTNAME          -------------------- I =  --------------------
___ 02 D A.FIRSTNAME         -------------------- I =  --------------------
___ 03 Y B.DEPTITLE          -------------------- I =  --------------------
___ 04   A.DEPT              B.DEPT               I >  '0000'--------------
___ 05 Y -------------------- -------------------- I =  --------------------
___ 06 Y TIMESTAMP---------- --------------------   BT '2001-01-01-00.00.00
___ 07 Y -------------------- --------------------      .000000'------------
___ 08 Y -------------------- --------------------   &  '2001-12-31-11.59.59
___ 09 Y -------------------- --------------------      .999999'------------


Top area Up/Down: PF7/8 ********************** Bottom area Up/Down: PF10/11
_ A.USERID1.JEMPTAB
___    1 LASTNAME          CHAR          ___    2 FIRSTNAME         CHAR
___    3 DEPT              CHAR          ___
_ B.USERID1.JDEPTTAB
___    4 DEPT              CHAR          ___    5 DEPTITLE          CHAR
___    6 TIMESTAMP         TIMESTMP
******************************** BOTTOM OF DATA ********************************
```

In this example:

- You want to select A.LASTNAME, A.FIRSTNAME, and B.DEPTITLE for display, sorted in descending order of A.FIRSTNAME.
- You want to JOIN table A (USERID1.JEMPTAB) to table B (USERID1.JDEPTTAB) on the fields A.DEPT and B.DEPT. This is an INNER JOIN, so null values will be excluded.
- You want to include rows that have DEPT higher than '0000' and a TIMESTAMP between the two values specified (note the use of the continuation column for the timestamp literals here).
- Row 05 will be ignored because it is empty (it has no SELECT column).

When you are satisfied with the data, you can run the SQL or save it as a DB2 view:

- To generate a view, set the Save field to Y and if necessary, enter as the View Creator one of the user IDs to which you have access. SORT information cannot be saved in a view and will therefore be lost.
- To generate the SQL, press F3.

Join Template

The template option allows creation of a new join using an existing join as a template. A template session is actually a create session with the additional step of selecting a join as a starting point. You must change the name of the templated join, but the other fields do not have to change.

Note: Dependent objects are not included in the template operation.

More information:

Create a Join (see page 491)

Join Alter

The Alter Join option lets you change any characteristic of a join. Change fields as necessary, according to the procedure outlined in Join Create.

DDL Execution

When F3 (End) is pressed from the Create, Alter, Template or Drop screen, a confirmation screen appears before the DDL is executed. Accept, edit, or reject the DDL to be used to complete the request.

Note: For more information, see the “Using CA RC/Update” chapter.

If your changes are complex, require dropping and recreating objects (and their dependencies), or the current CA RC/Update Operation Mode is set to A (RC/Alter), you can perform a complete analysis of your changes that includes rebuilding the DDL and optionally including utility statements and extended DDL generation and analysis options.

Join/Edit Commands

The following commands can be used when processing join/edits:

ALL

Selects all displayed columns.

CAPS

Sets translation to uppercase or lowercase for fields for which it is optional. Specify **CAPS ON** or **CAPS OFF**.

COLS

Controls the display of the column area at the bottom of the screen. Specify **COLS ON** or **COLS OFF**.

HEADER

Controls the display of the header area at the top of the screen. Specify **H** in the command line to toggle the header.

NC/CN

Inverts NULLS and CAPS settings.

NULLS

Sets nulls on or off in displayed fields. Nulls On allows insertion of extra characters in fields.

Note: Other commands are available through the SQL Editor. For more information about these commands, see the online help or the *General Facilities Reference Guide*.

Appendix A: Command Summary

This section contains the following topics:

Notation Conventions

The following table describes how notation conventions are used:

Notation	Description
UPPERCASE characters	Must be entered as shown.
lowercase characters	User-specified variables.
<column name>	The full DB2 column name or the RC/Edit column abbreviation.
( )	Must be entered where shown.
[ ]	Enclose optional parameters, choose 1.
[[]]	Enclose optional parameters, choose 1 or more.
{ }	Enclose required parameters.
{{ }}	Enclose required parameters, choose 1 or more.
|	Or, as in Y | N.
Underlined	Abbreviations.

RC/Objects Commands

The following table lists the RC/Objects commands that are available from object screens:

Command	Syntax	Abbr	Mode
ALL	ALL	ALL	V
COMPARE	COMPARE	C	A, T
EQU	EQU	EQU	SC
MODEL ON/OFF	MODEL	M	RID
TEXT	T	T	V
TOTALS	T	T	SC

Valid mode values are:

- ALL—All object screens and selection lists
- A—Alter
- T—Template
- RID—Referential integrity drop
- SC—Space calculations
- V—View

Scrolling Commands

The following commands control the scrolling of the column definition section on the table screen.

Note: For more information about the scrolling commands, see the *CA Database Management Solutions for DB2 for z/OS General Facilities Reference Guide*.

EXCLUDE

Excludes an attribute from the column list on the Table Screen.

SHOW

Displays the column attribute again that was excluded. This can be helpful to limit the column attributes displayed on the screen.

FIND

Finds a string within the column list. FIND can be column specific or global.

RFIND

Repeats the previous find. RFIND can be column specific or global.

FREEZE

Lets users freeze attribute columns for horizontal scrolling. The CMD, ### (column number), and COLUMN NAME attributes are automatically frozen.

MELT

Removes column freezes.

UNFREEZE

Removes column freezes.

SCROLLB

Sets the horizontal scrolling mode to BYTE.

SCROLLC

Sets the horizontal scrolling mode to COLUMN (the default).

QPRINT

Prints the table's column list.

PPRINT

Prints the columns displayed on the current page.

SORT

Is not functional for the column list.

RC/Edit and RC/Browse Commands

The following table shows the RC/Edit and RC/Browse commands. These commands can also be used on the RI/Edit and RI/Browse screens.

Command	Syntax	Abbr	Mode
CAPS ON	CAPS ON [*column_name*]	N/A	E
CAPS OFF	CAPS OFF [*column_name*]	N/A	E
CHANGE	CHANGE *old_value new_value* [*column_name*] [ALL] [X | NX | Y | NULL | N | ¬NULL]	C	E
COMMA On/Off	COMMA	N/A	E,B

Command	Syntax	Abbr	Mode
END	END	N/A	E,B
EXPLODE	EXPLODE	N/A	E,B
HEX	HEX	N/A	E,B
LCASE	LCASE *column_name*	N/A	E
RESET	RESET	RES	E
RCHANGE	RCHANGE	N/A	E
SAVE	SAVE	SA	E
SET	SET *column_name* = *new_value* [where ...] Available only with Searched Update method of editing.	N/A	E
SHOW	SHOW	S	E
SORT	SORT [*column_name*] [A | D]]	N/A	E,B
SORT?	SORT?	N/A	E,B
SQL	SQL	N/A	E,B
TRACE	TRACE [ON | OFF]	N/A	E
TYPE	TYPE	N/A	E,B
UCASE	UCASE *column_name*	N/A	E

Valid mode values are:

- E—RC/Edit
- B—RC/Browse

RI/Edit and RI/Browse Commands

The following table shows the RI/Edit and RI/Browse commands. This includes commands from RC/Edit and RC/Browse Commands.

Command	Syntax	Abbr	Mode
ACTIVE	ACTIVE [.sequence number]	ACT	E,B
CHECK ON	CHECK [ON]	N/A	E
CHECK OFF	CHECK [OFF]	N/A	E
CURRENT	CURRENT [.sequence number]	CUR	E,B

Command	Syntax	Abbr	Mode
LINK	LINK [creator.tablename]	L	E,B
MAX	MAX	N/A	E,B
MIN	MIN	N/A	E,B
RELATED	REL [.sequence number)	REL	E,B
REMOVE	REMOVE [.sequence number]	REM	E,B
RSIZE	RSIZE	N/A	E,B
SIZE	SIZE	N/A	E,B
SYNC ON	SYNC [ON]	N/A	E,B
SYNC OFF	SYNC OFF	N/A	E,B
TREE ON	TREE [ON]	N/A	E,B
TREE OFF	TREE OFF	N/A	E,B

Valid mode values are:

- E—RC/Edit
- B—RC/Browse

RC/Copy Commands

The following table lists RC/Copy commands that are available from the Column Mapping screen:

Command	Syntax	Abbreviation
AUTOMAP	AUTOMAP [N | T | L]	AM
RESET	RESET	RES

EQF Commands

The following table lists EQF commands that are available on the EQF Data Query screen:

Command	Syntax	Abbreviation
ALL	ALL	Not Applicable

Command	Syntax	Abbreviation
ALL NUMBER	ALL NUMBER	Not Applicable
AUTOSEL	AUTOSEL	Not Applicable
CANCEL	CANCEL	CAN
COUNT	COUNT	Not Applicable
END	END	Not Applicable
RESET	RESET	RES
SAVE	SAVE	Not Applicable
SHRINK	SHRINK	Not Applicable
SORT	SORT [SEL | COL]	SORTS SORTC
SQL	SQL	Not Applicable
TYPE	TYPE	Not Applicable

ISQL Commands

The following table lists ISQL commands that are available in the SQL Editor.

Command	Syntax	Abbreviation
CAPS ON/OFF	CAPS	C
COLS ON/OFF	COLS	Not Applicable
EXPLAIN	EXPLAIN	EXPL
NULLS ON/OFF	NULLS	N
PARSE ON/OFF	PARSE	Not Applicable
PRED	PRED	Not Applicable
RESET	RESET	RES
STAND	STAND	Not Applicable
SYNTAX	SYNTAX	Not Applicable
TEXT ON/OFF	TEXT	T
WORD WRAP	WW	WW
WORD SPLIT	WS	WS

These commands are available only if you have a license for CA SQL-Ease: EXPLAIN, PRED, STAND, and SYNTAX.

Batch Processor Commands

The following table lists the Batch Processor commands:

Notation	Description
ALLOC	.ALLOC {FILE(ddname) | DDNAME(ddname)} {DATASET('name') | DUMMY} [[UNIT(type) | VOLSER(serial-list) | STORCLAS(storage-class) | DATACLAS(data-class) | MGMTCLAS(management-class) | [OLD | SHR | MOD | NEW] | [KEEP | CATALOG | DELETE | UNCATALOG] | SPACE(qty,increment) | BLKSIZE(value) | LRECL(value) | RECFM(A,B,D,F,M,S,T,C, and | or V) | DIR(value) | RLSE | CONTIG | [BLK (value) | TRACKS | CYL] | LABEL(type | RETPD | EXPTD) POS(seq_no)]] VOLREF (dsname) RETAIN
	.ALLOC FI(SYSOUT) {SYSOUT(class) | SYSOUT(*)} [[DEST(destination) | FORM(form-ID) | COPIES(value)]]
AUTH	.AUTH [userid] Batch mode only.
CALL UTIL	.CALL UTIL DB2-utility-name PARM() Batch mode only.
CALL AMS	.CALL AMS
CALL DSN	.CALL DSN PARM(sub-system)

Notation	Description
CALL program	.CALL program-name PARM() + | INDDn() ALLOC(YES | NO) OUTDDN() ALLOC(YES | NO)
CONNECT	.CONNECT subsystem
CONTROL	.CONTROL {{BPID(name) PLAN(plan_name) LOGID(subsystem) | ABEND | UNIT(type)}}
DATA	.DATA -- input stream -- .ENDDATA
DISCONN	.DISCONN
ENDDATA	.DATA -- input stream -- .ENDDATA
EXIT	.EXIT [return-code]
FREE	.FREE {FI(ddname) | DA('dataset')}
LIST	.LIST [[{TERM | NOTERM} | {FILE(dataset) | NOFILE} | {SYSOUT(class,forms,destination) | NOSYSOUT}]]
MSG	.MSG {{LOG | OPER(descriptor,route) | USER(userid)}} 'message'
OPTION	.OPTION [{NOLOG | LOG} | {AUDIT(LONG | SHORT)} | {NOERRORS | ERRORS} | {NOSQL | SQL | NOLOAD | LOAD} | {NOUNLOAD | UNLOAD} | {NORUNSTATS | RUNSTATS} | {NOCOPY | COPY} | ROWLIMIT(value) | MAXCHAR(value)]
RESTART	.RESTART {OVERRIDE | SYNC | SYSTEM(name,...)}
SYNC	.SYNC value 'description'
SYSTEM	.SYSTEM name

Appendix B: Static SQL Performance

This section contains the following topics:

Overview of Static SQL

A static SQL statement is a statement that is defined at the time the program is bound. Static SQL does allow for the substitution of values for WHERE predicates at execution time, but it does not allow changes to table names, columns, or actual WHERE conditions defined within the SQL statement. Static SQL is used whenever feasible. The advantages of static SQL over dynamic SQL:

- A dynamic bind of an SQL statement is not performed at execution time. This decreases the processing time necessary to execute the SQL statement.
- The user needs EXECUTE authority only on the plan, not on the underlying tables (such as the DB2 system catalog tables) processed by the plan. By having EXECUTE authority on the plan, the user automatically gains the necessary authority to execute all static SQL contained within the plan.

Usage

All attempts are made to use static SQL. The only instance where static SQL is not used is when the very nature of the component requires the use of dynamic SQL. In many organizations, SELECT authority on the DB2 system catalog tables is not given to users. Because we use static SQL against the system catalog tables, SELECT authority is not needed for system catalog tables; only EXECUTE authority is needed on the CA RC/Update plan.

Many organizations also create views for the DB2 system catalog tables to limit the data a user can see. Options are provided that let you use views instead of the actual system catalog tables. These options are discussed later in this appendix.

Note: This appendix discusses performance issues along with special installation options that permit customization. Reading this appendix is not required reading for using the product.

Object Selection Panels-Static

Object selection lists display information about the DB2 objects available on the DB2 subsystem. An object selection list appears whenever the user is requested to select an object for processing. Static SQL is used to produce an object selection list unless one of the following is used.

- Extended Query Facility. Because you are dynamically entering WHERE conditions to control the selection lists, static SQL cannot be used. Dynamic SQL is only used for the object selection lists referencing EQF. All other object selection lists remain static. When to EQF is removed for an object selection list, static SQL can again be used.
- Alternate Catalog Mapping. Alternate Catalog Mapping permits definition of alternate table (or view) names for the system catalog tables. Because alternate names are not known until execution time, dynamic SQL is used.

  Note: For more information about ACM, see the *CA Database Management Solutions for DB2 for z/OS Value Pack Reference Guide*.

RC/Objects-Static

RC/Objects uses static SQL for all its processing.

CA Batch Processor-Dynamic

When generated through the CA Technologies Batch Processor, the DDL is executed as dynamic SQL. The appropriate DB2 authorities must be held before executing the generated DDL. To reduce possible system catalog contention, execute the generated DDL in batch mode during low DB2 usage.

The ability to write the generated DDL to a batch data set for later execution permits the separation of functions within an organization: The individual generating the DDL does not require the necessary authorities to execute the DDL; the execution can be performed by another user with the necessary authorities.

RC/Edit-Static/Dynamic

Due to the nature of the processing performed by RC/Edit, dynamic SQL is used for all processing except for the authorization checking performed against a table entering edit or browse mode. The edit authorization checking is done with static SQL. Therefore, the user does not need SELECT authority on the DB2 authorization tables.

RC/Alter-Static

The DDL to perform a non-supported DB2 alteration of an object is completely generated by static SQL.

ISQL-Dynamic

Due to the very nature of ISQL, all statements are executed dynamically. The necessary DB2 authorities must be held to execute the entered SQL or PBP statements.

RC/Copy-Dynamic

Because it is unknown which tables will be involved in the copy process, dynamic SQL is used.

Index

E

F

G

H

I

J

S

T

U

V