Siebel

Implementing Siebel Business Applications on DB2 for z/OS

January 2026

Siebel
Implementing Siebel Business Applications on DB2 for z/OS

January 2026

Part Number: F87457-02

Authors: Siebel Information Development Team

Contents

Preface

This preface introduces information sources that can help you use the application and this guide.

Using Oracle Applications

To find guides for Oracle Applications, go to the Oracle Help Center at *https://docs.oracle.com/*.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the *Oracle Accessibility Program website*.

Contacting Oracle

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit *My Oracle Support* or visit *Accessible Oracle Support* if you are hearing impaired.

Comments and Suggestions

Please give us feedback about Oracle Applications Help and guides! You can send an email to: *oracle_fusion_applications_help_ww_grp@oracle.com*.

1 What's New in This Release

What's New in Implementing Siebel Business Applications on DB2 for z/OS, Siebel CRM 26.1 Update

The following information lists the changes in this revision of the documentation to support this release of the software.

What's New in Implementing Siebel Business Applications on DB2 for z/OS, Siebel CRM 26.1 Update

Topic	Description
Siebel Scripting Considerations	New topic. Describes best practices for managing cursor modes in Siebel scripts when querying DB2 on z/OS and OS/390.

What's New in Implementing Siebel Business Applications on DB2 for z/OS, Siebel CRM 21.3 Update

The following information lists the changes in this revision of the documentation to support this release of the software.

What's New in Implementing Siebel Business Applications on DB2 for z/OS, Siebel CRM 21.3 Update

Topic	Description
Applying Schema Extensions to the Local Development Database Applying Schema Changes to Other Local Databases	Removed topics from *Customizing a Development Environment*. Siebel Tools development now uses the enterprise Siebel database only, and no longer uses a local database.

What's New in Implementing Siebel Business Applications on DB2 for z/OS, Siebel CRM 20.1 Update

No new features have been added to this guide for this release. This guide has been updated to reflect only product name changes.

What's New in Implementing Siebel Business Applications on DB2 for z/OS, Siebel CRM 19.1 Update

No new features have been added to this guide for this release. This guide has been updated to reflect only product name changes.

2 Preparing to Deploy Siebel CRM on DB2 for z/OS

Preparing to Deploy Siebel CRM on DB2 for z/OS

This chapter provides information to help you plan your implementation of Oracle's Siebel CRM using IBM DB2 Version 11.1 for z/OS (hereafter referred to as DB2 for z/OS). It contains the following topics:

- *About Deploying Siebel CRM on DB2 for z/OS*
- *Preparing to Deploy Siebel CRM on DB2 for z/OS*
- *About the Deployment Planning Worksheet*
- *About File Path and Directory Naming Conventions*

About Deploying Siebel CRM on DB2 for z/OS

Many of the tasks involved in installing and configuring Siebel CRM are the same for all database platforms. However, there are several considerations you need to keep in mind when implementing Oracle's Siebel CRM using DB2 for z/OS, and several implementation tasks that are specific to the DB2 for z/OS platform. These considerations and tasks are described in depth throughout this guide.

For general information on installing the Siebel Enterprise Server components, including the Siebel Gateway, Siebel Server, and Database Configuration Utilities, see the *Siebel Installation Guide* . This guide describes how to install and configure the Siebel Schema on DB2 for z/OS.

Preparing to Deploy Siebel CRM on DB2 for z/OS

Preparing to deploy Siebel CRM on DB2 for z/OS involves the following steps:

1. Review the information in the Certifications tab on My Oracle Support for your Siebel application version and make sure that all requirements are met.

   For DB2 databases, the requirements include:

   - Installing a supported version of DB2 Connect and the appropriate IBM Fix Pack, as specified in the Certifications tab.
   - Ensuring that the z/OS Unicode Conversion Services have the conversions required by the code pages for your DB2 subsystem.

     > **Note:** The Siebel Bookshelf is available on Oracle Technology Network (http://www.oracle.com/technetwork/indexes/documentation/index.html) and Oracle Software Delivery Cloud. It might also be installed locally on your intranet or on a network location.

2. Review the Siebel architecture of the DB2 for z/OS database server.
3. Decide on the DB2-related architecture and other related choices:
    - Understand and design the database storage layout for the Siebel Schema.
    - Review possible table space partitioning.

      If you plan on using batch processing through Siebel EIM, ensure the partitioning scheme is appropriate for both batch and online activities.
    - Decide whether to use a separate or shared DB2 subsystem for production.
    - Configure the DB2 Data Distribution Facility and the z/OS Workload Manager (WLM).
    - Review and set up the required and recommended DB2 system parameters (DSNZPARMs).
    - Decide whether to select an ASCII, EBCDIC, or Unicode encoding scheme code page. (Unicode is required for new deployments.)

      > **Note:** All environments in the organization (development, test, production, and others) must use the same encoding choice.

    - Decide whether to implement Coordinated Universal Time (UTC). (UTC is required for new deployments.)
    - Decide which security scheme to use, either native DB2 security or Lightweight Directory Access Protocol (LDAP).
    - Decide on backup and recovery mechanisms.
    - Decide on maintenance procedures, such as RUNSTATS and REORG.
    - Ensure you know how to deploy the user-defined functions (UDFs) and stored procedures that Siebel CRM requires.
    - Specify the appropriate Siebel parameters and environment for your deployment after the installation is completed.
    - Set up development, test, and other environments.
4. The implementation team must review Siebel Alerts and *Siebel CRM Update Guide and Release Notes* on My Oracle Support to make sure that they are aware of known anomalies.
5. Establish a project staff:
    - Dedicate database administrators (DBAs) to support the Siebel application deployment.
    - Establish good working relationships with midtier administrators because some operations, such as installation or upgrade, require close cooperation with these groups of people.
    - DB2 DBAs using the Siebel application for the first time need to be aware that in the Siebel application:
        - There are many database objects, some of them unused.
        - The Siebel data model uses different data types, mainly VARCHAR.
        - Dynamic SQL is generated versus traditional SQL precisely coded by developers.

About the Deployment Planning Worksheet

Before starting your DB2 for z/OS implementation, make a copy of the worksheet in *Deployment Planning Worksheet*. Use this copy to record information about your deployment team and your deployment environment.

Distribute a copy of the completed worksheet to each member of the deployment team. You will need the information you have recorded on the worksheet at various stages during the implementation process. Record changes to your deployment environment on the worksheet for future reference.

About File Path and Directory Naming Conventions

This guide uses the following file path and directory naming conventions to specify file locations:

- SIEBEL_ROOT. This location is the main Siebel release installation directory. Enterprise Server components, such as the Siebel Gateway and the Siebel Servers, are installed into this directory. For example, your top-level installation directory might be `c:\siebel\ses` (Windows) or `/siebel/ses` (UNIX).

  > **Note:** $SIEBEL_ROOT (no italics) represents the value of the SIEBEL_ROOT environment variable, which usually corresponds to a module-specific installation directory, for example, `/siebel/siebsrvr` for the Siebel Server.

- SIEBSRVR_ROOT. This location is the Siebel Server installation directory, such as `c:\siebel\ses\siebsrvr` (Windows) or `/siebel/ses/siebsrvr` (UNIX).
- DBSRVR_ROOT. This location is the directory in which the Siebel Database Configuration Utilities are installed on the Siebel Server computer. For example, the Database Configuration Utilities might be installed in `c:\siebel\ses\dbsrvr` (Windows) or `/siebel/ses/dbsrvr` (UNIX).

Use lowercase for all file names, directory names, path names, parameters, flags, and command-line commands, unless you are instructed otherwise. For more information, see the *Siebel Installation Guide* .

3 Security Concepts for a DB2 for z/OS Environment

Security Concepts for a DB2 for z/OS Environment

This chapter discusses the security concepts that are important in running Siebel CRM on DB2 for z/OS. This chapter also describes the roles and permissions you require to implement Siebel CRM on DB2 for z/OS and to set up a connection between the Siebel Server and the Siebel database.

This chapter includes the following topics:

- *About Siebel Application Data Security*
- *Operating System Security*
- *User Password Change and Expiration*
- *About Changing z/OS Passwords from Client Computers*
- *DB2 z/OS Authorization IDs*
- *Data Transmission Security for Siebel Clients*
- *Required Authorizations*

For a general discussion of Siebel CRM architecture, including a discussion of supported client types, see *Siebel Deployment Planning Guide* and the *Siebel Installation Guide* .

About Siebel Application Data Security

All users must provide a user ID and password to connect to Siebel CRM applications, regardless of whether they access them through the Siebel Web Client or through the Siebel Developer Web Client or the Siebel Mobile Web Client.

Each user must be preregistered within the Siebel application and associated with a unique record. This unique record determines a user's access to data by association with Positions and Responsibilities:

- Responsibilities determine the screens and views that are available to an employee.
- Positions determine the data rows that appear in those views, and the data rows that are synchronized to a Mobile Web Client's local database.

Siebel CRM allows you to control user access to information from within the application. Because the need to know specific information is usually related to a particular responsibility, data access is defined by responsibility rather than by user.

Attachments that are not stored in the database are stored in a compressed, encrypted format on the Siebel File Server. Attachments are linked to data rows, and access to attachments is therefore restricted by a user's responsibilities and position. For more information on these topics, see *Siebel Security Guide* .

Operating System Security

The DB2 Connect middleware passes to DB2 the user ID and password used to access Siebel CRM applications to establish a connection. The user ID and password can be user credentials supplied by the external security adapter, if you are using an external security adapter.

The Security Administrator on z/OS must preregister all user IDs and passwords passed through DB2 Connect with the z/OS security package in use, for example, Resource Access Control Facility (RACF), ACF/2, or TOP SECRET. The z/OS Security Administrator can associate the user ID with a secondary authorization ID (also known as the Security Group ID on the *Deployment Planning Worksheet*) within the z/OS security package to simplify database security privilege administration.

DB2 Connect supports password encryption on the DB2 for z/OS platform.

> **Note:** The password rules for Siebel user IDs for DB2 are determined by the operating system security interface (RACF, ACF/2, or TOP SECRET).

User Password Change and Expiration

Because of security constraints, if you are using Siebel CRM on z/OS, user passwords might be set up to periodically expire. User passwords can be changed as follows:

- **When a password has expired.** In Siebel CRM, expired user passwords can be changed from the login window without administrative intervention.
- **When a password has not yet expired.** If a user password has not yet expired, a user can change the password by amending the user preferences in the client application. For information on amending user preferences, see *Siebel Fundamentals* .

  > **Note:** A user's access to the functionality in the User Preferences screen depends on how the Siebel application is configured. For more information, contact your Siebel administrator.

About Changing z/OS Passwords from Client Computers

To change z/OS passwords from client computers, set the DB2 extended security option (EXTSEC) to `YES`. The default DB2 value for extended security is `NO`. There are two ways to set the extended security option to `YES`:

- In the DSNTIPR installation panel, change EXTENDED SECURITY to `YES`.
- In the DSN6SYSP macro within the DSNZPARM creation job, set option EXTSEC to YES.

Setting the extended security option to `YES` enables the following two functions:

- Users on client workstations can change their z/OS passwords without signing onto TSO.
- Siebel CRM receives descriptive error codes generated by DB2 when security violations occur.

DB2 z/OS Authorization IDs

By default, DB2 restricts access to database resources unless privileges are specifically granted. A complete description of DB2 security is available in the vendor documentation on the IBM Web site. Access to database resources is established using one or more of the following authorization IDs:

- **Primary authorization ID.** The user ID used to log into DB2. All users have a primary authorization ID.
- **Secondary authorization ID.** Generally known as a group ID. Siebel CRM effectively uses secondary authorization IDs that are enabled through the DB2 installation exit, DSN3@ATH. IBM provides a sample exit for setting secondary authorization IDs. For information on using secondary authorization IDs, see *About Using a Secondary Authorization ID*.
- **Package owners.** The ID of the owner of a DB2 Connect package. The package owner ID is used for authorization purposes if the package is bound with DYNAMICRULES(BIND) on the BIND command. For more information on DB2 Connect packages, see an IBM Corporation Redbook on the subject.

About Using a Secondary Authorization ID

Using a secondary authorization ID significantly reduces the administrative tasks associated with database security. The administrator grants privileges only once to a secondary authorization ID rather than to each Siebel CRM user.

> **Note:** When you install the Siebel Schema, you are prompted to enter a Security Group ID/Grantee. This is the same as a secondary authorization ID.

During the Siebel Schema installation process, you can specify a secondary authorization ID for client access with the default group of SSEROLE. The installation process generates the appropriate SQL grant statements for that group to allow INSERT, UPDATE, SELECT, and DELETE authority to application tables. Furthermore, that same group is specified in a SET CURRENT SQLID statement so that reuse of the statement cache is maximized. Therefore, it is important that the selected group is among the list of secondary authorization IDs for all users of the applications.

Grant Statements for Additional Secondary Authorization IDs

You must create secondary authorization IDs separately. Siebel CRM includes the grantstat.sql script; this script generates grant statements which allow access to interface tables. For a discussion of the grantstat.sql script, see *Granting Table Privileges*.

Either the table owner, or users with DBADM or SYSADM privileges, must execute the grant statements. To disable a grant, issue a revoke statement.

About Using an External Security Adapter on z/OS

An external security adapter is an interface that lets you use an external system to authenticate users. For example, you might employ an LDAP repository, a protocol for storing and retrieving directory-related information that includes authentication services. LDAP can reside on the mainframe.

When users log in to the Siebel application, the external security adapter validates user names, passwords, user roles, and database credentials against the information in the external system. If the external security adapter finds a match, it retrieves a generic set of user credentials (user name and password) that supply access to the database. For LDAP, the generic set of user credentials can be the same for every user, if desired. For more information on implementing an external security adapter, see *Siebel Security Guide* .

Data Transmission Security for Siebel Clients

Siebel Web Clients and Mobile Web Clients access Siebel CRM applications over the Internet and Mobile Web Clients can also synchronize with the corporate database over the Internet. To provide data transmission security for these situations, Siebel CRM supports compression and encryption of data between these clients and the Siebel Application Object Manager and Remote Manager processes. For more information about this topic, see the *Siebel Installation Guide* .

DB2 Connect also supports password encryption on the DB2 for z/OS platform. For more information about this topic, refer to the IBM documentation.

Roles and Permissions Used to Connect to DB2

The following roles and permissions are used to connect to DB2 and to install Siebel CRM on a DB2 database:

- SYSADM
- DBADM
- CREATEDBA

SYSADM Privileges Used for Connecting to DB2

A DB2 subsystem is a prerequisite for installing Siebel CRM. Although you do not need to use an ID with SYSADM privileges to install Siebel CRM, you might need such an ID to create underlying DB2 resources. For detailed information on setting up a DB2 subsystem for Siebel CRM, see *Preparing for Implementation on the DB2 Host*.

Functions that require SYSADM authority and that are necessary when you install Siebel CRM on DB2 for z/OS include:

- Allocating and accessing buffer pools
- Allocating and accessing storage groups
- Granting CREATEDBA or DBADM authority to the Siebel user ID used for the Siebel database installation
- Creating user-defined functions and stored procedures.

DBADM/CREATEDBA Privileges Used for Connecting to DB2

To run the Database Configuration Wizard requires access similar to that of DBADM; you must be able to create Siebel objects and access the necessary utilities. Therefore, it is recommended that you grant CREATEDBA privileges to the primary or secondary authorization IDs that will be used to run the Database Configuration Wizard to perform the database installation and configuration.

Granting SELECT Authority to Access the DB2 Catalog

Siebel CRM accesses the DB2 catalog to validate installation inputs. To grant appropriate users access privileges to the DB2 catalog, the system administrator must grant SELECT authority on certain catalog tables to users of the Siebel database and to users of the database installation utility, as follows:

- Siebel database users (also known as privileged users who make changes to the Siebel database) require SELECT authority for `SYSIBM.SYSTABLES`.
- Database installation users require SELECT authority for the following tables:
  - SYSIBM.SYSAUXRELS
  - SYSIBM.SYSCOLUMNS
  - SYSIBM.SYSDATABASE
  - SYSIBM.SYSINDEXES
  - SYSIBM.SYSINDEXPART
  - SYSIBM.SYSKEYS
  - SYSIBM.SYSROUTINES
  - SYSIBM.SYSSTOGROUP
  - SYSIBM.SYSTABLESPACE
  - SYSIBM.SYSTABLES
  - SYSIBM.SYSTABLEPART
  - SYSIBM.SYSTRIGGERS

  The following procedure describes how to grant SELECT authority to access the DB2 catalog.

To grant SELECT authority to access the DB2 Catalog

- Use this command:

```
GRANT AUTHORITY_TYPE ON TABLE TABLENAME TO USER;
```

  For example, to grant SELECT authority on the table SYSIBM.SYSTABLES to SSEROLE, use the following command:

```
GRANT SELECT ON TABLE SYSIBM.SYSTABLES TO SSEROLE;
```

Granting Authorization to Views in DB2

GRANT VIEW statements can fail on DB2 for z/OS if you use an external security manager, such as RACF, to protect DB2 resources, and if internal security authorizations are not also in place. Such failures might occur because when a GRANT VIEW statement is issued on DB2 for z/OS, DB2 only carries out internal database security checks before giving authorization to the view. If an internal security mechanism defining privileges to views does not exist, because these privileges are defined using an external security manager, the GRANT VIEW statement fails.

GRANT VIEW statements are issued only in the ddlview.sql file; this contains the DDL to create the Siebel Schema.

Required Authorizations

This topic lists the DB2 authorizations required to install and configure the Siebel database on DB2 for z/OS. It also lists the authorizations that are required for Siebel database accounts when implementing and using DB2 for z/OS.

DB2 Authorizations Required

The following table lists the authorizations that are necessary to implement Siebel CRM on DB2 for z/OS.

Task	Authorization Required	Task Command Example
Alter a buffer pool.	SYSADM, SYSCTRL, SYSOPR	`ALTER BUFFERPOOL (BP32K1) VPSIZE(4000);`
Grant use of a buffer pool.	SYSADM, SYSCTRL	`GRANT USE OF BUFFERPOOL BP32K1 TO PUBLIC;`
Grant CREATEIN for triggers.	SYSADM, SYSCTRL	`GRANT CREATEIN ON SCHEMA SIEBTO;`
Create a storage group.	SYSADM, SYSCTRL, CREATESG	`CREATE STOGROUP SIEBEL VOLUMES('*') VCAT SIEBEL;`
Grant use of a storage group.	SYSADM, SYSCTRL	`GRANT USE OF STOGROUP SIEBEL TO PUBLIC;`
Grant CREATEDBA and DBADM authority.	SYSADM, SYSCTRL	`GRANT CREATEDBA TO SIEBTO;`
Create a database.	SYSADM, SYSCTRL, CREATEDBA, CREATEDBC	`SET CURRENT SQLID='SIEBTO'; CREATE DATABASE SIDB0001 CCSID ASCII BUFFERPOOL BP1;`
Alter a table space.	DBADM, SYSADM, SYSCTRL	`ALTER TABLESPACE D0010002.H0677000 BUFFERPOOL BP16K0;`

Task	Authorization Required	Task Command Example
Create a table space.	SYSADM, SYSCTRL, DBADM, DBCTRL, DBMAINT, CREATETS	`SET CURRENT SQLID='SIEBTO'; CREATE TABLESPACE H1004000 IN SIDB1004 USING STOGROUP SIEBEL PRIQTY 720 SECQTY 720 BUFFERPOOL BP32K1 SEGSIZE 64 COMPRESS YES LOCKSIZE PAGE;`

Modify DB2 Connect package (if package already exists).	DBADM, SYSADM, BIND privilege on the package, ALTERIN privilege on the schema	`BIND`
		
		`c:\sqllib\bnd\@ddcsmvs.lst COLLECTION SIEBEL BLOCKING ALL DYNAMICRULES BIND OWNER SIEBTO QUALIFIER SIEBTO SQLERROR CONTINUE`

Add DB2 Connect package (if a package does not already exist).	DBADM, SYSADM, BINDADD privilege, and
	
	IMPLICIT_SCHEMA authority on the database if the schema name does not exist
	
	CREATIN privilege on the schema if the schema name of the package exists
		`BIND`
		
		`c:\sqllib\bnd\@ddcsmvs.lst COLLECTION SIEBEL BLOCKING ALL DYNAMICRULES BIND OWNER SIEBTO QUALIFIER SIEBTO SQLERROR CONTINUE`

Alter a table.	DBADM, SYSADM, SYSCTRL	`ALTER TABLE SIEBTO.S_CONTACT`
		
		`ALTER COLUMN COMMENTS SET DATA TYPE VARCHAR (100);`

Create a table.	SYSADM, SYSCTRL, DBADM, DBCTRL, DBMAINT, CREATETAB	`SET CURRENT SQLID='SIEBTO'; CREATE TABLE S_SSA_ID,(…)IN SIDB1932.H1932000;`
Alter an index.	DBADM, SYSADM, SYSCTRL	`ALTER INDEX SIEBTO.S_CONTACT_U1 BUFFERPOOL BP30;`
Create an index.	SYSADM, SYSCTRL, DBADM, DBCTRL	`SET CURRENT SQLID='SIEBTO';CREATE UNIQUE INDEX S_SSA_ID_P1 ON S_SSA_ID(ROW_ID)USING STOGROUP SIEBEL PRIQTY 720 SECQTY 720 DEFINE NO CLOSE YES PCTFREE 17 BUFFERPOOL BP2;`
Grant CREATE or PACKADM for stored procedures.	SYSADM, SYSCTRL	`GRANT CREATE ON COLLECTION SIEBINST TO SIEBDBA;`
Grant BINDADD.	SYSADM, SYSCTRL	`GRANT BINDADD TO SIEBDBA;`
Grant SELECT on catalog tables.	SYSADM, SYSCTRL	`GRANT SELECT ON SYSIBM.SYSTABLES TO SIEBTO;`
Create User-Defined Functions	SYSADM, DBADM	`CREATE FUNCTION SIEBEL.NEXTSEQ (INTEGER) RETURNS SYSIBM.INTEGER SPECIFIC SIEBEL.NEXTSEQ EXTERNAL NAME 'NEXTSEQ' LANGUAGE C PARAMETER STYLE DB2SQL NOT DETERMINISTIC FENCED RETURNS NULL ON NULL`

Task	Authorization Required	Task Command Example
		`INPUT NO SQL NO EXTERNAL ACTION SCRATCHPAD 100 NO FINAL CALL DISALLOW PARALLEL NO DBINFO WLM ENVIRONMENT DB27WLM STAY RESIDENT NO PROGRAM TYPE SUB SECURITY DB2 INHERIT SPECIAL REGISTERS`

Siebel Database Account Authorizations

Before installing and configuring the Siebel database, the DBA must create the following database accounts:

- Table owner (Siebel schema owner) account

  The table owner is the Siebel schema owner, that is, the user account assigned to the schema that owns the Siebel database objects. Privileges required for this account include DBA administration (DBADM) privileges.
- Siebel security group authorization account

  Specify a security group ID, for example, SSEROLE, for client access to the Siebel database. The *security group ID* is also referred to as the *secondary authorization ID*.
- Siebel administrator account

  The Siebel administrator account, for example, SADMIN, must be added as a member of the Siebel security group.

Note: The password assigned to the table owner and Siebel administrator accounts must not be the same as the user name specified for these accounts. To increase the security of your Siebel implementation, it is also recommended that you change the password of the Siebel administrator account regularly. For information on this task, see *Siebel Security Guide* .

The following table lists the authorizations that the database accounts created for Siebel CRM might need. Your enterprise might have unique role names that it assigns with the authorities listed in this table. Therefore, the role names in the following table are examples only.

Task	Role	Authorization Required	Task Command Example
Performing the following actions on Siebel tables: • Delete • Insert • Select • Update	Siebel group ID (for example, SSEROLE group).	Table privileges are granted automatically during the installation of the Siebel database.	`GRANT ALL ON TABLE S_SSA_ID TO SSEROLE;`
Setting the current SQL ID	Schema qualifier group or individual ID, for example, SIEBTO.	This user owns the schema objects (created by the database administrator) that are used during the installation of Siebel CRM.	`SET CURRENT SQLID='SIEBTO';`

Task	Role	Authorization Required	Task Command Example

Performing server functions, such as:
• Runstats
• Generate triggers
	Siebel administrator group, for example, SADMIN.	This user is:
		• A member of the generic user role
		• A member of the Resource Access Control Facility (RACF) group, or of another security package group, selected to act as administrator
		• A Siebel database administrator (SIEBDBA)
			`RUNSTATS TABLESPACE D0010677.H0677000 TABLE(SIEBEL.S_CONTACT)FREQVAL COUNT 10`
			
			`CREATE TRIGGER SIEBEL.PTH0477 NO CASCADE BEFORE INSERT ON REFERENCING NEW AS N FOR EACH ROW MODE DB2 SQL WHEN (N.ROW_ID IS NOT NULL)`
			`BEGIN ATOMIC`
			`SET N.PARTITION_COLUMN = RIGHT (N.ROW_ID, 2);`
			
			`END`

4 Preparing for Implementation on the DB2 Host

Preparing for Implementation on the DB2 Host

This chapter is intended primarily for the z/OS system programmer and DBA who must prepare for the Siebel Schema installation on the DB2 host. It contains the following topics:

About System Connectivity Architecture

Siebel CRM communicates with DB2 for z/OS through IBM DB2 Connect middleware. For a list of the supported versions of DB2 Connect, see the Certifications tab on My Oracle Support.

The four editions of DB2 Connect are:

- **DB2 Connect Enterprise Edition (EE).** Supports database connectivity for users running Siebel Web Client when communicating with the Siebel Server. This edition is installed on a midtier server, such as a Siebel Server computer.
- **DB2 Connect Application Server Edition (ASE).** Provides the functionality of DB2 Connect Enterprise Edition but is licensed and priced differently.
- **DB2 Connect Unlimited Edition (UE).** Provides the functionality of DB2 Connect Enterprise Edition but is priced differently.
- **DB2 Connect Personal Edition (PE).** Supports database connectivity for an individual user running Siebel Developer Web Client on a workstation. This edition is installed on a user's local workstation.

DB2 Connect EE, ASE, or UE can reside either on the same computer as the Siebel Server or on common connection gateway computers. (Do not confuse the concept of a *gateway* computer with the Siebel Gateway.) DB2 Connect EE, ASE or UE broker the connections to DB2 for multiple database clients. For information on DB2 Connect configurations, see *About Connecting to the Database Using DB2 Connect*.

Note: If you deploy DB2 Connect EE, ASE or UE with Siebel Developer Web Client, you must also install a DB2 Connect run-time component on your Siebel Developer Web Client computer. The DB2 Runtime must be installed on the Siebel Server if it does not coexist with DB2 Connect.

Siebel Servers and Siebel Developer Web Clients communicate with DB2 Connect through TCP/IP. (DB2 Connect also supports communication through SNA, but SNA currently does not support connection pooling. For this reason, it is recommended that you use TCP/IP.) For instructions on installing TCP/IP, refer to the vendor documentation on the IBM Web site.

About Connecting to the Database Using DB2 Connect

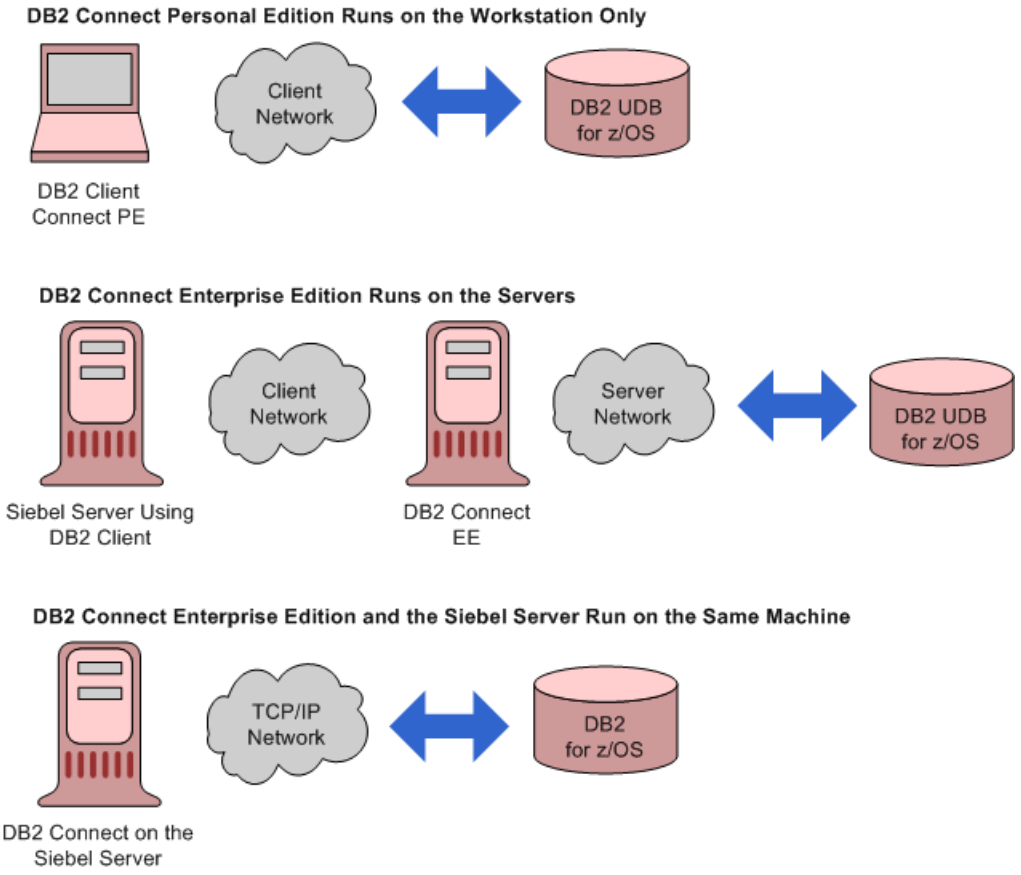
You install the Siebel Schema on DB2 for z/OS using the Database Configuration Wizard which resides on a client Siebel Server. For a description of the installation process, see *Installing the Siebel Database on the DB2 Host*. After installation, you execute SQL on the DB2 for z/OS host.

Configuration Options for DB2 Connect

When using DB2 Connect, your configuration options depend on whether you are deploying Siebel CRM on a Web Client or on a Developer Web Client, as shown in the following table.

DB2 Connect Edition	Siebel CRM Deployed on	Install and Run DB2 Connect on
Enterprise Edition (EE, ASE or UE)	Siebel Web Client	Siebel Server computer or another computer on midtier.
Enterprise Edition (EE, ASE or UE)	Siebel Developer Web Client	Any computer on midtier.
Personal Edition (PE)	Siebel Developer Web Client	Each workstation.

The following figure illustrates some of the configurations possible with DB2 Connect Personal Edition (PE) and DB2 Connect Enterprise Edition (EE). DB2 Connect PE runs on the workstation only; DB2 Connect EE can run on the same computer as the Siebel Server or on a different computer. The configuration you choose largely depends on the types of Siebel clients your enterprise supports.



For detailed information about DB2 Connect, refer to the IBM documentation.

About the Required IBM Fix Packs

After you have installed DB2 Connect, install any relevant IBM Fix Packs. To find out which IBM Fix Pack you need to use, see the Certifications tab on My Oracle Support or the vendor documentation.

Refer to IBM documentation for installing the Fix Pack for your environment.

> **Note:** When using UNIX operating systems, if you install an IBM Fix Pack after you create the DB2 database instance, you must update the instance using the db2iupdt command.

You can verify the currently installed version of DB2 Connect and the IBM Fix Pack by running the command db2level from the DB2 Command Window. In a UNIX environment, this command is run from a UNIX shell after sourcing the appropriate db2profile. Make a note of the information and compare the Informational tokens from the output against the information in the Certifications tab on My Oracle Support.

In addition to IBM Fix Pack information, the Certifications tab on My Oracle Support contains any other installation requirements.

Process of Setting Up DB2 Connect

The process of setting up DB2 Connect involves the following tasks:

1. Configure DB2 Connect to support the Web clients used in your environment:
    - *Configuring DB2 Connect EE, ASE or UE to Support Siebel Web Client*
    - *Configuring DB2 Connect EE, ASE or UE to Support Siebel Developer Web Client*
    - *Configuring DB2 Connect PE to Support Siebel Developer Web Client*
2. *Performing Postinstallation Tasks for DB2 Connect*
3. *Configuring DB2 Connect*

Configuring DB2 Connect EE, ASE or UE to Support Siebel Web Client

This task is a step in *Process of Setting Up DB2 Connect*.

Using DB2 Connect EE, ASE or UE to support the Siebel Web Client involves:

- A Siebel Server, running Application Object Managers to support the Siebel Web Client. In some cases, DB2 Connect EE is also installed on this computer.
- Optionally, an additional computer on which you install DB2 Connect EE, to act as a gateway to the database.
- DB2 for z/OS.
- Individual workstations running Siebel Web Client.

This approach is illustrated in the middle and lower parts of the previous figure.

Perform the following steps to configure DB2 Connect EE for a server computer.

To connect to DB2 using DB2 Connect EE

1. Using vendor instructions, install DB2 Connect EE on a server computer.

   The DB2 Connect EE Server computer functions as a DB2 server, with protocol support for DB2 for z/OS.

   You can also add DB2 Connect to an existing server on which DB2 is already installed.
2. On the DB2 Connect computer, upgrade DB2 Connect to the appropriate Fix Pack.

   For further information, see *About the Required IBM Fix Packs*.
3. On the DB2 Connect EE Server computer, catalog your DB2 for z/OS database, as appropriate, using the DB2 Client Configuration Assistant or the DB2 Command Line Processor.

   After installation, use a standard DB2 client to access the DB2 Connect EE Server.

Configuring DB2 Connect EE, ASE or UE to Support Siebel Developer Web Client

This task is a step in *Process of Setting Up DB2 Connect*.

Using DB2 Connect EE to support the Siebel Developer Web Client involves:

- A computer on which you install DB2 Connect EE, to act as a gateway to the database. This computer can be a Siebel Server computer or a separate computer.
- DB2 for z/OS.
- A Siebel Developer Web Client workstation.

This configuration enables a high volume of concurrent network transactions between the Siebel Developer Web Client and the DB2 Connect EE Server computer. This approach combines elements from the middle and lower configuration options shown in the following figure.

To connect to DB2 using DB2 Connect EE

1. Using vendor instructions, install DB2 Connect EE on a gateway computer.
   The DB2 Connect EE Server computer functions as a DB2 server, with protocol support for DB2 for z/OS.
2. On the DB2 Connect EE Server, upgrade DB2 Connect EE to the appropriate Fix Pack.
   For further information, see *About the Required IBM Fix Packs*.
3. On the DB2 Connect EE Server computer, catalog your DB2 for z/OS database, as appropriate, using the DB2 Client Configuration Assistant or the DB2 Command Line Processor.
   After installation, use a standard DB2 client to access the DB2 Connect EE Server.

Configuring DB2 Connect PE to Support Siebel Developer Web Client

This task is a step in *Process of Setting Up DB2 Connect*.

Using DB2 Connect PE to support the Siebel Developer Web Client involves:

- A Siebel Developer Web Client workstation on which you install DB2 Connect Personal Edition (PE), to act as a gateway to the database.
- DB2 for z/OS.

With this approach, a user on a Siebel Developer Web Client connects directly to DB2, as shown in the upper part of the previous figure.

To connect to DB2 using DB2 Connect PE

1. Install DB2 Connect PE on the Siebel Developer Web Client workstation computer.
2. On the computer, upgrade DB2 Connect PE to the appropriate Fix Pack.
   For further information, see *About the Required IBM Fix Packs*.

If you previously installed a DB2 Connect run-time client, the DB2 Connect installer upgrade adds only the functionality required for the existing client. (This is also the case if you have a DB2 server or SDK installed on your workstation.)

3. On the workstation computer, catalog your z/OS databases as appropriate, using the DB2 Client Configuration Assistant or the DB2 Command Line Processor.
   After installation, use a standard DB2 client to access DB2 Connect.

Performing Postinstallation Tasks for DB2 Connect

This task is a step in *Process of Setting Up DB2 Connect*.

After installing DB2 Connect, applying the necessary IBM Fix Pack, and setting up the network connectivity (TCP/IP and FTP) from the DB2 Connect server to the mainframe, there are a number of additional steps you must perform to use DB2 Connect with Siebel CRM. These steps include:

- *Defining a Database Alias and Testing a Connection*
- *Binding the DB2 Connect Packages*

Defining a Database Alias and Testing a Connection

This topic describes how to define a database alias and test a connection.

When you create a connection to DB2 on z/OS, the connection is defined with a database alias that has the appropriate connection information for your database. This alias is the connect string that you specify when you install the Siebel Server. Record the connect string in the copy you made of *Deployment Planning Worksheet*.

To define your database alias, use the DB2 Configuration Assistant or the DB2 Command Line Processor. For information on setting values in the db2cli.ini file and for using the DB2 Configuration Assistant or Command Line Processor, refer to the relevant IBM documentation.

> **Note:** When you define database aliases, you must add the `TXNISOLATION` parameter to the database alias entry in the db2cli.ini file, and set it to have a value of 1.

Testing the Database Connection in a Windows Environment

Use the following procedure to test the database connection in a Windows environment.

To test the database connection on the Windows platform

1. To ensure that you can connect to the database alias that you have defined, you can use either the DB2 Configuration Assistant, or the DB2 Command Window or Command Line Processor.
   To use the DB2 Configuration Assistant method, start the utility, highlight the database alias that you created, and right-click.
2. Select the Test Connection option.
3. Enter an authorized user ID and password, and select ODBC and CLI as the test parameters.
4. Click Test Connection.
   The following message indicates a successful connection:

```
CLI connection tested successfully.
ODBC connection tested successfully.
```

Alternatively, you can use the DB2 Command Window or the DB2 Command Line Processor to test the database connection by typing the following, then pressing Enter:

```
DB2 connect to database_alias user database_userID using database_password
```

where:

- database_aliasis the database alias you created using DB2
- database_userID and database_password are an authorized user ID and its associated password on the DB2 host.

If the connection is successful, database connection information appears.

Testing the Database Connection in a UNIX Environment

Use the following procedure to test the database connection in a UNIX environment.

To test the database connection on the UNIX platform

1. Open a UNIX shell.
2. Go to the SIEBSRVR_ROOT directory, that is, the directory in which the Siebel Server is installed.
3. Enter the following command:

```
DB2 connect to database_alias user database_userID using database_password
```

where:

- database_alias is the database alias you created using DB2
- database_userID and database_password are an authorized user ID and its associated password on the DB2 host.

If your connection is valid, database connection information appears.

Binding the DB2 Connect Packages

To enable ODBC to point to DB2, bind the CLI/ODBC Support packages. DB2 Connect is installed with bnd files that bind to the host server. You can bind these packages using the default collection ID, or you can create a new collection ID for these packages. The Bind options are as listed:

- Use DB2 Configuration Assistant's BIND option to bind the packages. Highlight the dbalias, right-click, and choose the BIND option from the pop-up dialog.
- Issue explicit BIND commands. Refer to the relevant IBM documentation for further details on the BIND command.

Make sure that the authid has sufficient authority to bind these packages (BINDADD).

Configuring DB2 Connect

This task is a step in *Process of Setting Up DB2 Connect*.

DB2 Connect configuration parameters allow you to specify some of the ways DB2 Connect works, for example, pooling and thread reuse. These configuration changes vary depending upon site-specific details and whether the Enterprise or Personal version of DB2 Connect is installed. Changes to these parameters change the database manager configuration file. For information on changing the database manager configuration parameters, refer to the appropriate IBM documentation.

DB2 Connect configuration tasks include the following:

- *Setting the DB2CONNECT_IN_APP_PROCESS Environment Variable*
- *Setting the DB2CONNECT_ENABLE_EURO_CODEPAGE Environment Variable*
- *Setting DB2 Connect EE Configuration Options*
- *Tuning DB2 Connect by Increasing the I/O Block Size*

Setting the DB2CONNECT_IN_APP_PROCESS Environment Variable

The DB2 Connect system environment variable, DB2CONNECT_IN_APP_PROCESS, determines whether or not DB2 Connect clients running on a DB2 Connect EE server must run within an agent. When this variable is set to NO, local clients have to run within an agent. The default value for the DB2CONNECT_IN_APP_PROCESS variable is YES.

You can use DB2 Connect simply as a means of allowing a Siebel application to connect to the Siebel database on the z/OS host. Alternatively, you can implement the following features of z/OS provided by DB2 Connect with Siebel CRM:

- Connection load balancing across Sysplex members
- Connection pooling
- Connection concentration

To enable these features when DB2 Connect is located with your Siebel CRM installation, you must set the DB2 environment variable, DB2CONNECT_IN_APP_PROCESS, to NO.

For additional information on DB2 Connect features and the DB2CONNECT_IN_APP_PROCESS variable, refer to the relevant IBM documentation. For additional information on connection concentration, see *Setting DB2 Connect EE Configuration Options*. For additional information on server clustering for the Siebel database, see *Siebel Deployment Planning Guide* .

Setting the DB2CONNECT_ENABLE_EURO_CODEPAGE Environment Variable

The DB2 Connect system environment variable, DB2CONNECT_ENABLE_EURO_CODEPAGE, determines whether or not support for the euro symbol is implemented. The default value of this variable is NO. To ensure the euro symbol is implemented correctly on the z/OS host, set this variable to YES.

When you set the DB2CONNECT_ENABLE_EURO_CODEPAGE parameter to YES, the Unicode code page on the DB2 Connect client computer is mapped to a comparable CCSID which includes support for the euro symbol. It is recommended that you set the value of this variable to YES on all DB2 Connect computers used to connect to the Siebel database on the z/OS host. For additional information on the DB2CONNECT_ENABLE_EURO_CODEPAGE environment variable, see the relevant IBM documentation.

Setting DB2 Connect EE Configuration Options

The DB2 Connect EE product is a server version that allows multiple users to connect to DB2 on the z/OS host. Because it is a server, you must complete additional configuration steps:

- You must specify whether or not to enable connection concentration.

  For information on connection concentration, see *About Connection Concentration*.
- You must define the maximum number of concurrent users.

  For information on specifying the number of concurrent users allowed, see *Setting the MAXAGENTS Parameter Value*.

About Connection Concentration

Connection concentration provides additional workload balancing opportunities for DB2 Data Sharing by allowing connections to be rebalanced across DB2 Data Sharing members at the commit phase, instead of only at the time of initial connection. This dynamic rebalancing of connections across members can be beneficial after a planned or unplanned outage to DB2.

Note: Connection concentration is disabled by default during the installation of DB2 Connect.

The default settings for the configuration parameters that control connection concentration are:

- Priority of agents (AGENTPRI) is SYSTEM.
- Maximum number of existing agents (MAXAGENTS) is 100. MAXAGENTS is the maximum number of worker agents. This value represents the maximum number of concurrent connections to DB2 for z/OS.
- Agent pool size (NUM_POOLAGENTS) is 50 (calculated). NUM_POOLAGENTS is the maximum number of idle pool agents. This value represents the number of concurrent connections to DB2 for z/OS. Setting this parameter to 0 disables connection pooling.
- Initial number of agents in pool (NUM_INITAGENTS) is 0.
- Maximum number of coordinating agents (MAX_COORDAGENTS) is equal to (MAXAGENTS minus NUM_INITAGENTS).
- Maximum number of concurrent coordinating agents (MAXCAGENTS) is equal to MAX_COORDAGENTS.
- Maximum number of client connections (MAX_CONNECTIONS) is equal to the value of MAX_COORDAGENTS. When MAX_CONNECTIONS is equal to MAX_COORDAGENTS, Connection Concentration is not enabled.

Enabling Connection Concentration

The following procedure describes how to enable connection concentration.

To enable Connection Concentration

- Set the value of the maximum number of client connections so that MAX_CONNECTIONS is greater than MAX_COORDAGENTS (MAX_CONNECTIONS equals MAX_COORDAGENTS plus 1).

  Setting MAX_CONNECTIONS to equal MAX_COORDAGENTS plus1 enables the Connection Concentration dynamic rebalancing capabilities and still allows the transaction pooling to be handled by Siebel pooling.

  > **Note:** If you enable Connection Concentration, you must first set the value for the maximum number of agents (MAXAGENTS) parameter; this value determines the value of other parameters. Set the value of the MAXAGENTS parameter as appropriate for your site; this value varies according to the memory available and the number of server connections expected.

Setting the MAXAGENTS Parameter Value

If you do not want to enable connection concentration, and if you have not made any adjustments to the connection concentration parameters, then the only configuration to do is to set the number of MAXAGENTS to the number of concurrent connections to DB2 that you expect will be required. To set this number, issue an `update dbm cfg using` statement. For more information on updating the database manager configuration, see the vendor documentation on the IBM Web site.

If you are using Siebel connection pooling (and it is recommended that you do), the value you choose for MAXAGENTS determines the value you specify for the MaxSharedDbConns parameter value. Further, it determines the value you specify for the MAXDBATS parameter on the z/OS host. For more information about this feature, see *Database Connection Pooling*.

Tuning DB2 Connect by Increasing the I/O Block Size

The RQRIOBLK database manager configuration parameter specifies the maximum size of the client I/O blocks used to store the results of queries sent by a database client to a remote database.

To ensure that database queries which return large blocks of data do not cause DB2 Connect users to experience long response times, you can change the value of the client RQRIOBLK database manager configuration parameter.

The default value for the RQRIOBLK parameter on DB2 Connect is 32 KB and the maximum size is 65 KB. DB2 for z/OS can support up to 10MB cursor blocks so you can increase the value of the RQRIOBLK parameter to 65 KB and so improve performance.

To change the DB2 Connect I/O block size (RQRIOBLK) value

- Using the Command Line Processor, change the RQRIOBLK size by typing the following:

```
DB2 UPDATE DBM CFG USING RQRIOBLK 65535
```

About Setting Up the DB2 Subsystem

In setting up the DB2 subsystem in preparation for deploying a Siebel application on DB2 for z/OS, you must perform a number of tasks, and consider a number of factors, including the following:

- *Advantages of Using a Separate DB2 Subsystem*
- *About Unicode Character Conversions on z/OS*
- *Considerations in Choosing the Database CCSID*
- *About Data Distribution Facility and Workload Manager*
- *DSNZPARM Parameter Settings for Siebel CRM*

Advantages of Using a Separate DB2 Subsystem

Production deployments of Siebel CRM can be in both separate and shared DB2 subsystems. However, setting up a separate subsystem for Siebel CRM is preferable, particularly for larger deployments, for the following reasons:

- **DSNZPARM optimization.** The DSNZPARMs used with Siebel CRM are optimized for the Siebel CRM applications, but might not be optimal for use with other applications. It is recommended, but not required, that you run Siebel CRM applications on its own subsystem.
- **OLTP and OLAP characteristics.** Siebel CRM possesses the characteristics of both OLTP and OLAP applications. However, most other vendors' applications have the characteristics of either one or the other.
- **System catalog locking during Siebel database recovery.** Siebel CRM defines one table space for each database so system catalog locking during Siebel database recovery is not a problem. However, if you use database layouts with multiple table spaces for each database, recovery of the Siebel Schema might affect other applications on a shared DB2 subsystem if the system catalog becomes locked during the recovery process. Because recovery typically requires the restoration of all Siebel table spaces, locking could last many hours.

Even though a separate DB2 subsystem is preferable for implementing Siebel CRM, there are numerous successful Siebel CRM deployments in shared DB2 subsystems.

About Unicode Character Conversions on z/OS

Unicode provides an industry-standard, universal, character-encoding standard. Siebel CRM supports the Unicode character set. On the z/OS platform, Siebel CRM supports the Unicode character set on all versions of DB2 for z/OS, New Function Mode and later releases.

On DB2 for z/OS, character conversions from Unicode code pages to ASCII and EBCDIC code pages are performed by the z/OS Unicode Services; these conversions are required if Siebel CRM applications are to function correctly. For information on setting up the z/OS Unicode Services, see the IBM document z/OS Support for Unicode: Using Unicode Services on the IBM z/OS information center Web site at

http://publib.boulder.ibm.com/infocenter/dzichelp/v2r2/index.jsp

Considerations in Choosing the Database CCSID

The current Siebel CRM release supports ASCII- and EBCDIC-based coded character set IDs (CCSIDs) for all versions of DB2 for z/OS, and Unicode character set IDs on all versions of DB2 for z/OS, New Function Mode and later releases. The database CCSID you use determines a number of factors, for example, the sort order used in list applets.

Review *Siebel CRM Update Guide and Release Notes* on My Oracle Support for information about known restrictions; for example, the following features are not supported on databases with EBCDIC code pages:

- Web Client migration
- Siebel Data Warehouse
- Siebel Dun and Bradstreet server components

You must run development databases with ASCII or Unicode code pages, because databases with EBCDIC code pages do not support the following procedures in a development environment upgrade:

- Merging prior configuration changes into a new custom configuration repository (for upgrades only)
- Publishing a new Siebel Runtime Repository

You can use an ASCII-, an EBCDIC-, or a Unicode-based CCSID to create your Siebel Schema, but, if you mainly use single-byte character sets, it is recommended that you use ASCII or EBCDIC CCSIDs to minimize storage space requirements. Review the following recommendations:

- Use an ASCII-based encoding scheme to reduce the overhead required for character conversion. Character conversion is performed by Distributed Relational Database Architecture (DRDA), between the database and the midtier ASCII servers.
- Use a Unicode-based encoding scheme if you require a character set that includes many characters that do not fit well in the ASCII or EBCDIC character set, for example, double-byte character sets for Chinese, Japanese, and Korean characters.

You can partition table spaces on a database with either an ASCII, an EBCDIC or a Unicode code page, so that limit keys reflect the sort sequence difference between ASCII, EBCDIC and Unicode. For more information, see *Configuring the Siebel Database Layout*.

About Setting the CCSID

Configure the CCSIDs on the DB2 installation panel, DSNTIPF. The APPLICATION ENCODING field allows you to specify the default application encoding scheme. The value of the DEF ENCODING SCHEME field on the same panel determines whether the default format in which data is stored on DB2 is ASCII, EBCDIC, or Unicode.

> **Note:** Set the value of the MIXED DATA field on the DSNTIPF panel to NO for all supported languages except Japanese.

> **Note:** The EBCDIC CCSID of the DB2 subsystem containing the Siebel schema must be identical to the z/OS system CCSID. Otherwise data corruption may occur during various data loading processes.

About Data Distribution Facility and Workload Manager

The Data Distribution Facility (DDF) is a component of DB2 that provides a connection between your DB2 subsystem and remote clients and systems. Specifically, it communicates with DB2 Connect for remote application access. You must configure and activate DDF before you install Siebel CRM. DDF setup is described in the vendor documentation on the IBM Web site.

Workload Manager (WLM) is an integrated component of z/OS that manages system resources across z/OS subsystems. You must configure WLM for use with DDF threads and Siebel application stored procedures and user-defined functions (UDFs).

In relation to DB2, WLM manages incoming requests for DB2 stored procedures and UDFs, and allocates resources to process them. If you do not have WLM installed and configured, you cannot use Siebel components that require stored procedures and UDFs. Stored procedures are used with EIM (when UPDATE STATISTICS is set to TRUE) and the Siebel Upgrade application component. UDFs are used with EIM Export and currency aggregations. For information on setting up WLM, see the vendor documentation on the IBM Web site.

DSNZPARM Parameter Settings for Siebel CRM

To run your Siebel CRM applications, you must set the values of some of the DSNZPARM parameters to required settings; other DSNZPARM parameter values are recommended to improve the performance of your Siebel CRM applications. This topic describes both the recommended and the required DSNZPARM parameter settings.

> **Note:** Use the IBM default values for any DSNZPARMs for which recommended or required values are not listed in this topic. The IBM default values for any DSNZPARMs not listed here are the most appropriate values for running the current release of Siebel CRM.

You can configure some DSNZPARM parameters online, but to configure other parameters, you must shut down DB2. For information on the parameters that can be updated online, refer to the vendor documentation on the IBM Web site.

The following table lists the required and recommended DSNZPARM parameter settings.

Parameter	Explanation	Setting	Required or Recommended
DSN6SPRM			
`CACHEDYN`	Turns on dynamic statement caching.	`YES`	Required
`CDSSRDEF`	Turns off parallelism for dynamic statements.	`1`	Required
`CONTSTOR`	Compresses storage on a regular basis. Set this value to `YES` if DBM1 storage is an issue.	`YES`	Recommended
`EVALUNC`	Allows predicate evaluation on uncommitted data.	`YES`	Recommended

Parameter	Explanation	Setting	Required or Recommended
MAXKEEPD	The total number of prepared, dynamic SQL statements that can be saved past a commit point by applications that run with the KEEPDYNAMIC(YES) bind option.	**0**	Recommended
NPGTHRSH	Allows small tables to use indexes.	**10**	Recommended
NUMLKUS	Number of locks for each user. It is recommended that the DBA monitors and sets this value. If you experience persistent locking problems, consider setting the parameter to **0** (unlimited number of locks). During Incremental Repository Merge, Oracle recommends setting the parameter to 0 (unlimited number of locks). For more information about Incremental Repository Merge, see *Siebel Database Upgrade Guide* .	**100,000**	Recommended
PARTKEYU	Allows update of partitioning keys.	**YES**	Required if using partitioning
RETVLCFK	Allows index-only access of varying-length characters. Set this value to **NO** due to padding.	**NO**	Required
DSN6SYSP			
CHKFREQ	Avoids frequent checkpoints in a high-update environment. It is recommended that DBAs monitor and set this value for between 10 and 20 minutes.	**500,000**	Recommended
CONDBAT	Maximum number of concurrent remote connections. It is recommended that the DBA monitors and set this value.	**10,000**	Recommended
EXTSEC	Allows DB2 Connect to receive more complete error messages. Allows you to change passwords from DB2 Connect.	**YES**	Required
LOBVALA	Specifies the maximum amount of storage, in KB, assigned to each user for storing large object (LOB) values in the subsystem. If the value specified for this parameter is too low, the job that imports seed data during the Siebel database installation process can fail.	**48,000**	Recommended
LOBVALS	Specifies the maximum amount of memory, in MB, assigned to each subsystem for storing LOB values.	**24,000**	Recommended
MAXDBAT	Maximum number of database threads (DBAT). It is recommended that the DBA monitors and set this value.	**500**	Recommended

Parameter	Explanation	Setting	Required or Recommended
DSN6FAC			
CMTSTAT	Allows a greater number of remote threads without affecting storage. Enables DDF thread pooling.	INACTIVE	Recommended
IDTHTOIN	Number of seconds before an idle thread is canceled. Prevents long-running threads from holding resources. The DBA should monitor and set this value.	600	Recommended
POOLINAC	Number of seconds an inactive thread remains in the DDF pool. The DBA should monitor and set this value.	120	Recommended

PADIX Considerations

The PADIX parameter determines whether or not new indexes are padded by default. If you specify YES, a new index is padded unless the NOT PADDED option is specified on the CREATE INDEX statement. The IBM default value for this parameter is YES. If you do not want new indexes to be padded by default, you must specify NO for this parameter.

In some Siebel implementations, setting the PADIX parameter to NO has been found to lead to performance improvements and savings in disk space; it also allows for index-only access to data. If you do set PADIX to NO, ensure that you carefully monitor the impact of doing so in your environment.

NUMLKUS Considerations

If a resource-unavailable error occurs, because the value set for the NUMLKUS parameter has been exceeded while performing a Siebel operation, increase the NUMLKUS parameter value.

> **Note:** The NUMLKUS parameter value is important when running large EIM batches and during the use of Siebel Remote for the initial database extract. If this value is too small, EIM runs or the database extract can fail.

If EIM fails because the value specified for the NUMLKUS parameter has been exceeded, take one of the following actions:

- Reduce the size of the batch.
- Increase the value of NUMLKUS.

IDTHTOIN Considerations

The IDTHTOIN DSNZPARM recommended setting is 600. (The parameter is specified in seconds.) Monitor this value in your installation to ensure that it is set properly. The IDTHTOIN parameter time- outs active threads that have been idle (no network traffic) for more than 600 seconds. (This time is the recommended value; your value might differ.)

The events that can occur when an active thread times out include:

- The thread is not returned to the DB2 Connect thread pool for reuse.
- When the client or component tries to reuse the DB2 Connect thread, the thread is not available. The client or component has to issue a full Connect to continue processing the work.
- If the component or client cannot issue a full Connect, the client receives a SQL30081N SQLCODE. An example of an error message you might receive if the component or client cannot reconnect is:

```
[IBM][CLI Driver][DB2] SQL30081N A communication error has been detected.
```

```
Communication protocol being used: "TCP/IP". Communication API being used:
"SOCKETS". Location where the error was detected: "". Communication function
detecting the error: "recv". Protocol specific error code(s): "*", "*", "0".
SQLSTATE=08001
```

Another consideration to bear in mind when specifying a value for the IDTHTOIN parameter is that connection pooling on the ODBC driver must be turned off. Using connection pooling with the driver is not certified with Siebel CRM version 8 releases.

Estimating the Storage Space Required

The space needed by DB2 varies, depending on the total number and types of users supported. Consult the IBM DB2 for z/OS technical documentation for more information on these requirements.

The minimum DB2 storage space required for a typical Innovation Pack 2017 installation on DB2 for z/OS (installation of repository tables and seed data but with no user data loaded into the base tables) is between 5 GB and 20 GB, depending on the DEFINE parameter setting specified when the database objects were created.

If database objects are created with the DEFINE parameter set to YES, a DB2 catalog entry is created for the object and a data set is allocated for the object. If database objects are created with the DEFINE parameter set to NO, only a DB2 catalog entry is created for the object, so less space is required. Siebel CRM provides all table space and index schema definitions with the DEFINE NO option as the default. Therefore, no physical data sets are created until data is actually inserted into the table space.

> **Note:** DEFINE NO is only applicable to DB2-managed (STOGROUP) data sets and is ignored for user-managed (VCAT) data sets and for LOB table spaces.

The minimum space you require varies depending on the Siebel functionality you implement and the amount and nature of data supporting it. The process for making accurate database size calculations is complex, involving many variables. Use the following guidelines to assist you:

- Determine the total number and types of users of Siebel CRM applications (for example, 500 sales representatives and 75 sales managers).
- Determine the Siebel functionality that you will implement and the entities required to support them. Typically, the largest entities are:
  - Accounts
  - Activities
  - Contacts
  - Forecasts
  - Opportunities
  - Service Requests
- Estimate the average number of entities for each user (for example, 100 accounts for each sales representative) and calculate an estimated total number of records for each entity for your total user base.
- Calculate the average record size for each entity and multiply by the total number of records using standard sizing procedures for your specific database and the Siebel Schema definition. Typically, these entities span

multiple physical tables, all of which must be included in the row size calculation. This calculation determines the estimated data sizes for the largest entities.

- Add extra space for the storage of other Siebel data. A rough guideline for this amount is one-half the storage required.
- Allow for a margin of error in your total size calculation.
- Factor growth rates into your total size calculation.
- A Siebel database that uses a Unicode encoding format will generally require more space than one that uses an ASCII or EBCDIC encoding format. For further information on the space requirements for Siebel Unicode databases, see *About Migrating a Siebel Database to Unicode Format*.

Note: If you carry out a repository import and it fails, additional storage space is also required.

Allocating Space for Buffer Pools and Storage Groups

The following example illustrates how you can alter space for buffer pools, in preparation for installing the Siebel Schema, using a group ID.

```
ALTER BUFFERPOOL (BP32K1) VPSIZE (4000)
GRANT USE OF BUFFERPOOL BP32K1 TO PUBLIC;
```

The following example illustrates how you can allocate space and access to storage groups using a group ID.

```
CREATE STOGROUP SIEBEL VOLUMES ('*') VCAT SIEBEL;
GRANT USE OF STOGROUP SIEBEL TO PUBLIC;
```

Estimating the Number of Database Objects You Need

Oracle ships over 4,000 tables and 21,000 indexes. The number of objects created in the Siebel Schema is determined by the Siebel product line you purchase. The number of objects determines how much space you must allocate. It is up to you to determine which of the objects shipped might actually be required for your deployment, based on your business needs.

CAUTION: You must not remove unused tables from the Siebel database; doing so causes Oracle to discontinue technical support of your deployment. However, unused, nonunique, Siebel indexes can be removed from the database in a z/OS Siebel deployment. To deactivate indexes, contact your Oracle sales representative for Oracle Advanced Customer Services to request assistance from Oracle's Application Expert Services.

5 Configuring the Siebel Database Layout

Configuring the Siebel Database Layout

This chapter describes the storage control file templates that are provided with Siebel CRM for database schema configuration during the installation process. The primary audience for this chapter is the DB2 DBA.

This chapter contains the following topics:

Control Files Used in the Siebel CRM Installation

The Siebel Database Schema installation process uses two types of Siebel control files:

- **DDL control file**

  The DDL control file contains the logical definitions for all Siebel tables and indexes for the current version of Siebel CRM. The DDL control file (ddl.ctl or schema.ddl) is read-only to protect the integrity of the Siebel data model.

- **Storage control file**

  The storage control file contains the physical database layout. The storage control file is specific to DB2 for z/OS.

  Carefully review the storage control file (or the resulting schema.sql file that is based on the storage control file) and adjust the database layout to satisfy your database requirements. You can modify any option available for each database object, for example, by fine-tuning FREEPAGE and PCTFREE for a particular table space or a group of table spaces.

  > **Note:** It is especially important that you review SEGSIZE, PRIQTY, and SECQTY for the table spaces that are to contain the major tables for your implementation. The table spaces used for Siebel Repository tables are appropriately sized. However, because sizing requirements for base Siebel tables vary from deployment to deployment, it is your responsibility to set storage parameters capable of accommodating your installation.

Together, the DDL control file and the storage control file provide Siebel CRM with all the information necessary to create DDL for the Siebel Schema in an output file (by default named schema.sql).

You can choose either of two methods to execute the SQL files from a designated Siebel Server:

- Automatic execution from the Siebel Server using an ODBC utility (such as odbcsql) or a version control tool for file execution. To use this method, review *Configuration Options for DB2 Connect* to determine the configuration appropriate to the Client your enterprise supports, and then follow the instructions in *About the Required IBM Fix Packs*.
- Direct execution on the DB2 host using Siebel-provided scripts, your own FTP, or any other file transfer program to transfer files to the DB2 host, and then use a native IBM utility, such as SPUFI for file execution.

  Note: If you want to execute SQL using mainframe native tools, refer to the IBM documentation.

About Storage Control File Templates

The storage control file is unique to the DB2 for z/OS platform for Siebel CRM. The storage control file contains storage information (for example, partitioning indexes, table spaces, and storage groups) that is used as the basis for the storage layout of your Siebel database. Even if you are using a preconfigured storage layout, make sure that the layout is valid for your Siebel Schema.

The Siebel database schema structure implements a 1:1:1 model (one table in each table space, and one table space in each database). This layout reduces logging, removes concurrency issues, and allows for more databases to be used to achieve high database parallelism. While this is the most appropriate schema structure to avoid database descriptor (DBD) locking and logging, you can choose to regroup tables in table spaces and table spaces in databases. However, if you do implement a structure other than the 1:1:1 model (for example, a 1:M:1 schema layout), ensure that you carefully monitor the impact of doing so in your environment. If you notice a decrease in the performance of Siebel CRM, this could be due to lock contention occurring as a result of SQL executed by Siebel CRM applications or Siebel utilities. In this case, you might have to make adjustments to the schema layout.

The Siebel application installation process installs five storage control file templates in the DBSRVR_ROOT `\db2390` directory (Windows) or the DBSRVR_ROOT `/db2390` directory (UNIX). Select one of the following templates, based on your partitioning needs and your encoding scheme:

- **storage_np.ctl**

  This template contains a database storage layout for a non-Unicode Siebel Schema with one table for each table space, and one table space for each database. No partitioned table spaces are provided.
- **storage_np_u.ctl**

  This template contains a database storage layout for a Unicode Siebel Schema with one table for each table space, and one table space for each database. No partitioned table spaces are provided.
- **storage_p.ctl**

  This template is the database storage layout for the Siebel Schema with partitioning for ASCII encoding. Every nonpartitioned table resides in its own segmented table space. Each table space resides in its own database.
- **storage_p_u.ctl**

  This template is the database storage layout for the Siebel Schema with partitioning for Unicode encoding. Every nonpartitioned table resides in its own segmented table space. Each table space resides in its own database.
- **storage_p_e.ctl**

This template is the database storage layout for the Siebel Schema with partitioning for EBCDIC encoding. Every nonpartitioned table resides in its own segmented table space. Each table space resides in its own database.

The layouts of the storage control file templates are generic; your DBA must customize them to suit the needs of your enterprise, for example, to change the default space allocation for Siebel objects, seed data, and views. Although Siebel-provided templates might be sufficient for your development environment without any changes, it is recommended that your DBA reviews and modifies the database storage layout to best fit the needs of your deployment in a production environment.

The storage control file templates are the starting point for your customization process. Before you select a Siebel storage control file template, you need to understand how Siebel objects are grouped and the object naming conventions used.

About Siebel Objects

An understanding of Siebel objects and object naming conventions is a foundation for understanding how your customizations affect the underlying data sets. The following are described in this topic:

- *Symbolic Variables in Storage Control Files*
- *Default Objects in Storage Control Files*
- *Buffer Pools Used in Storage Control Files*
- *About Using Storage Groups*
- *Database Objects in Storage Control Files*
- *Table Objects in Storage Control Files*
- *Table Space Objects in Storage Control Files*
- *Object Naming Conventions*

Symbolic Variables in Storage Control Files

The storage control file contains definitions of objects (such as databases, table spaces, and tables) that define the physical storage layout of your database. Many of the options used in the object definitions are the equivalent of options in DB2 SQL statements.

While most options within the objects are defined by actual values (for example, SegSize is set to 32), some options include symbolic variables that are substituted with the actual values that comply with your organization's standards. Review the following table for a list of the symbolic variables in the storage control file templates and the values substituted for them.

Symbolic Variable	Actual Value Substituted for Symbolic Variable
$DbnamePrefix	Database Name Prefix
$StogroupTables	Table Storage Group for Tables
$StogroupIndexes	Index Storage Group for Indexes

Symbolic Variable	Actual Value Substituted for Symbolic Variable
$IndexBufferPool	Index Buffer Pool Name
$4KBufferPool	4-KB Buffer Pool Name
$8KBufferPool	8-KB Buffer Pool Name
$16KBufferPool	16-KB Buffer Pool Name
$32KBufferPool	32-KB Buffer Pool Name
$DbType	Encoding Scheme (acceptable values are either ASCII or EBCDIC)

Default Objects in Storage Control Files

The first object in the storage control file is a Defaults object. If no actual value exists for a given object, but a value for that object is required by a DB2 statement, then the default value is either derived from the Defaults object or inherited from a higher-level object.

The following example illustrates what the Defaults object looks like in the storage control file:

```
[Object 1]
Type = Defaults
Name = Defaults
Database = $DbnamePrefix0000
Tablespace = SIEBTS00
Stogroup = $StogroupTables
IndexStogroup = $StogroupIndexes
IndexBp = $IndexBufferPool
Bufferpool = $4KBufferPool
LockSize = Page
SegSize = 32
LockMax = 0
PriQty = 48
SecQty = 1440
PctFree = 17
FreePage = 0
Compress = Yes
Define = No
Erase = No
CCSID = $DbType
```

You can override default values for any given object by explicitly defining the value in the object. For example, if the Tablespace object does not contain PriQty or SecQty values, then these values are inherited from the PriQty and SecQty values defined in the Defaults object. However, if the values are defined in the Tablespace object, then the explicitly provided value overrides the default and is used in the output DDL.

Note: If you create an extension table in Siebel Tools and use the apply process without creating new underlying objects for this table in the storage control file, then the necessary storage layout is derived from the default definitions. For information about the apply process, see *Roadmap for Creating Custom Extensions to the Siebel Schema*.

Buffer Pools Used in Storage Control Files

Siebel CRM uses four buffer pools in the storage control file templates:

- One 4-KB buffer pool for 4-KB table spaces (the default name is BP1)
- One 4-KB buffer pool for all indexes (the default name is BP2)
- One 8-KB buffer pool for 8-KB table spaces (the default name is BP8K1)
- One 16-KB buffer pool for 16-KB table spaces (the default name is BP16K1)
- One 32-KB buffer pool for 32-KB table spaces (the default name is BP32K1)

Multiple buffer pools can be used for each page size. If you require multiple buffer pools, use the Siebel Database Storage Configurator (dbconf.xls) or any other method described in *Preparing a Storage Control File* to enter these values.

Your DBA must activate and grant use of buffer pools before you can perform the Siebel installation. For an example of buffer pool definitions, see *Allocating Space for Buffer Pools and Storage Groups*.

About Using Storage Groups

Siebel CRM requires one storage group for indexes and one storage group for tables (data). Before performing your installation, ask your DBA to provide you with one storage group name for each. You can use the same name for data and indexes, or you can use different names. You can set up storage group usage in any way that makes sense for your deployment. For an example of storage group definitions, see *Allocating Space for Buffer Pools and Storage Groups*.

Database Objects in Storage Control Files

In the Siebel database storage layout, each table resides in its own table space, and each table space resides in its own database. The number of databases you receive depends on which modules your organization purchases. You can regroup table spaces in databases, as required for your enterprise.

Note: The maximum number of databases is 64,000 for each DB2 subsystem.

The following example illustrates what a Database object looks like in the storage control file:

```
[Object 4]
Type = Database
Name = SIDB2532
LockSize = Page
```

The Database object illustrated in this topic translates into the following DDL output:

```
CREATE DATABASE SIDB2532 CCSID ASCII BUFFERPOOL BP1 /
```

In Siebel CRM, CCSID is defined on the database level only. CCSID is always taken from the Defaults object.

LockSize is not used in the CREATE DATABASE statement, because LockSize is a default value for all table spaces that belong to a given database.

The names of databases in the storage control file comprise two parts:

- The first part of the name is a variable, `$DbnamePrefix` .

  The default value of the database name prefix is SIDB. During the installation process, you can substitute the default database name prefix with a literal value up to four characters long that conforms to your organization's naming convention.

  > **Note:** The database name prefix must be the same for all database objects in the Siebel schema because the prefix identifies an object as belonging to the Siebel schema. Siebel utilities can recognize and use Siebel objects only if they follow Siebel naming conventions.

- The second part of the name is a value, generally a four-digit number, that is generated by Siebel CRM and that must not be changed. (A number of databases that contain temporary tables are not assigned database names ending in numbers.) Where a four-digit number is assigned, this number in the database name is the same four-digit number used in the table space name that resides in this database. For more information on the table space name structure, see *Table Space Naming Conventions*.

Table Objects in Storage Control Files

The following example illustrates what the Table object looks like in the storage control file:

```
[Object 8005]
Type = Table
Name = S_ORG_EXT
Database = SIDB1465
Tablespace = H1465000
Clobs = No
```

The Table object illustrated in this topic translates into the following DDL output:

```
CREATE TABLE SIEBTO.S_ORG_EXT (
ROW_ID VARCHAR(15) NOT NULL,
CREATED TIMESTAMP DEFAULT NOT NULL,
CREATED_BY VARCHAR(15) NOT NULL,
...

VAT_REGN_NUM VARCHAR(30),
DIRECTIONS LONG VARCHAR) IN SIEBTO.H1465000 /
```

Siebel CRM includes one template for each encoding scheme when partitioning is used and one template for each encoding scheme for nonpartitioning.

Most tables use page-level locking by default, but a few use row-level locking. The following tables use row-level locking:

- `S_ESCL_ACTN_REQ`
- `S_ESCL_LOG`
- `S_ESCL_REQ`

- S_DOCK_TXN_LOG
- S_DOCK_TXN_LOGT
- S_DOCK_TXN_SET
- S_DOCK_TXN_SETT

The following tables use table-level locking:

- S_DOCK_INIT_ITEM
- S_DOCK_INITM_XX

Note: LockSize is not defined on the table level but for the corresponding table space.

Table Space Objects in Storage Control Files

Siebel tables are created in one of four table space sizes, 4 KB, 8KB, 16 KB, and 32 KB. Most tables are defined within a 4-KB table space.

Note: You can create new table spaces, and you can regroup tables as required for your enterprise.

The following example illustrates what the segmented Tablespace object for the Siebel Repository looks like in the storage control file.

```
[Object 4340]
Type = Tablespace
Name = H1004000
Database = SIDB1004
LockSize = Page
Bufferpool = BP1
Define = No
Partitions = 0
```

This Tablespace object translates into the following DDL output:

```
CREATE TABLESPACE H1004000 IN SIDB1004
USING STOGROUP SYSDEFLT
PRIQTY 48 SECQTY 1440 FREEPAGE 0 PCTFREE 17
DEFINE NO
SEGSIZE 32
BUFFERPOOL BP1 LOCKSIZE PAGE LOCKMAX 0 COMPRESS YES /
```

Note: If you compare the input and the output, you can see that there are more options in the DDL output. These additional options occur because some values, such as `FREEPAGE` and `PCTFREE`, are taken from the default values.

Object numbering can change when you perform a storage control file extract. Therefore, object numbering can be different from one storage control file to another (except for default storage objects, for which numbering remains consistent).

Object Naming Conventions

Understanding Siebel object naming conventions allows you to keep track of existing Siebel objects and to successfully create and maintain your own objects.

If you understand Siebel naming conventions, you can more easily identify underlying data sets (VSAM files); for example, the name of the underlying data sets for the H0401000 table space might look like Q10E.DSNDBC.SIEBTO.H0401000.I0001.A001, where the fourth node represents the table space name.

> **Note:** Siebel utilities can recognize and use custom objects only if the objects follow Siebel naming conventions; therefore, it is strongly recommended that you use the Siebel naming conventions for objects that you create.

Table Space Naming Conventions

The table space naming convention applies to all Siebel table spaces, whether you create the table spaces or the table spaces are shipped as part of Siebel CRM.

Table spaces are named with eight characters (for example, H0401000), consisting of three elements in the following order:

1. A leading letter (H, V, or S) designates the product group that the table belongs to.
   - H represents Horizontal (also known as Core).
   - V represents Vertical (also known as Siebel Industry Applications or SIA).
   - S represents your custom tables and auxiliary indexes (those that you create for the Siebel application).
2. A four-digit number that is assigned to the base table and is stored in the `GROUP_CD` column of the `S_TABLE` table in the Siebel Repository, for example, 0401. (This number cascades down to table spaces, auxiliary tables, and auxiliary indexes.)

   Table names are stored in the repository table `S_TABLE`, and table numbers are stored in the Group Code column (`GROUP_CD`) in `S_TABLE`. You can view Table and Group names in Siebel Tools by navigating to the Table menu, and then selecting the Object option.

   > **CAUTION:** You can edit the Group property for tables you create but do not edit the Group property for tables provided in the Siebel application; doing so can cause operating anomalies in the application.
3. The ending three-digit number which is always 000 for table spaces on the database storage layout.

> **Note:** When defining a new table space for new custom tables and auxiliary indexes, use the prefix S, follow it with a unique four-digit number greater than 6000, then end with the three-zero suffix (000), for example, S6001000. Using this prefix and suffix maintains object naming conventions.

Naming Conventions for Auxiliary Objects

In the current release of Siebel CRM, the value of the CURRENT RULES special register is set to STD, which causes DB2 to execute SQL statements in accordance with the rules of the SQL standard. As a result, DB2 implicitly creates auxiliary

objects and assigns names to them. For information on the naming conventions used by DB2 when creating auxiliary objects, refer to the IBM DB2 for z/OS documentation.

Naming Conventions for Partitioning Triggers

Partitioning Trigger names comprise eight characters (for example, PTH0401) that consist of three elements in the following order:

1. The leading two letters must be `PT`, to designate that this is a partitioning trigger.
2. The third letter (`H`, `V`, or `S`) designates the product group or component that the trigger belongs to.
   - `H` represents Horizontal (also known as Core)
   - `V` represents Vertical (also known as Siebel Industry Applications or SIA).
   - `S` represents partitioning triggers used for Assignment Manager or other components.
3. The ending four-digit number uniquely identifies the trigger based on the table number, for example, 0401.

   The trigger name is normally related to a corresponding table space name; for example, the trigger name PTH0401 relates to the table space name H0401000.

About Modifying the Database Layout

There are two methods you can use to modify the database storage layout for your deployment:

- **Method 1: Adjust the Storage Control File to Reflect Your Database Layout.** Adjust the object definitions and options in the storage control file to reflect your database storage layout. When you subsequently perform a customized database installation or when you use database utilities, select your new storage control file when you are prompted for the storage control file input parameter. The resulting output file, schema.sql, contains the DDL that reflects your modifications. For information on modifying the storage control file, see *About Modifying Storage Control Files*.

  If you choose to modify the database layout by adjusting the storage control file, you must generate an updated version of the storage.ctl file before running any of the database configuration utilities that require this file as input, for example, the Synchronize Schema Definition (ddlsync) and Migrate Repository (dev2prod) utilities. You can generate an updated version of the storage.ctl file using the Database Configuration Wizard. For information on this task, see *Extracting a Storage Control File from the DB2 Catalog*.
- **Method 2: Modify Generated DDL Using DB2 Tools.** To use this method, perform a standard installation as described in *About Standard Installations*, and select the Generate DDL into Files installation mechanism. After the Database Configuration Wizard generates the DDL to a file named schema.sql, your DBA can directly edit the output DDL using native DB2 tools. Some DBAs prefer this method, because it involves using customary DB2 tools and does not involve manipulation of the storage control file.

  When your DBA has modified the DDL, extract the storage control file so that it reflects the schema layout on the mainframe. This extraction saves time in the future as it means less manual work is involved when you run database utilities, such as Migrate Repository (dev2prod).

About Modifying Storage Control Files

You can modify a storage control file using either of the following methods:

- Open the storage control file with a text editor program and edit it directly to adjust object definitions or options.

  This approach is useful for making minor adjustments to the database layout but only use it if you have a thorough understanding of the storage control file structure. Otherwise, you could introduce errors into the file and affect the operation of your Siebel application.

- Use the Siebel Database Storage Configurator (dbconf.xls).

  This is a Siebel Microsoft Excel macro tool that allows you to edit the attributes of objects in the Siebel-provided storage control file templates. For further information, see *Using the Siebel Database Storage Configurator*.

Using the Siebel Database Storage Configurator

When you install Siebel CRM, the Siebel Database Storage Configurator file, dbconf.xls, is also installed. This tool helps you to configure your Siebel Schema by allowing you to import and edit the attributes of the Siebel-provided storage control file templates. You can also use this tool to configure an existing storage control file.

Dbconf.xls is installed in the DBSRVR_ROOT\ `db2390` (Windows) or DBSRVR_ROOT/ `db2390` (UNIX) directory. To open dbconf.xls, you must have Microsoft Excel installed on your workstation. If your installation is UNIX, you must transfer dbconf.xls and the storage control file that you want to configure to your Microsoft Windows environment. Use binary FTP to transfer the dbconf.xls file.

After you edit a storage control file template using the Database Storage Configurator, save the file with a new name, then direct the Database Configuration Wizard to your newly configured file when prompted to specify a storage control file during installation (see *Installing the Siebel Database on the DB2 Host*). The output DDL is generated from the information in the (configured) storage control file that you specify.

> **Tip:** It is recommended that you preserve the original dbconf.xls spreadsheet and save your modified spreadsheet with a new name. Saving the spreadsheet allows you to review your current database layout while continuing to work on the storage control file. For example, because the spreadsheet allows you to sort data, you can easily identify all table spaces that belong to the same database.

Modifying a Storage Control File Using the Database Storage Configurator

To modify a storage control file using dbconf.xls, perform the following procedure.

To configure one of the Siebel storage control files

1. Open dbconf.xls and select Enable Macros when prompted.

   If you are not prompted to Enable Macros, verify that your Microsoft Excel security setting is Medium, as follows:

   a. Launch Excel, and choose Tools, Macro, and then Security.
   b. Set the security level to medium, and then restart your computer.

      When the dbconf.xls spreadsheet opens, the Home tab is active. Click the Functions tab to select one of the functions available in dbconf.xls. Click the Structures to display information about database objects.
2. Import the storage control file you want to use as your template:

   a. With the Home tab active, click Import.
   b. Go to the directory where your files with a ctl suffix are located.
   c. Double-click the appropriate file.

      When the import process is completed, a message appears, stating that you have imported the storage control file successfully.
   d. Click OK.
3. When the message, `Please enter default values for your system` appears, either type the values for the following parameters into the corresponding fields, or accept the default values.

   Note: Replace variables that are preceded with the $ symbol with actual values for your deployment.

   - Table Storage Group for Table
   - Index Storage for Indexes
   - 4K Buffer Pool Name
   - 8K Buffer Pool Name
   - 16K Buffer Pool Name
   - 32K Buffer Pool Name
   - Index Buffer Pool Name
   - Database Name Prefix
   - Encoding Scheme (ASCII or EBCDIC)
4. Click Set.
5. You are prompted to indicate whether or not you want to import row lengths. Select No.

   You need to import row lengths if you intend to do either of the following:

   - Convert your database from a non-Unicode to a Unicode encoding scheme
   - Perform space calculations using dbconf.xls.

6. Select the Functions tab, then display the template defaults by clicking the Defaults tab.

   The default values are the values that apply to an individual object in the storage control file if no other value is specified. You can change the default value for an individual object by navigating to the relevant tab (select the Structures tab, then the relevant object type tab) and changing the information displayed.

   For example, when you create a new extension table, the Database Storage Configurator takes the Siebel default database (SIEB0000) and table space (SIEBTS00) values of the template used unless you explicitly define new values.

   When you review the values under the object type tabs, for example, Tablespace, an empty parameter cell indicates that the cell takes the default value for the object.
7. Review the default values, and update them as required.

   The parameters in the spreadsheet correspond to native DB2 syntax. For example, the parameters `PriQty` (Primary Quantity) and `SecQty` (Secondary Quantity) are used during the Siebel Schema creation.
8. Review the databases by clicking Structures, and then Database.

   Be aware that the values visible on any tab also reflect the defaults for any objects at a lower level to the object for which the tab exists. Thus, if no locksize is shown for a table space under the Tablespace tab, then that table space takes the default locksize from the database object it belongs to. For example, locksize at the database object level applies to all table spaces created in a given database unless the locksize is specified explicitly in the table space object.
9. Review the table spaces by clicking Structures, and then Tablespace.

   To review any partitions, select a row that displays a value in the Partitions column greater than zero, and then click Show TSPARTs. This displays the rows of corresponding partitions.
10. Edit the values as desired, and then move off the cell to save the value.
11. Review the tables by clicking Structures, and then Table.
12. When you have updated the template, click the Home tab, and then click Export.

    The Export worksheets to storage control file screen appears.
13. Rename the storage control file, for example, **my_storage_p.ctl,** and save it. A message is displayed saying that the file will be validated.
14. Click OK.

    When the validation process is completed, a message is displayed if the file contains any invalid values that you need to review. The following is an example of the type of message that might be displayed:

    ```
    Please review index partitions which are marked in red
    ```

15. Click OK.

    The errors in the file are displayed, highlighted (in red font).
16. Make a note of the object that is generating the errors. The relevant object type tab and the object are highlighted. Click OK.
17. Review and correct the object values that generated the validation errors. For example, if the values specified for a partitioning index partition are incorrect, do the following:

    a. Select Structures, and then Partitioning Index.
    b. Select the relevant index, then click the Show Index Parts button.

       The index partitions are displayed and the cells that contain invalid values are highlighted.
    c. Amend the invalid values, then export the file again.

You cannot export the file until the validation process completes without error.

18. When you have exported the file, click Home, and then Clean, to make the Database Storage Configurator ready for use with a different template.

    A text box appears letting you know that you have cleaned all data successfully.

Validating Your Work

You can use the Database Storage Configurator Validate button at any time to validate the syntax in the dbconf.xls spreadsheet. Wait to validate until after you have finished editing to ensure your worksheets are consistent.

> **Note:** The spreadsheet validation option is not a substitute for using the Database Configuration Wizard to validate the storage control file after you amend it. The Validate Storage File option of the Database Configuration Wizard validates the new storage control file against the physical layout of the database.

To validate your dbconf.xls worksheet

1. After updating values for any of the template defaults, click the Validate button.

   The Database Storage Configurator validates the new value; if the value is incorrect, it is highlighted (in red font). A text box also appears prompting you to review the invalid settings.
2. Update any incorrect value, move off the cell, and validate the value again.

   The (red) highlighting disappears when the validation shows that the value is correct.

Working with CLOBs Using the Database Storage Configurator

Your Siebel application is delivered with the objects needed to create character large objects (CLOBs). You can use these objects to create CLOBs at the table level only, to avoid excessive processing overhead.

The following procedure describes how to activate a CLOB.

To activate a CLOB

1. Launch the Database Storage Configurator.

   This task is described in *Modifying a Storage Control File Using the Database Storage Configurator*.
2. Click Structures, and then the Table tab.
3. Review the contents of the CLOB column.

   To activate a specific CLOB, type Yes in the appropriate row in the CLOB column, and then save the value by stepping off the cell.

   > **Note:** The Auxiliary Tablespace tab displays a table name that is used as a pointer to a real auxiliary table space. The parameters displayed under this tab are the only parameters you can change for the auxiliary object.

4. Make any other changes needed to the template you are using, then verify and save the changes.

   For information on this task, see *Modifying a Storage Control File Using the Database Storage Configurator*.

6 About Siebel Table Partitioning

About Siebel Table Partitioning

This chapter provides information on partitioning large tables in Siebel CRM. Partitioning a DB2 table is not a complicated procedure, but to optimally partition Siebel tables, it is critical that you understand how partitioning works in DB2.

This chapter consists of the following topics:

About Siebel Partitioning

Partitioning table spaces on DB2 allows tables to be spread across multiple physical partitions based on a partitioning key, a set of key value ranges for each partition, and optionally, a partitioning index. Using partitioned table spaces increases the maximum size of a table and improves the manageability of large tables.

Partitioning EIM table spaces by batch number improves EIM performance for batching and parallel processing, and allows distribution of key ranges across multiple data sets.

Any table can be partitioned during the installation or upgrade process. You can define partitioned table spaces and key ranges for Siebel tables during or after installation, based on your business requirements. For a complete list of prepartitioned Siebel tables, see *Prepartitioned Siebel Tables*.

You can partition tables yourself by following Oracle's guidelines, or you can take advantage of the default partitioning scheme that Oracle developed, based on customer experience using the Siebel data model with DB2 for z/OS. If you use the Siebel default partitions, you can either accept them as they are, or you can reconfigure them to suit your requirements.

DB2 for z/OS Version 8 introduced a number of new partitioning capabilities. None of these capabilities are required to run Siebel CRM applications, but many of the key features are supported in Siebel releases since 8.0, including the following:

- You can specify up to 1296 partitions for tables partitioned on the last 2 bytes of ROW_ID. For further information, see *Partitioning for Even Data Distribution*. For tables that use any other partitioning key, up to 4096 partitions can be specified.

- The current Siebel CRM release supports table-controlled and index-controlled partitioning. If you use table controlled partitioning, you do not have to maintain indexes that are used for partitioning only.
- With table-controlled partitioning, clustering is separated from partitioning. The partitioning index is not automatically the clustering index, so data in a partitioned table space does not have to be clustered by the partitioning key. Data can be clustered by partition using a secondary unique index that you choose.

  > **Note:** Clustering related records within a partition generally improves performance and is particularly important for tables with non-unique access paths, for example, intersection tables.

- Partitioning indexes can be defined for indexes if the associated table uses table-controlled partitioning and is not partitioned by ROW_ID. For further information, see *Considerations in Partitioning Tables*.

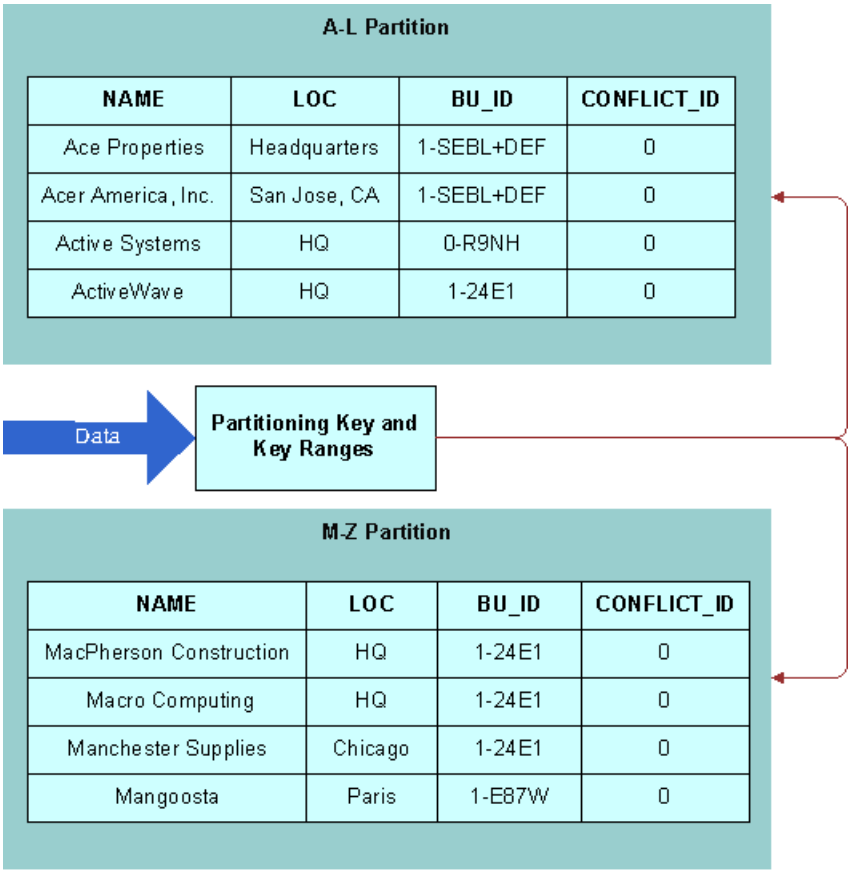
About Partitioning Keys

When a table is created on a partitioned table space, the table is assigned a partitioning key that is composed of one or more columns. Value ranges are assigned to each partition based on value ranges within the partitioning key. The value ranges determine which partition a particular row is assigned to.

How key values are specified for each table partition depends on whether the table uses table-controlled or index-controlled partitioning. With index-controlled partitioning, a partitioning index specifies the partitioning key and the key value ranges (limit keys) that determine how data is partitioned. With table-controlled partitioning, the partition key and key value ranges are contained in the table definition, so a partitioning index is not required. In the current release, partitioned tables use table-controlled partitioning by default.

> **Note:** Multi-column limit keys are not supported on tables that are partitioned using a partitioning index. If you want to specify multi-column limit keys for a partitioned table, use table-controlled partitioning.

An example of partitioning in shown in the following figure. You can divide a large table such as S_ORG_EXT (which holds a list of new accounts) to store records of names beginning with letters A-L in one partition and records of names beginning with letters M-Z in another partition. In this example, the partitioning key is the Name column and a partitioning index defines the key ranges (A-L; M-Z) for the partitioning column. A key range must be specified for each partition.



Note: For more information about partitioning table spaces on DB2, consult your IBM documentation.

Partitioning and the Storage Control Files

Partitioning is defined in the Siebel storage control files. You can use one of the following storage control file templates provided with Siebel CRM to partition tables:

- **storage_p.ctl.** This template is the database storage layout for the Siebel Schema with partitioning for ASCII encoding.
- **storage_p_u.ctl.** This template is the database storage layout for the Siebel Schema with partitioning for Unicode encoding.
- **storage_p_e.ctl.** This template is the database storage layout for the Siebel Schema with partitioning for EBCDIC encoding.

Storage control files are described in detail in *Configuring the Siebel Database Layout*.

Each of the partitioning storage control file templates contain the same Siebel-recommended partitioning schema. The templates provide a set of 17 partitioned base tables for the Horizontal product line and a set of 24 partitioned base tables for Siebel Industry Applications. The default partitioning scheme is optimized for online activities from the Siebel Client.

Siebel CRM' partitioning templates provide 10 partitions for each partitioned table. You select the number of partitions for your implementation based on your business needs, and define partitioning keys based on the number of partitions.

If you want to customize the Siebel table-partitioning layout, you can do so by editing the appropriate storage control file template directly. Alternatively, you can use the Siebel Database Storage Configurator (dbconf.xls) to modify the template (recommended). For more information, see *Partitioning Tables and Indexes Using the Database Storage Configurator*.

Considerations in Partitioning Tables

Careful planning, requirements analysis, and monitoring are necessary to achieve optimal partitioning and optimal performance. If partitioning is not planned and carried out carefully, performance and scalability are adversely affected. Decide which Siebel tables to partition and how to partition them on the basis of table size and usage in your deployment. You can partition any table in accordance with your business requirements; you can select all the tables that provide for partitioning, or you can select a subset of them.

About Choosing a Partitioning Key Column

You can use any column from a Siebel base table as the partitioning key for a Siebel table but you cannot use extension columns you create for partitioning purposes. In deciding which column is most appropriate for partitioning, consider data access and data distribution factors. You also need to determine the number of partitions your implementation requires.

When choosing a partitioning key for a Siebel table, follow all the rules, restrictions, and concerns listed in the IBM DB2 documentation. Factors to consider include data distribution and potential updates of partitioning keys. For example, when partitioning keys are updated, system performance is adversely affected. When choosing partitioning keys, select fields that are seldom updated, when possible. This helps avoid performance problems.

The partitioning key that provides optimum performance and scalability varies according to the type of table being partitioned. For example, if in an organization Contact records are retrieved by LAST_NAME range predicates, for example, `LAST_NAME LIKE 'Huang%'`, then partitioning on the LAST_NAME column is recommended to keep all similar records in one partition, while clustering on the LAST_NAME column ensures the records returned by the query are contiguous.

However, a different approach is required for child tables, especially those in a one-to-many (1:M) relationship with the parent table. If child tables are partitioned, it is recommended that you use the PAR_ROW_ID column for both the partitioning and clustering keys to reduce the input/output steps required to return related child records.

As an alternative to using Siebel table columns as partitioning keys, you can use either the Siebel PARTITION_COLUMN or MEMBER_NAME columns. These columns are used for partitioning purposes only. The PARTITION_COLUMN column is a physical column that is defined in the storage control file, but is not accessed by the Siebel application. It exists to provide a method of ensuring even data distribution across table partitions where none of the table columns are good candidates for partitioning. The column is populated with data using a trigger to generate the partitioning value for each row. For additional information on the role of PARTITION_COLUMN, see *Partitioning for Even Data Distribution*. For information on the role of the MEMBER_NAME column, see *Partitioning Strategies for Special Types of Tables*.

About Table-Controlled Partitioning and Using Indexes

It is recommended that you use table-controlled partitioning with Siebel database tables.

In the current Siebel release, partitioned tables use table-controlled partitioning by default. If you use table-controlled partitioning, partitioning is defined at the table level so you do not require a partitioning index. Maintaining indexes that are used for partitioning purposes only is no longer necessary, which reduces storage requirements and allows you to select a secondary index as the clustering index.

About Choosing a Clustering Index

In previous versions of DB2, the partitioning index was automatically the clustering index. In DB2 for z/OS Version 8 and later releases, data can be clustered by any index. Data in a partitioned table space no longer needs to be clustered by the partitioning index, although in many installations, this might be the most efficient option. For example, it is recommended that child tables are partitioned so that all records related by the parent table ROW_ID are contiguous within the same partition and a similar approach is recommended for intersection tables. By choosing the most appropriate clustering index for each of the Siebel-partitioned tables, you can optimize query performance.

In choosing a clustering index, consider both day-to-day online access to the Siebel tables and processes such as EIM, Siebel Remote, Assignment Manager, and Workflow. The recommended clustering index for each of the Siebel-partitioned base tables is shown in the following table.

Only one clustering index can be defined for a table. When you have altered the index to make it clustering, run the REORG utility on the table space and then run RUNSTATS.

Note: It is recommended that you do not partition secondary indexes for performance reasons.

About Using Partitioning Indexes

Although the prepartitioned Siebel tables use table-controlled partitioning, you can also define partitioning indexes for Siebel tables that are not partitioned using PARTITION_COLUMN. For information on the role of PARTITION_COLUMN in partitioning tables, see *Partitioning for Even Data Distribution*.

You might want to define a partitioning index for a Siebel table-controlled partitioned table to improve query performance, if, for example, the table and its associated index are very large. You can partition the index provided it has the same columns and limit keys as the partitioning key values defined for the table.

Partitioning indexes are distinguished from partitioned indexes. A partitioning index must:

- Contain all of the columns from the partitioning key, at a minimum

  Note: A partitioning index can also contain additional columns, for example, columns required to support optimal clustering or performance enhancement.

- The columns must be in the same order
- The columns must have the same sort order

A partitioned index does not have the same partitioning key values as the key values on the table. Siebel CRM does not support partitioned indexes.

Partitioning indexes are defined in the storage control file templates for the prepartitioned Siebel base tables that do not use PARTITION_COLUMN. These tables are:

- S_CONTACT
- S_OPTY
- S_ORG_EXT
- EIM_ACCOUNT
- EIM_PROD_INT
- EIM_OPTY
- EIM_CONTACT
- EIM_SRV_REQ
- EIM_ACTIVITY
- EIM_DEFECT
- EIM_ASSET
- EIM_ACCNT_CUT
- EIM_FN_CONTACT1
- EIM_FN_ASSET
- EIM_FN_ASSET1
- EIM_FN_ASSET2
- EIM_FN_INSITM1
- EIM_FN_INSCLM1
- EIM_FN_ORGGRP
- EIM_ASSET_AT
- EIM_VHCL_SRV

For more information about the Siebel partitioned tables, see *Prepartitioned Siebel Tables*.

About Table Partitioning Methods

Siebel CRM supports two methods for partitioning tables:

- *Partitioning Based on Business Data*
- *Partitioning for Even Data Distribution*

Partitioning Based on Business Data

In this partitioning method, a table is partitioned based on existing columns in the Siebel Schema. For example, S_OPTY is partitioned by columns in the U1 index. In this case, no special action is needed except to define the key ranges and number of partitions.

If multiple organizations are defined in your Siebel application, partitioning on the BU_ID column might be more efficient than using the Siebel-supplied partitioning index. In these cases, clustering on the BU_ID column, then other

relevant columns, ensures related records are contiguous within a partition for a given business unit, which optimizes query performance.

If your data analysis reveals that the number of records associated with business units is skewed, so that some business units have few records, it might be appropriate to allocate groups of business units to one partition; in these cases, choose your clustering key column with care.

Partitioning for Even Data Distribution

This partitioning method involves partitioning a table using an additional partitioning column, PARTITION_COLUMN, designed specifically for Siebel partitioning with even data distribution. This column is populated with data using a BEFORE INSERT trigger option to generate the partitioning value for each row.

Most Siebel tables are tied together by the ROW_ID column from a parent table. Columns with an _ID suffix are used to define a parent-child relationship, for example, OU_ID. These columns might seem to be good partitioning candidates because they support the DB2 access path but, in fact, these columns are poor candidates for partitioning because ROW_IDs are generated in sequential order. (An exception is if the values in a column with an _ID suffix identify groups of related records; in this case, the column might be a suitable partitioning key.)

To resolve the limitation caused by the sequential order of ROW_ID, Siebel CRM provides a column, PARTITION_COLUMN, which is used only for partitioning purposes. It is a physical column defined in a storage control file, but it is not a part of the Siebel Repository. It is important that you continue to use the name PARTITION_COLUMN if the Siebel application is to recognize this column. You must also define this column as NOT NULL WITH DEFAULT.

A DB2 BEFORE INSERT trigger is used to populate the PARTITION_COLUMN values; it is defined in the Siebel storage control file. This trigger generates a random number between 00 and 10 and uses it to populate PARTITION_COLUMN. The partition limit keys are also numbers between 01 and 10. Rows are assigned to each of the 10 partitions based on the value of PARTITION_COLUMN generated for the row. For example, if the number generated for PARTITION_COLUMN is 6, the row is assigned to the partition with a limit key value of 6. By using a trigger to populate the columns, you can partition tables that do not have good candidate columns for a partitioning key due to their data content. Using a trigger, you can still generate values that distribute the data well.

The EIM tables and some of the prepartitioned base tables do not use PARTITION_COLUMN. For information on partitioning EIM tables, see *EIM Tables and Partitioning*.

When partitioning a nonpartitioned Siebel table, or when customizing a prepartitioned table, you need to assess whether PARTITION_COLUMN is the most appropriate partitioning key for the specific table. To improve query performance, ensure the partitioning key you use keeps related records together within each partition.

Example of Partitioning the S_ADDR_ORG Table

You can use the Siebel-selected partitioned tables exactly as they are shipped. You can also customize the Siebel partitioned tables, or partition other Siebel schema base tables. For information on customizing the partitioned tables or partitioning nonpartitioned tables, see *Partitioning Tables and Indexes Using the Database Storage Configurator*.

This topic gives one example of how to partition a table and a table space. The example definitions in this topic reflect a partitioning scenario for the S_ADDR_ORG table, which resides in the H0401504 partitioned table space. The Siebel partitioning template used for this scenario is the storage_p_e.ctl storage control file.

In a storage control file, the partitioned table space is defined with two types of storage control file objects: Tablespace and Tspart (table space partitions). The storage control file objects related to partitioning are:

- Tablespace
- Tspart (table space partitions)
- Table
- PartitionBase (defines the partitioning key for table-controlled partitioned tables)
- PartitionPart (defines the key value ranges used for partitioning data for table-controlled partitioned tables)
- IndexBase (base definition of a partitioning index)
- IndexPart (index partitions)

A table definition always includes the same options, whether the corresponding table space is partitioned or not. However, the table space, partition and index objects include different options, depending on whether the table space is partitioned or nonpartitioned.

Note: The object numbers that identify the objects in this example might differ in your implementation.

Example of a Table Object Definition

The following is an example object definition for table S_ADDR_ORG:

```
[Object 6855]
Type = Table
Name = S_ADDR_ORG
Database = D0059504
Tablespace = H0401504
Partitions = 10
Clobs = No
```

Example of a Table Space Definition

The definition for Tablespace object H0401000 follows:

```
[Object 3643]
Type = Tablespace
Name = H0401504
Database = D0059504
LockSize = Page
Bufferpool = BP16K1
Define = No
Partitions = 10
Lockpart = Yes
```

You can easily identify a partitioned table space by reviewing the Partitions option in the Tablespace object in the storage control file. If the Partitions option is greater than zero, then the table space is partitioned and requires additional objects, such as Tableparts (Tspart) and partition or index objects. The number of Tspart objects is equal to the number of partitions.

Example of Definitions for Table Space Partitions

The example for the Tablespace object in *Example of a Table Space Definition* specifies 10 partitions; therefore, ten Tspart objects must be defined, as shown in the following example for objects 3644 through 3653).

```
[Object 3644]
Type = Tspart
Name = H0401504
PartNum = 1
...
[Object 3653]
Type = Tspart
Name = H0401504
PartNum = 10
```

Example of the DDL for the Partitioned Table Space

The storage control file definitions for the partitioned table space in the previous example result in the following output DDL statements:

```
CREATE TABLESPACE H0401504 IN D0059504 USING STOGROUP SYSDEFLT PRIQTY 48 SECQTY 1440
FREEPAGE 0 PCTFREE 17 LOCKPART YES DEFINE NO NUMPARTS 10 ( PART 1 USING STOGROUP
SYSDEFLT ,
PART 2 USING STOGROUP SYSDEFLT ,
PART 3 USING STOGROUP SYSDEFLT ,
PART 4 USING STOGROUP SYSDEFLT ,
PART 5 USING STOGROUP SYSDEFLT ,
PART 6 USING STOGROUP SYSDEFLT ,
PART 7 USING STOGROUP SYSDEFLT ,
PART 8 USING STOGROUP SYSDEFLT ,
PART 9 USING STOGROUP SYSDEFLT ,
PART 10 USING STOGROUP SYSDEFLT ) BUFFERPOOL BP16K1 LOCKSIZE PAGE LOCKMAX 0 COMPRESS
YES
```

Example of a Partition Definition

If you use the table S_ADDR_ORG, the column OU_ID appears to be a good candidate for a partitioning key. However, OU_ID contains data in the Siebel row ID format and row IDs are generated in ascending order. Instead a trigger is used to generate a random number between 00 and 10 and this number is stored in a new physical PARTITION_COLUMN. This column is defined in the storage control file for S_ADDR_ORG.

> **Note:** If you have evenly sized table partitions, using PARTITION_COLUMN as the partitioning key provides query performance optimization. If you do not have evenly sized table partitions, a different partitioning column might be more effective in enhancing performance and scalability.

For tables that use table-controlled partitioning, you must define two types of objects in the storage control file: PartitionBase and PartitionPart. These objects specify the partitioning key column and partition value ranges. An example definition for each is shown.

Note: If you are defining a partitioning index, you must create IndexBase and IndexPart objects to specify the partitioning key and value ranges.

PartitionBase Definition

The example for object 9401 contains a DB2 BEFORE INSERT trigger to populate PARTITION_COLUMN. This is always defined by the SpecialCol keyword in a PartitionBase section. The syntax for the trigger implements the column PARTITION_COLUMN to S_ADDR_ORG as `CHAR(2) NOT NULL` with default of a space. It also implements a trigger to populate the partitioning column.

```
[Object 9401]
Type = PartitionBase
Name = S_ADDR_ORG
Partitions = 10
SpecialCol = PARTITION_COLUMN WCHAR(2) NOTNULL DEFAULT ' '
Function = BEGIN ATOMIC SET N.PARTITION_COLUMN = RIGHT(DIGITS(INT(MOD(RAND()*
9991, 10)) + 1), 2); END
Column 1 = PARTITION_COLUMN ASC
```

The performance overhead of using a trigger is measured as a percentage of the inserts. Performance overhead depends on several factors, including row length and the number of indexes in the table.

Triggers that use the S_PARTY partitioned table space, 10 partitions, 14 columns, and 443 bytes with 8 indexes, have demonstrated a performance overhead of less than 1% for each insert.

PartitionPart Definition

In the example for object 9401, the PartitionBase section defines ten (10) partitions. Therefore, it requires ten PartitionPart objects. These are illustrated in the example for objects 9402 through 9411:

```
[Object 9402]
Type = PartitionPart
Name = S_ADDR_ORG
PartNum = 1
LimitKey = '01'
...
[Object 9411]
Type = PartitionPart
Name = S_ADDR_ORG
PartNum = 10
LimitKey = MAXVALUE
```

Siebel CRM provides generic values for the partitioning keys. Review these LimitKey values in the PartitionPart object and change them to suit the special requirements of your implementation.

Note: The sorting order for EBCDIC values is different from ASCII.

Example of the DDL for a Partitioned Table

The storage control file definitions for the partitioned table in the previous example result in the following output DDL statements:

```
CREATE TABLE CQ10L007.S_ADDR_ORG ( "ACTIVE_FLG" CHAR(1) DEFAULT 'Y' NOT NULL,
"ADDR" VARCHAR(200),
"ADDR_LINE_2" VARCHAR(100),
"ADDR_LINE_3" VARCHAR(100),
"ADDR_NAME" VARCHAR(100) DEFAULT 'x' NOT NULL,
"ADDR_NUM" VARCHAR(30),
"ADDR_TYPE_CD" VARCHAR(30),
```

```
"ALIGNMENT_FLG" CHAR(1) DEFAULT 'N' NOT NULL,
"BL_ADDR_FLG" CHAR(1) DEFAULT 'Y' NOT NULL,
"BU_ID" VARCHAR(15),
"CITY" VARCHAR(50),
"COMMENTS" VARCHAR(255),
"CONFLICT_ID" VARCHAR(15) DEFAULT '0' NOT NULL,
"COUNTRY" VARCHAR(30),
"COUNTY" VARCHAR(50),
"CREATED" TIMESTAMP DEFAULT NOT NULL,
"CREATED_BY" VARCHAR(15) NOT NULL,
"DB_LAST_UPD" TIMESTAMP,
"DB_LAST_UPD_SRC" VARCHAR(50),
"DCKING_NUM" NUMERIC(22, 7) DEFAULT 0,
"DFLT_SHIP_PRIO_CD" VARCHAR(30),
"DISA_CLEANSE_FLG" CHAR(1) DEFAULT 'N' NOT NULL,
"EMAIL_ADDR" VARCHAR(100),
"FAX_PH_NUM" VARCHAR(40),
"INTEGRATION2_ID" VARCHAR(30),
"INTEGRATION3_ID" VARCHAR(30),
"INTEGRATION_ID" VARCHAR(30),
"LAST_UPD" TIMESTAMP DEFAULT NOT NULL,
"LAST_UPD_BY" VARCHAR(15) NOT NULL,
"LATITUDE" NUMERIC(22, 7),
"LONGITUDE" NUMERIC(22, 7),
"MAIN_ADDR_FLG" CHAR(1) DEFAULT 'Y' NOT NULL,
"MODIFICATION_NUM" NUMERIC(10, 0) DEFAULT 0 NOT NULL,
"NAME_LOCK_FLG" CHAR(1) DEFAULT 'N' NOT NULL,
"OU_ID" VARCHAR(15) NOT NULL,
"PARTITION_COLUMN" CHAR(2) DEFAULT ' ' NOT NULL,
"PH_NUM" VARCHAR(40),
"PROVINCE" VARCHAR(50),
"ROW_ID" VARCHAR(15) NOT NULL,
"SHIP_ADDR_FLG" CHAR(1) DEFAULT 'Y' NOT NULL,
"STATE" VARCHAR(10),
"TRNSPRT_ZONE_CD" VARCHAR(30),
"ZIPCODE" VARCHAR(30)) IN D0059504.H0401504
PARTITION BY (PARTITION_COLUMN ASC)(
PARTITION 1 ENDING AT ('01') ,
PARTITION 2 ENDING AT ('02') ,
PARTITION 3 ENDING AT ('03') ,
PARTITION 4 ENDING AT ('04') ,
PARTITION 5 ENDING AT ('05') ,
PARTITION 6 ENDING AT ('06') ,
PARTITION 7 ENDING AT ('07') ,
PARTITION 8 ENDING AT ('08') ,
PARTITION 9 ENDING AT ('09') ,
PARTITION 10 ENDING AT (MAXVALUE) )
/
CREATE TRIGGER CQ10L007.PTH0401 NO CASCADE BEFORE INSERT ON CQ10L007.S_ADDR_ORG
REFERENCING NEW AS N FOR EACH ROW MODE DB2SQL BEGIN ATOMIC SET N.PARTITION_COLUMN =
RIGHT(DIGITS(INT(MOD(RAND()* 9991, 10)) + 1), 2); END
/
CREATE UNIQUE INDEX CQ10L007.S_ADDR_ORG_U1 ON CQ10L007.S_ADDR_ORG ("ADDR_NAME",
"OU_ID", "CONFLICT_ID") USING STOGROUP SYSDEFLT PRIQTY 48 SECQTY 1440 PCTFREE 17
DEFINE NO CLUSTER BUFFERPOOL BP2
/
```

The trigger name, PTH0401, is based on a number assigned to the S_ADDR_ORG table within the Siebel Repository. The S_ADDR_ORG table is defined in table space H0401504. The trigger name and the table space name are both based on the number assigned to the S_ADDR_ORG table and stored in the table Group Code in the Siebel Repository.

The table is partitioned on the PARTITION_COLUMN column, which is populated by the trigger PTH0401. The index S_ADDR_ORG_U1 is defined with the CLUSTER parameter so it becomes the clustering index; that is, rows within each partition are clustered according to the key of S_ADDR_ORG_U1.

Schedule routine REORGs of the table space and index space regularly to sustain clustering order. Running REORGs regularly accommodates insert activity and reclaims PCTFREE and FREEPAGE definitions.

When you are defining key ranges, remember that EBCDIC sorting order differs to Unicode and ASCII sorting orders. Numeric characters precede alphabetical characters in ASCII and Unicode, but the opposite applies in EBCDIC.

> **Note:** For most processes, the Siebel Row ID is generated as a BASE 36 value that contains the alphabetical characters A through Z and the numeric characters zero through nine (0-9). The EIM process uses a suffix that contains numeric characters zero through nine, so take this into account in defining the key ranges.

Partitioning Strategies for Special Types of Tables

This topic explains the special considerations you must take into account when partitioning tables that support specific Siebel business processes, such as Siebel Remote, Siebel Assignment Manager, Siebel Workflow, and Siebel Enterprise Integration Manager (EIM).

The following topics describe partitioning considerations for specific tables:

- *Siebel Remote Transaction Logging Tables*
- *S_ESCL_REQ and S_ESCL_LOG Tables*
- *EIM Tables and Partitioning*

Siebel Remote Transaction Logging Tables

This topic explains the special considerations you must take into account when partitioning tables that support Siebel Remote.

Siebel Remote uses the S_DOCK_TXN_LOG table for transaction logging. All user changes on the Siebel Server are logged in this table. Changes are then routed to mobile users according to their responsibilities and user privileges.

To reduce the performance impact on the coupling facility in such a data sharing environment, avoid giving multiple members ReadWrite access to the same partition. You can do this by partitioning the table S_DOCK_TXN_LOG by the MEMBER_NAME column. The MEMBER_NAME column is populated by the CURRENT MEMBER special register, thereby ensuring that all write processes associated with the member affect only one partition.

Partitioning by MEMBER_NAME is not the default for the S_DOCK_TXN_LOG table, because such partitioning is only required for a data sharing environment. Therefore, the names of the data sharing members are not known by the Siebel application.

To specify MEMBER_NAME as the partitioning key, create a PartitionBase object for the table in the storage control file using the syntax in the following example:

```
[Object 8837]
Type = PartitionBase
Name = S_DOCK_TXN_LOG
Partitions = 10
Column 1 = MEMBER_NAME
```

Turning Off Transaction Logging

If you do not have mobile users or if you intend to use your mobile users for extract only, it is recommended that you turn off transaction logging to improve performance.

To turn off transaction logging

1. Navigate to the Administration - Siebel Remote screen, and then the Remote System Preferences view.
2. In the Remote System Preferences form, clear the Enable Transaction Logging check box.

S_ESCL_REQ and S_ESCL_LOG Tables

The tables S_ESCL_REQ and S_ESCL_LOG, which are used by Assignment Manager and Siebel Business Process Designer, tend to grow very large. S_ESCL_REQ is processed based on the GROUP_ID, whereas S_ESCL_LOG is processed by RULE_ID.

You can partition these tables in one of the following ways:

- **Hard-code the actual values for GROUP_ID and RULE_ID.**

  This approach is recommended when the number of groups is static. You can define additional partitions to accommodate new values that are unknown at creation time. To use this option, first define the groups and then extract the GROUP_ID values you want to use for partitioning.

- **Add the PARTITION_COLUMN and a trigger to populate the column.**

  The number of partitions you define should ensure that data from multiple groups goes into multiple partitions. Start with 36 partitions, because this is the base number for the Siebel row ID.

> **Note:** If resource contention occurs because multiple groups are using the same partition, consider increasing the number of partitions when you create new groups and assign new group IDs; this allows for a more granular level of GROUP_ID assignments to a partition range.

EIM Tables and Partitioning

To speed up Enterprise Integration Manager (EIM) load and reduce I/O (input and output) contention, spread partitions for EIM tables across the entire I/O subsystem.

It is recommended that you partition EIM tables based on their U1 indexes, that is, the IF_ROW_BATCH_NUM and ROW_ID columns. This method of partitioning allows an EIM batch input to be assigned to one partition, thereby allowing multiple EIM batches to be run in parallel.

The sample storage control files provided with Siebel CRM contain partitioning for a number of EIM tables based on their IF_ROW_BATCH_NUM column; these are the tables that are recommended for partitioning. You can accept this recommended partitioning approach or use the Database Storage Configurator to perform your own custom partitioning of EIM tables.

> **Note:** If you choose to use the partitioned EIM tables in the Siebel partitioning schema, before using EIM, verify that you are using the correct key ranges, because key ranges depend on the batch numbers used.

To compute the optimal number of partitions for an EIM table, divide the number of rows in the EIM table by the number of parallel processes you intend to run. The result is the approximate number of partitions you need to create for the table.

Example of Using a Non-U1 Partitioning Index

Although the recommended partitioning key for EIM tables is usually the U1 index (IF_ROW_BATCH_NUM and ROW_ID), in some cases, you might need to consider a different partitioning key.

The following examples illustrate how you might partition the EIM tables that are used to populate the S_ORG_EXT and S_OPTY target base tables. Using the columns shown ensures that the data is physically clustered based on the clustering index keys of the target tables.

- EIM_ACCOUNT IF_ROW_BATCH_NUM ASC, NAME ASC, ACCNT_BI ASC, LOC ASC
- EIM_OPTY IF_ROW_BATCH_NUM ASC, OPTY_NAME ASC

For information on using the Database Storage Configurator to perform your own custom partitioning of EIM tables, see *Modifying a Storage Control File Using the Database Storage Configurator*.

Prepartitioned Siebel Tables

Siebel CRM provides in the storage control file templates the partitioned tables shown in the following table. The database tables listed represent partitioned Core product tables; seven exceptions are identified as Siebel Industry Applications (SIA) product tables.

For each table in the following table, the partitioning key columns for the table are listed, as is the recommended clustering index for the table. A partitioning index is defined for three of the partitioned tables (`S_CONTACT`, `S_OPTY`, and `S_ORG_EXT`); the name of the index is listed for these tables.

When determining the partitioning strategy for your implementation, you must assess whether the partitioning key columns and recommended clustering indexes specified for the prepartitioned tables are appropriate for your environment. For example, in assessing the most appropriate partitioning column to use for intersection tables, take into account the relationship of candidate partitioning columns to their respective parent tables. Additionally, if multiple organizations are defined in your Siebel application, the BU_ID column might be the most appropriate partitioning column for a table. For information on customizing partitioned tables, see *Changing the Number of Table Space Partitions*.

Table	Partitioning Key Columns	Recommended Clustering Index
`S_ACT_EMP`	`PARTITION_COLUMN`	`S_ACT_EMP_M1`
`S_ADDR_ORG`	`PARTITION_COLUMN`	`S_ADDR_ORG_U1`
`S_ADDR_PER`	`PARTITION_COLUMN`	`S_ADDR_PER_U1`
`S_APPLD_CVRG` Siebel Industry Applications (SIA) table	`PARTITION_COLUMN`	`S_APPLD_CVRG_U1`

Table	Partitioning Key Columns	Recommended Clustering Index
S_ASSET	PARTITION_COLUMN	S_ASSET_U1 (ASSET_NUM, PROD_ID, REV_NUM, BU_ID, CONFLICT_ID)
S_ASSET_CON	PARTITION_COLUMN	S_ASSET_CON_U1 (ASSET_ID, CONTACT_ID, RELATION_TYPE_CD, CONFLICT_ID)
S_ASSET_POSTN Siebel Industry Applications (SIA) table	PARTITION_COLUMN	S_ASSET_POSTN_M1
S_COMMUNICATION	PARTITION_COLUMN	S_COMMUNICATION_U1
S_CONTACT A partitioning index, S_CONTACT_M12, is defined for this table.	LAST_NAME **Note:** If multiple organizations are defined in your Siebel application, the BU_ID column might be a more effective partitioning key than the LAST_NAME column. In this case, cluster on the S_CONTACT_M14 index (BU_ID, LAST_NAME, FST_NAME, PRIV_FLG).	S_CONTACT_M12 (LAST_NAME, FST_NAME, MID_NAME, PRIV_FLG, ALIAS_NAME)
S_EVT_ACT	PARTITION_COLUMN	S_EVT_ACT_P1
S_EXP_ITEM	PARTITION_COLUMN	S_EXP_ITEM_U1
S_FN_ACCNT_TXN Siebel Industry Applications (SIA) table	PARTITION_COLUMN	S_FN_ACCNT_TXN_U1
S_FNCVRG_ELMNTS Siebel Industry Applications (SIA) table	PARTITION_COLUMN	S_FNCVRG_ELMNTS_U1
S_INS_CLAIM Siebel Industry Applications (SIA) table	PARTITION_COLUMN	S_INS_CLAIM_U1
S_INS_ITEM Siebel Industry Applications (SIA) table	PARTITION_COLUMN	S_INS_ITEM_U1
S_INSCLM_ELMNT	PARTITION_COLUMN	S_INSCLM_ELMNT_U1

Table	Partitioning Key Columns	Recommended Clustering Index
Siebel Industry Applications (SIA) table		
S_OPTY A partitioning index, **S_OPTY_U1**, is defined for this table.	**NAME** **Note:** If multiple organizations are defined in your Siebel application, the **BU_ID** column might be a more effective partitioning key than the **NAME** column. In this case, cluster on the **BU_ID**, **NAME** columns.	**S_OPTY_U1 (NAME, PR_DEPT_OU_ID, BU_ID, CONFLICT_ID)**
S_OPTY_POSTN	**PARTITION_COLUMN**	**S_OPTY_POSTN_U1 (OPTY_ID, POSITION_ID, CONFLICT_ID)** **Note:** If your Siebel application is configured to use My Views to access the **S_OPTY_POSTN** table by the **POSITION_ID** column, cluster on the **S_POSTN_CON_M12** index.
S_ORG_EXT A partitioning index, **S_ORG_EXT_U1**, is defined for this table.	**NAME**	**S_ORG_EXT_U1 (NAME, LOC, BU_ID, CONFLICT_ID)**
S_PARTY	**PARTITION_COLUMN**	**S_PARTY_P1**
S_PARTY_REL	**PARTITION_COLUMN**	**S_PARTY_REL_U1**
S_POSTN_CON	**PARTITION_COLUMN**	**S_POSTN_CON_M1 (POSTN_ID, CON_LAST_NAME, CON_FST_NAME)** **Note:** If the **S_POSTN_CON** table is not implemented as partitioned, cluster on either the **POSTN_ID** or the **CON_ID** column, depending on the one-to-many relationship of each column to its parent table.
S_REVN	**PARTITION_COLUMN**	**S_REVN_U1 (REVN_ITEM_NUM, BU_ID, CONFLICT_ID)**
S_SRV_REQ	**PARTITION_COLUMN**	**S_SRV_REQ_U1 (SR_NUM, BU_ID, CONFLICT_ID)**

Partitioning Tables and Indexes Using the Database Storage Configurator

You can use the Database Storage Configurator file (dbconf.xls) to partition tables or customize the prepartitioned tables after you have worked out a partitioning scheme.

The following sections describe how to partition a nonpartitioned table and how to customize a partitioned table. Partitioning is carried out at the table level; however, you must also adjust the related table space and partitioning key values.

> **Note:** The dbconf.xls spreadsheet Validate option is not a substitute for using the Database Configuration Wizard to validate the storage control file after you amend it by partitioning tables. The Validate Storage File option of the Database Configuration Wizard validates the new storage control file against the physical layout of the database.

Partitioning a Table

The following procedure describes how to partition a nonpartitioned table in a table space.

To partition a table

1. Launch the Database Storage Configurator, and import the storage control file you want to amend.

   For information on this task, see *Modifying a Storage Control File Using the Database Storage Configurator*.
2. Click Structures, Table, and then select the table you want to partition.
3. Click the Partition Table button.
4. You are prompted to specify the number of partitions you want to define for the table. Enter the appropriate number and click OK.
5. Click the Structures tab, and then TBL Partition Base; enter values for the following fields:

   - SpecialCol.
   - Function
   - Column 1

   In the Column 1 field, enter the name of the column to be used to partition the table.

   Enter values for the SpecialCol and Function fields if you are partitioning the table using PARTITION_COLUMN. For information on specifying values for these fields, see *Example of Partitioning the S_ADDR_ORG Table* .
6. Save the values you entered by stepping off the cell but keep your cursor in the same row.
7. Click the Show Partition Parts button.
8. Enter LimitKey values for each partition you created for the table. As you step off each cell, the value is saved.
9. Select the Structures tab, and then the Tablespace tab, and locate the table space associated with the table you have partitioned.

   The table space is automatically marked as partitioned.

10. Select the table space and click the Show TSPARTS button.

    You can specify values for any of the Tsparts fields or leave them empty to accept the default values. For information on the default values, see *Default Objects in Storage Control Files*.
11. Make any other edits needed to the template you are using and follow the steps under *Modifying a Storage Control File Using the Database Storage Configurator* to verify and save your changes.

Changing the Number of Table Space Partitions

The following procedure describes how to increase or decrease the number of partitions for a partitioned table in a given table space.

To change the number of table space partitions

1. Launch the Database Storage Configurator, and import the storage control file you want to amend as described in *Modifying a Storage Control File Using the Database Storage Configurator*.
2. Click Structures, and then Table, and locate the name of the table space associated with the table for which you want to change the number of partitions.
3. Click Structures, and then Tablespace, and locate the appropriate table space.
4. Change the value in the Partitions column as appropriate. Save the new value by stepping off the cell but keep your cursor in the same row.
5. Click Show TSPARTs.

   If you have changed the number of partitions to a lower value, a message appears indicating that there are additional table space partitions for this table. Review and delete the additional partitions.

   If you have increased the number of partitions for a table space, you receive a message indicating that the partitions have been created.
6. Click Structures, TBL Partition Base, and locate the table you are amending.
7. In the Partitions column, update the number of partitions to match the number of table space partitions you entered previously, then step off the cell to save the value but keep your cursor on the same row.
8. Click Show Partition Parts.

   - If you decreased the number of partitions, review and delete the additional Partition Part definitions.
   - If you increased the number of partitions, review the values in the LimitKey column, and update them appropriately.

     > **Note:** When using Excel, you must type the first quote as two single quotes. Excel saves it as a single quote when you step off the cell. If you only enter a single quote, Excel does not save it.
9. Make any other edits needed to the template you are using and follow the steps under *Modifying a Storage Control File Using the Database Storage Configurator* to verify and save your changes.

7 Installing the Siebel Database on the DB2 Host

Installing the Siebel Database on the DB2 Host

This chapter describes how to install the Siebel database on a z/OS host. It describes how to perform a standard and a custom Siebel database installation, and provides instructions for creating the storage control file for a custom installation.

This chapter includes the following topics:

About the Siebel Database and the Database Configuration Utilities

The Siebel database on the DB2 host stores the data used by Siebel CRM. Siebel Server components (particularly Application Object Managers supporting Siebel Web Clients), Siebel Developer Web Clients and Siebel Tools Clients connect directly to the Siebel database and make changes in real time.

Siebel Mobile Web Clients download a subset of the server data to use locally, and periodically synchronize with the Siebel database through the Siebel Server (Siebel Remote components) to update both the local database and the Siebel database.

The Database Configuration Utilities (formerly Siebel Database Server) comprise a set of files that you install on a Siebel Server computer. These files are accessed when you run the Database Configuration Wizard and the Siebel Upgrade

Wizard to install and configure the Siebel database (Siebel Schema and seed data) on the DB2 host, or to perform operations on the Siebel database after it is installed.

The Database Configuration Utilities must be installed in the same directory location as the Siebel Server. The Siebel Gateway can be installed in a different location on the same computer or on a different computer.

The topics in this chapter describe how to install the Siebel database using the Database Configuration Wizard and the Upgrade Wizard. It is assumed that:

- You do not have an existing Siebel database installed.
- You have already installed the Siebel Enterprise Server components (Siebel Gateway, Siebel Server, and Database Configuration Utilities) and Siebel Application Interface.
- You have configured the Siebel Gateway and the Siebel Enterprise.

After you have installed the Siebel database, you configure the Siebel Server. For information on these tasks, see *Siebel Installation Guide* . For information on starting the Database Configuration Wizard, see *Running the Database Configuration Wizard*.

> **Note:** The Database Configuration Utilities installed software has no run-time role in managing database operations for users running Siebel CRM applications.

Some of the tasks you can perform with the Database Configuration Wizard are for upgrade scenarios only and are described in *Siebel Database Upgrade Guide* and *Siebel Database Upgrade Guide for DB2 for z/OS* .

Running the Database Configuration Wizard

You run the Database Configuration Wizard to perform a number of the tasks involved in installing a standard or customized Siebel database on DB2 for z/OS, including the following:

- Installing the Siebel database
- Extracting and validating the storage control files used in performing a custom installation
- Performing post-installation tasks, such as importing the Siebel Repository, installing multilingual seed data, and importing a new language to a repository.

The Database Configuration Wizard is also used when customizing development environments to perform tasks such as migrating a Siebel Repository, or synchronizing a Siebel Repository definition with the Siebel schema.

> **Note:** Start the Siebel Gateway before running the Database Configuration Wizard. The Database Configuration Wizard runs in live mode only so you must be connected to the Siebel Gateway. For further information on Siebel Configuration Wizard running modes, see *Siebel Installation Guide* .

The following topics describe how to run the Database Configuration Wizard and Upgrade Wizard from a Windows or UNIX platform. For information on performing a specific task using the utilities, see the relevant topic in the guide.

Running the Database Configuration Wizard on Microsoft Windows

The procedures in this topic describe how to do the following tasks on Microsoft Windows:

- Start the Database Configuration Wizard on Windows
- Save the configuration information you enter using the Database Configuration Wizard
- Launch the Upgrade Wizard

To run the Database Configuration Wizard on Microsoft Windows

1. From the Start menu, navigate to All Programs, Siebel Enterprise Server Configuration, and then Database Server Configuration.
2. The first Database Configuration Wizard screen appears. Enter the information you are prompted for in this screen and click Next to continue.
3. Enter the information you are prompted for in all subsequent screens. Use the Next and Back buttons to navigate to the next or previous screens.
4. When the Configuration is complete screen appears, select one of the following options, and click Next:
   - **Yes apply configuration changes now.** The configuration information you entered is saved and the generated SQL is executed in the Siebel Upgrade Wizard.
   - **No I will apply configuration changes later.** The configuration information is saved but the generated SQL is not executed in the Siebel Upgrade Wizard. You can run the Upgrade Wizard manually later to execute the generated SQL.
5. On the Configuration Parameter Review screen, review all the configuration values you have entered in the Database Configuration Wizard screens. To change any of the values, click Back to return to the screen with the parameter you need to change. If the values are correct, click Next to continue.
6. Depending on the option you selected in Step 4, one of the following occurs:
   - If you selected the option, Yes apply configuration changes now, click OK and the Siebel Upgrade Wizard is launched. Click OK to proceed or click Cancel to cancel the Upgrade Wizard.

     If you click OK, the Upgrade Wizard runs the SQL scripts to perform the task you selected when you ran the Database Configuration Wizard. After the Siebel Upgrade Wizard has finished running, click Exit to exit the Database Configuration Wizard.
   - If you selected the option, No I will apply configuration changes later, a message appears saying that the Siebel Configuration Wizard completed successfully. Click OK to finish.

     The configuration information you entered in the Database Configuration Wizard is saved in a master file located in `SIEBSRVR_ROOT\bin` but the Upgrade Wizard is not launched. You can run the Upgrade Wizard later by entering the following command from the command line:

     ```
     SIEBSRVR_ROOT\bin\siebupg /m master_option_mf.ucf
     ```

     where:

     - `master_option_mf` is the name of the master file containing the configuration information you entered.
     - option is the name of the Siebel database operation you selected.

For example, enter the following command if you had selected the Install Database option:

```
SIEBSRVR_ROOT\bin\siebupg /m master_install_mf.ucf
```

This command is generated dynamically; the path name varies based on the location of your Siebel Server installation.

Note: You can also launch the Database Configuration Wizard from the command line. For information on running the Database Configuration Wizard in GUI or command line mode, see *Siebel Installation Guide* .

Running the Database Configuration Wizard on UNIX

The procedures in this topic describe how to do the following tasks on UNIX:

- Start the Database Configuration Wizard on UNIX
- Save the configuration information you enter using the Database Configuration Wizard
- Launch the Upgrade Wizard

To run the Database Configuration Wizard on UNIX

1. Before running the Database Configuration Wizard on UNIX, you must configure your environment by running the CreateDbSrvrEnvScript script. To run CreateDbSrvrEnvScript:
   a. Navigate to the SIEBSRVR_ROOT/ `install_script/install` directory.
   b. Run CreateDbSrvrEnvScript as follows:

   ```
   CreateDbSrvrEnvScript SIEBEL_ROOT
    LANG
    Database_Platform
   ```

   where:

   - SIEBEL_ROOT is the top-level Siebel Enterprise Server installation directory, for example, `/siebel/ses`
   - LANG is the three-letter code for the primary language of your Siebel database, for example,`ENU`
   - Database_Platform is Db2.390

   CreateDbSrvrEnvScript creates two environment setup scripts, dbenv.sh and dbenv.csh, in the `SIEBSRVR_ROOT/bin` directory.
2. Navigate to the `SIEBSRVR_ROOT/bin`directory, and source either the dbenv.sh or the dbenv.csh file, according to the type of shell you use:
   - **Korn or Bourne shell**

     ```
     . ./dbenv.sh
     ```

     Tip: Make sure there is a space between the initial period and `./dbenv.sh`.

- **C shell**

  ```
  source dbenv.csh
  ```

3. Review the values of the $SIEBEL_ROOT and LANGUAGE environment variables and verify that they are correct:
   - Set the LANGUAGE variable to the language in which the Database Configuration Wizard prompts appear, for example, ENU for U.S. English.
   - Set the $SIEBEL_ROOT variable to the path of your Siebel Server installation directory, for example, `/siebel/ses/siebsrvr`.
4. Navigate to the config subdirectory of the SIEBEL_ROOT directory. For example, navigate to a location like `/siebel/ses/config`.
5. Start the Database Configuration Wizard by running the following command:

   ```
   SIEBEL_ROOT/config/config -mode dbsrvr
   ```

   For a description of the command line syntax and options, see *Siebel Installation Guide* .
6. The first Database Configuration Wizard screen appears. Enter the information you are prompted for in this screen, and click Next to continue.
7. Enter the information you are prompted for in all subsequent screens. Use the Next and Back buttons to navigate to the next or previous screens.
8. After you have entered all the requested information, the following message is displayed:

   ```
   Configuration is complete: your output will be saved under $SiebelRoot/siebsrvr/bin/
   master_<process>_mf.ucf. Would you like to deploy the process you configured to the
   database now or later?
   ```

   Choose one of the following options, then click Next:
   - **Yes apply configuration changes now.** The configuration information you entered is saved and the generated SQL is executed in the Siebel Upgrade Wizard.
   - **No I will apply configuration changes later.** The configuration information is saved but the generated SQL is not executed in the Siebel Upgrade Wizard. You can run the Upgrade Wizard manually later to execute the generated SQL.
9. The utility displays the Parameter Review screen listing all the values you have entered. To amend any of the configuration values, click Back to return to the appropriate screen and make changes. Otherwise, click Next.
10. Depending on the option you selected in Step 8, one of the following occurs:
    - If you selected the option, Yes apply configuration changes now, click OK and the Siebel Upgrade Wizard is launched. Click OK to proceed or click Cancel to cancel the Upgrade Wizard.

      If you click OK, the Upgrade Wizard runs the SQL scripts to perform the task you selected when you ran the Database Configuration Wizard. After the Siebel Upgrade Wizard has finished running, click Exit to exit the Database Configuration Wizard.
    - If you selected the option, No I will apply configuration changes later, a message appears saying that the Siebel Configuration Wizard completed successfully. Click OK to finish.

      The configuration information you entered in the Database Configuration Wizard is saved in a master file located in SIEBSRVR_ROOT/ `bin` but the Upgrade Wizard is not launched. You can run the Upgrade Wizard later by entering the following command from the command line:

      ```
      srvrupgwiz /m master_process_mf.ucf
      ```

where `master_` process `.ucf` is the name of the master file containing the configuration information you entered and process is the name of the Siebel database operation you selected.

For example, enter the following command if you had selected the Install Database option:

```
srvrupgwiz /m master_install_mf.ucf
```

The Upgrade Wizard runs the SQL scripts to perform the task you selected when you ran the Database Configuration Wizard.

Roadmap for Installing the Siebel Database

To install the Siebel database on the z/OS host, you must perform the following tasks in the order shown:

1. Fill out your copy of the worksheet in *Deployment Planning Worksheet* with the parameter values that you need to perform the database installation and configuration.
2. Perform the prerequisite installation tasks described in *Required Tasks before the Siebel Database Installation*.
3. Review information about installing the tables, indexes, and seed data using either the standard or custom option as described in *About the Database Installation Option*.
4. Install the Siebel Schema:
    - If installing in standard mode, do the following:
        - Review the information in *About Standard Installations*.
        - Perform the steps in *Performing a Standard Installation*.
    - If installing in custom mode using a storage control file you have created, perform the tasks described in *Process of Performing a Custom Installation*.
5. Review the log files for any errors as described in *Reviewing the Log Files for Errors*.
6. Import the Siebel Repository into the Siebel database and complete post-installation tasks as described in *Importing the Repository and Performing Postinstallation Tasks*.

When the Siebel database is successfully installed and you have imported the Siebel Repository, you can configure the Siebel Server. For information on this task, see *Siebel Installation Guide* .

Required Tasks before the Siebel Database Installation

Before installing and configuring the Siebel database on the z/OS host, make sure that you have the following resources available to you and have completed the following tasks.

These tasks constitute a step in *Roadmap for Installing the Siebel Database*.

- You have reviewed *Siebel Installation Guide* and have installed the following:
    - Siebel Gateway
    - Siebel Server
    - Database Configuration Utilities software on a Siebel Server computer, in the same top-level installation directory as the Siebel Server

- You have configured the Siebel Gateway and the Siebel Enterprise.
- You have designated a Siebel Server to act as a client to DB2. This server is where you installed the Database Configuration Utilities.

  > **Note:** Verify that all the Siebel-supported languages you need are installed on the DB2-client Siebel Server. Otherwise, the installation program will not recognize the additional languages when you add them to the Siebel Schema.

- You have installed DB2 Connect and tested ODBC connectivity between the computer on which the Siebel Server is installed and DB2, as described in *Preparing for Implemention on the DB2 Host*.
- A security administrator (or someone with a logon that can create user IDs and security groups) is available to support your installation.
  - This administrator must have created the user ID that will act as the database user (for example, SADMIN). The database user must be a member of the security group with authorization to set the current SQLID to the schema name.
  - The security administrator must also create the groups needed for general Siebel application users (for example, SSEROLE).

  For more details on authorization requirements, see *Security Concepts for a DB2 for z/OS Environment*.
- A database user (for example, SADMIN) with DBA or CREATEDBA authority exists. This user must also have TSO logon capability for the manual installation process.
- Your system administrator (SYSADM) and database administrator (DBA) have set up the DB2 subsystem on the z/OS host.

  For information on this task, see *Preparing for Implemention on the DB2 Host*.
- The SYSADM has created storage groups and activated and granted use of buffer pools in preparation for installation.
- The Database Owner is a valid authorization ID.
- (Optional) You have installed Microsoft Excel on the client computer to allow use of the Database Storage Configurator tool (dbconf.xls).

About the Database Installation Option

This topic constitutes a step in *Roadmap for Installing the Siebel Database*.

When you choose the Install Siebel Database option from the Database Configuration Wizard, the utility performs several tasks within the database instance you created for your Siebel application:

- It creates Siebel tables and indexes in a specified database storage layout
- It installs Siebel triggers (optional) and views, packages, and procedures
- It installs Siebel seed data specific to your database

You can choose to run these operations directly against the database, or you can apply the install DDL manually on the z/OS host. You can also choose to perform a standard or custom Siebel database installation. This topic describes these installation options.

About Database Installation Modes

You can create the Siebel Schema using one of two modes: standard or customized.

- **Standard Install**

  If you choose the standard installation option, you perform all installation tasks using the Database Configuration Wizard. A standard install creates a Siebel preconfigured database layout.

- **Customized Install**

  If you choose the custom installation option, you provide a storage control file that specifies the custom database layout that is most suitable for your Siebel deployment. You must create and validate the custom storage control file before performing the Siebel Schema installation using the Database Configuration Wizard.

About Database Installation Mechanisms

You can choose whether the DDL used to create the Siebel Schema is automatically applied against the DB2 host or is applied manually. You can execute the DDL directly against the database by choosing the Run DDL Automatically option of the Database Configuration Wizard. Alternatively, by choosing the Generate DDL Into Files option, you can assign the DDL to an output file for later transfer to the DB2 host (using Oracle-provided scripts, your own FTP, or another file transport program) where the DBA applies the DDL using customary tools.

> **Note:** The Upgrade Wizard is the Siebel utility that performs Siebel database installation, upgrade, and database manipulations. The utility is launched by the Database Configuration Wizard. In Windows, the Siebel Upgrade Wizard is designated as siebupg.exe. In UNIX, it is designated as srvrupgwiz and is located in the SIEBSRVR_ROOT/ `bin` directory.

Installing a New Database Language

If you want to add a language to an existing Siebel database, you can do so by selecting the Install Siebel Database option on the Database Configuration Wizard, and then selecting the Add a language to an existing database option. Adding a language to an existing Siebel database installs seed data to support that language. For a description of this process, see *Installing Multilingual Seed Data*.

About Standard Installations

This task is a step in *Roadmap for Installing the Siebel Database*.

You can use the standard install option as your sole installation procedure, or you can perform the standard installation as the first step of a custom installation, and then customize your deployment after you have completed the standard installation. To perform a custom installation, see *Process of Performing a Custom Installation*.

The standard installation provides two schema layout options: Siebel Schema without Partitioning and Siebel Schema with Partitioning. As shown in the following table, the storage control file you use during the standard installation is

determined by your preferred Siebel Schema layout and your encoding scheme. For additional information on the templates, see *About Storage Control File Templates*.

Siebel Schema Layout	Storage Control File Template to Use	Storage Control File Description
Siebel Schema without Partitioning	storage_np.ctl	Contains a database storage layout for a Siebel Schema that does not use Unicode encoding. No partitioned table spaces are provided in this storage control file.
	storage_np_u.ctl	Contains a database storage layout for a Siebel Schema with Unicode encoding. No partitioned table spaces are provided in this storage control file.
Siebel Schema with Partitioning	storage_p.ctl	Contains a partitioning scheme for a set of tables for ASCII.
	storage_p_u.ctl	Contains a partitioning scheme for a set of tables for Unicode.
	storage_p_e.ctl	Contains a partitioning scheme for a set of tables for EBCDIC.

If you choose the Siebel partitioned layout, a selected subset of Siebel tables is defined as partitioned. For a discussion of partitioned table spaces, see *About Siebel Table Partitioning*.

Performing a Standard Installation

This task is a step in *Roadmap for Installing the Siebel Database*.

This topic describes how to perform a standard Siebel database installation using the Database Configuration Wizard.

When you select the Install Database option on the Database Configuration Wizard, you can choose the installation mechanism to use: either Run DDL Automatically or Generate DDL Into Files.

If you select the Run DDL Automatically option, performing the steps in the following procedure completes a standard Siebel database installation.

If you select the Generate DDL Into Files option, after performing the procedure in this topic, you must also complete the tasks described in *Completing the Siebel Schema Installation Using Generated DDL* to complete your standard Siebel database installation.

To perform a standard installation

1. Start the Siebel Gateway.
2. Start the Database Configuration Wizard using one of the following methods:
   - *Running the Database Configuration Wizard on Microsoft Windows*
   - *Running the Database Configuration Wizard on UNIX*
3. On the Siebel Server Directory screen, enter the directory path of the Siebel Server that the Siebel database will connect to. Do one of the following:

- Accept the default value displayed in the Siebel Server Directory field. (This value is the SIEBSRVR_ROOT directory, for example, `C:\siebel\ses\siebsrvr` (Windows) or `/siebel/ses/siebsrvr` (UNIX).
- Use the Browse button to select the directory path where the Siebel Server is installed if it was not installed in the default directory.

To continue, click Next.

4. On the Siebel Database Server Directory screen, enter the directory path to the Database Configuration Utilities software. Do one of the following:
   - Accept the default path displayed in the Siebel Database Server Directory field. (This path is the DBSRVR_ROOT directory, for example, `C:\siebel\ses\dbsrvr` (Windows) or `/siebel/ses/dbsrvr` (UNIX).
   - Select Browse to select the directory path where the Database Configuration Utilities were installed if they were not installed in the default directory.

   To continue, click Next.
5. On the database platform screen, select your database platform, in this case, IBM DB2 UDB for z/OS, and then click Next.
6. On the Siebel Database operation screen, select the Install Database option, and click Next.
7. On the Select Installation Operation screen, select the Install Siebel Database option to install the Siebel Database Schema and populate it with seed data.

   > **Note:** If you have already installed your Siebel Database Schema in its base or primary language, and want to install a new language, select the option, Add a language to an existing Siebel Database. For further instructions, see *Installing Multilingual Seed Data*.

8. Confirm that you want to install a new Siebel database, and click Next.
9. Identify the appropriate database encoding method:
   - **UNICODE Database.** If you select this option, click next and proceed to Step 11.
   - **Non-UNICODE Database.** If you select this option, click next and proceed to Step 10.

     > **CAUTION:** Choose the correct option for your database to prevent installing the wrong data types. The database will not be able to create Unicode data types on a non-Unicode page setting, so check this setting carefully before choosing the option.

10. If you previously indicated that your database is non-Unicode, the Database Encoding screen is displayed allowing you to indicate whether your DB2 subsystem is ASCII or EBCDIC. For more information about choosing the code page for your subsystem, see *About Setting Up the DB2 Subsystem*.

    Select the appropriate option, and click Next.
11. If you have installed more than one language pack onto the Siebel Server, the Base Language screen is displayed. Specify which language is the primary (base) language for the Siebel database, and click Next.
12. Enter the following values for your Siebel database:
    - **ODBC Data Source Name.** The Siebel Server installation process automatically creates the data source, using the format EnterpriseName_ DSN, for example, SBA_82_DSN or Siebel_DSN. Use the default name of the ODBC data source or enter the database alias you prefer to use for the data source.

      > **Note:** Database aliases must be registered within DB2 Connect. If you define database aliases, you must add the `TXNISOLATION` parameter to the database alias entry in the db2cli.ini file and set it to have a value of 1. For information on setting values in the db2cli.ini file, refer to the relevant IBM documentation.

- **Source Database DB2 Subsystem Name.** Enter the name of the DB2 database instance into which you will install the Siebel Schema, for example, Q20K.

To continue, click Next.

13. On the Siebel Database User Name screen, enter a value for the Database User Name field.

    Type the ID (for example, SADMIN) used to log into the Siebel database. This user ID must be part of the security group (secondary authorization group) with authorization to set the current SQLID to the schema name.
14. On the Siebel Database Password screen, type the password for the ID used to log into the Siebel database. Retype the password to confirm it, then click Next.
15. On the Siebel Schema Qualifier screen, enter the character ID that identifies the Siebel Schema owner, for example, SIEBTO. Click Next to continue.

    > **Note:** This ID can be up to eight characters in length, must be in uppercase, must start with a letter, and cannot include any special characters. This value is the SQL Authorization ID stored in the CURRENT SQLID special register.

16. On the Security Group ID/Grantee screen, type the group ID for which schema access is being granted, for example, SSEROLE.

    For more information about the group authorization ID privileges, see *Security Concepts for a DB2 for z/OS Environment*.
17. On the Select Installation Type screen, choose one of the following install options:
    - **Standard Install.** The standard installation option installs the database, using one of the Siebel-recommended default storage layouts.
    - **Customized Install.** For a customized installation, you provide a custom storage control file, which you have created and validated prior to performing this installation. This option allows you to create the database layout that is best suited to your Siebel deployment. For information on the remaining steps in the custom installation process, see *Process of Performing a Custom Installation*.

      Select the Standard Install option, click Next, and proceed to Step 18.
18. In the Siebel Schema Layout screen, select one of the following layout options, then click Next:
    - **Siebel Schema without Partitioning.** Choose this option if you want all tables in segmented table spaces.
    - **Siebel Schema with Partitioning.** This layout includes a set of tables that are recommended for partitioning. The remaining nonpartitioned tables are in segmented table spaces. For more information on this option, see *About Siebel Table Partitioning*. For details about layout considerations, see *Configuring the Siebel Database Layout*.
19. On the Storage Group for Table Spaces screen, enter the storage group name for table spaces, for example, SYSDEFLT, then click Next.
20. Enter the storage group name for indexes, for example, SYSDEFLT, then click Next.
21. The next four screens allow you to enter default names for the buffer pools assigned to table spaces. After entering a value on a screen, click Next to continue.

    > **Note:** Make sure that the buffer pools have previously been activated and that the DBA has granted access to them. For information about layout considerations, see *Configuring the Siebel Database Layout*.

    - **4KB Buffer Pool Name.** Type the 4-KB buffer pool name for your table spaces, or accept the default, BP1.
    - **8KB Buffer Pool Name.** Type the 8-KB buffer pool name for your table spaces, or accept the default, BP8K1.

- **16KB Buffer Pool Name.** Type the 16-KB buffer pool name for your table spaces, or accept the default, BP16K1.
- **32KB Buffer Pool Name.** Enter the 32-KB buffer pool name for your table spaces, or accept the default, BP32K1.

22. Enter the name of a 4 KB buffer pool assigned to indexes, or accept the default name, BP2. This buffer pool must be activated and the DBA must have granted access to it. Click Next.
23. **Database Name Prefix.** Type the prefix to assign to Siebel database names. The default value is SIDB. The prefix can consist of up to four characters, it must start with a letter, and it cannot contain any special characters.

    > **Note:** The database name prefix must be the same for all database objects in the Siebel schema because the prefix identifies an object as belonging to the Siebel schema. Siebel utilities can recognize and use Siebel objects only if they follow Siebel naming conventions.

24. On the Select Installation Mechanism screen, indicate which of the following installation mechanisms you want to use:
    - **Generate DDL Into Files.** Select this option to generate the DDL required to create the Siebel Schema into files. You must then transfer these files to the z/OS host where they can be applied by your DBA using customary methods, for example, SPUFI or DSNTEP2.
      If you select this option, click Next and proceed to Step 25.
    - **Run DDL Automatically.** Select this option to apply the DDL and DML required to create the Siebel Schema directly against the database, using ODBC to connect.
      If you select this option, click Next and then proceed to Step 27.
25. On the DDL Commit Frequency screen, select the number of DDL statements to be run before a COMMIT statement is issued. Click Next.
26. On the Output Directory screen, specify the name of the directory where you want to save the Siebel Schema DDL files generated. Use lowercase characters for directory and path names unless directed otherwise. The default directory is DBSRVR_ROOT `\db2390\dboutput\install` (Windows) or DBSRVR_ROOT `/db2390/dboutput/install` (UNIX).
27. On the Log Output Directory screen, accept the default log directory or enter a new directory name, and click Next.
    By default, the files are created in SIEBSRVR_ROOT `\log\install_mf` (Windows) or SIEBSRVR_ROOT `/log/install_mf` (UNIX).
28. Save the configuration information you have entered and launch the Siebel Upgrade wizard as described in the following relevant topic:
    - *Running the Database Configuration Wizard on Microsoft Windows*
    - *Running the Database Configuration Wizard on UNIX*
29. Depending on whether you chose to Run DDL automatically or Generate DDL into files, do one of the following:
    - If you selected the installation option Run DDL Automatically, the Siebel Upgrade Wizard applies the DDL commands to create the Siebel Schema directly against the database on the z/OS host. When you receive a message stating that the configuration was applied successfully, click OK. This step completes the Run DDL Automatically installation option.
    - If you selected the installation option Generate DDL Into Files, the Siebel Upgrade Wizard is launched. Click OK and the Wizard generates the following files into the output directory that you designated in Step 26:
      - schema.sql and ddlview.sql files. These contain the DDL to create the Siebel tables and indexes.

- instftp.bat. Use this batch file to run the Siebel-provided scripts, your own FTP, or a similar file transfer program to transport the DDL (schema.sql and ddlview.sql) to the DB2 host.
- instftp.txt. Use this file to transfer files to the mainframe.
- jobinstl.txt. Contains JCL and REXX execs to execute the DDL.

30. To complete the installation process, you must now perform the steps in *Completing the Siebel Schema Installation Using Generated DDL* to apply the schema.sql and ddlview.sql files on the DB2 host.

Completing the Siebel Schema Installation Using Generated DDL

If you selected the installation option Generate DDL Into Files, the Siebel Upgrade Wizard generates the schema.sql and ddlview.sql files that contain the DDL to create the Siebel Schema. You must apply these files on the z/OS host, then return to your Windows or UNIX Siebel Server computer and restart the Upgrade Wizard to generate the additional files required to complete the database installation. These tasks are described in this topic.

To complete the Siebel Schema installation using generated DDL

1. After the Siebel Upgrade Wizard generates the schema.sql and ddlview.sql files that contain the DDL to create the Siebel Schema, the following message appears:

   ```
   Pause #1: Please create Siebel Schema using schema.sql and ddlview.sql located in
   the Output Directory. Once the objects are created, please select Yes. To stop
   now and continue installation later please select No. (To resume, please start
   the Upgrade Wizard from the command line using option /m master_install_mf.ucf).
   ```

   Click No (Windows) or enter N (UNIX) to quit the Siebel Upgrade Wizard.
2. Transfer the schema.sql and ddlview.sql files to z/OS, and have your DBA apply them to create the Siebel tables and indexes. For information on these tasks, see *Applying Installation DDL on the DB2 Host*.

   The GRANT VIEW statements in ddlview.sql might cause SQL errors to be generated. For information on using GRANT VIEW statements with DB2 for z/OS, see *Granting Authorization to Views in DB2*.
3. Now that you have created the Siebel Schema on the DB2 host, restart the Siebel Upgrade Wizard from the command line as described in the relevant topic:
   - *Running the Database Configuration Wizard on Microsoft Windows*
   - *Running the Database Configuration Wizard on UNIX*
4. The Upgrade Wizard validates the schema, and generates the files that are used to create user-defined functions (UDFs). It then displays the following message:

   ```
   Pause #2: Please inspect contents of schemvld.sql file to ensure that Siebel
   Schema was created successfully. There must be no DDL statements, only CREATE
   TRIGGER statements are allowed. If you find unacceptable statements in
   schemvld.sql file, please click "No" button, and take steps to correct the issue.
   Otherwise, click "Yes" and continue: ftp the files to install UDF functions. (To
   resume, please start the Upgrade Wizard from the command line using option /m
   ```

```
master_install_mf.ucf). (Y/N)
```

The second pause in the Siebel Schema installation process generates the schemvld.sql file. The schemvld.sql file contains SQL statements for missing Siebel Schema objects, if such objects are found. Normally, schemvld.sql contains triggers only.

5. Click No (Windows) or enter N (UNIX) then review the schemvld.sql file to determine whether or not it contains DDL statements.
   - If schemvld.sql does not contain SQL statements, this means the Siebel Schema was created successfully.
   - If the file does contain SQL statements, the Siebel Schema was not created successfully, probably because the DBA missed a step during the manual schema creation process on the DB2 host. The schemvld.sql file contains the missing objects of the schema (the Delta schema). The DBA must apply the file with the missing objects to complete a full schema creation.

     > **Note:** If you are creating a Siebel Schema with partitioning, the generated schemvld.sql file contains triggers to remove and create objects; these are acceptable and can be ignored.

6. Apply the user-defined functions on the DB2 host.

   For information on this task, see *Deploying User-Defined Functions*.
7. When the UDFs are deployed on the DB2 host, restart the Upgrade Wizard as described in the relevant topic:
   - *Running the Database Configuration Wizard on Microsoft Windows*
   - *Running the Database Configuration Wizard on UNIX*

   The Siebel Upgrade Wizard performs the final step in the installation, that is, importing Siebel seed data.

   > **Note:** This operation fails if the Siebel Schema has not been created because there is no schema to populate.

8. When you have successfully populated the schema with seed data, you are ready to carry out the final step in the Siebel Schema installation process, importing the Siebel Repository. For further information on this task, see *Importing the Siebel Repository*.

> **Tip:** To preserve a record of the values you entered during the installation, locate and print the master_install_mf.ucf file in the bin subdirectory of your Siebel Server installation directory.

Applying Installation DDL on the DB2 Host

When you install the Siebel Schema, if you select the Generate DDL Into Files option, the schema.sql and the ddlview.sql files are generated in the DDL output directory that you specified during the installation. Before continuing with the installation, you must transfer the DDL files to the DB2 host and then apply them. Both procedures are described in this topic (the instructions in this topic apply to both Windows and UNIX operating systems).

> **Note:** Ensure that the tool you use (SPUFI or DSNTEP2) to apply the DDL on the DB2 host uses a slash as the SQL delimiter. For further information, see *Setting Up the SQL Delimiter on DB2*.

Transferring the DDL Files to the DB2 Host

The following procedure describes how to transfer the DDL files to the host.

To transfer the DDL files to the DB2 host

1. Navigate to the output directory:

   Windows: `DBSRVR_ROOT\db2390\dboutput\install`

   UNIX: `DBSRVR_ROOT/db2390/dboutput/install`
2. Open the instftp.txt file.
3. Edit the instftp.txt file using the following information:
   a. Change &IP into the IP address or domain name of your DB2 host, for example ZM01.
   b. Change &Username to your user name on the z/OS host, for example SADMIN.
   c. Change all occurrences of SIEBELQ1 to your own high-level qualifier (HLQ), for example, SADMIN. Save the file. The following is an example of the instftp.txt file.

```
open &IP user &Username
quote site cylinders primary=1 secondary=1
quote site recfm=fb lrecl=80 blksize=0
send &directoryPath1/jobinstl.txt 'SIEBELQ1.SIEBEL.INST.JOBINSTL'
send &directoryPath1/ddlview.sql 'SIEBELQ1.SIEBEL.INST.DDLVIEW'
quote site cylinders primary=10 secondary=25
send &directoryPath1/schema.sql 'SIEBELQ1.SIEBEL.INST.SCHEMA'
Quit
```

   Note: The &directoryPath1 variable is substituted with the appropriate UNIX or Windows path.
4. Save the changes you have made to instftp.txt.
5. Double-click instftp.bat from Windows or issue the following command from UNIX:

```
ftp -vn <instftp.txt> instftp.log
```

6. Enter the password associated with the user name you entered in the instftp.txt file in Step 2.

   Press any key to continue.
7. The files are sent to the DB2 host and the log file instftp.log is created in your DBSRVR_ROOT `\db2390\dboutput\install` directory (Windows) or your DBSRVR_ROOT / `db2390/dboutput/install` (UNIX) directory.

The log contains information on the file transfer, for example:

```
Connected to ZM01.siebel.com.

220-FTPD1 IBM FTP CS V1R4 at LOOPBACK, 22:44:26 on 2011-02-27.
220 Connection will close if idle for more than 5 minutes.
331 Send password please.
230 SADMIN is logged on. Working directory is SADMIN.
200 SITE command was accepted
200 SITE command was accepted
200 Port request OK.
125 Storing data set SADMIN.SIEBEL.INST.JOBINSTL
250 Transfer completed successfully.
ftp: 45038 bytes sent in 0.00Seconds 45038000.00Kbytes/sec.
200 Port request OK.
125 Storing data set SADMIN.SIEBEL.INST.DDLVIEW
250 Transfer completed successfully.
ftp: 1850 bytes sent in 0.00Seconds 1850000.00Kbytes/sec.
200 SITE command was accepted
200 Port request OK.
125 Storing data set SADMIN.SIEBEL.INST.SCHEMA
250 Transfer completed successfully.
ftp: 8369624 bytes sent in 2.42Seconds 3452.82Kbytes/sec.
```

```
221 Quit command received. Goodbye.
```

You can now apply the DDL on the DB2 host.

Applying the DDL Files on the DB2 Host

The following procedure describes how to apply the DDL files on the host.

To apply the DDL files on the DB2 host

1. When the DDL file transfer is successfully completed, log on to the mainframe and navigate to the Data Set List Utility (DSLIST) panel.

   The `DSLIST` panel reflects the data sets associated with the user ID you entered previously when logging on.
2. Edit the HLQ.`SIEBEL.INST.JOBINSTL` data set, for example, `SIEBLQ1.SIEBEL.INST.JOBINSTL`, by entering e before the data set name and pressing Enter twice.

   The Edit Entry Panel appears.
3. As instructed in the comments section, change all occurrences of `SIEBELQ1` to your HLQ using the following command:

   ```
   c SIEBELQ1 HLQ ALL
   ```

   > **Note:** Make sure that your entry is in capital letters.

4. Press Enter, and the changes are made.
5. Enter`submit` on the Command line, and press Enter to submit the job.
6. Details of the job are displayed. Press Enter again and a new data set named HLQ `.SIEBEL.INST.EXEC` is created.
7. Press F3 to return to the DSLIST screen.
8. On the `DSLIST` screen, edit the HLQ `.SIEBEL.INST.EXEC` data set by entering e before the data set name and pressing Enter.
9. Enter `exec` before the SBLINST member and press Enter to start the installation.
10. On the Siebel Install Main Menu screen, you can select one of the following options:
    - 1. Generate Siebel DDL - Generate JCL and DDL
    - 2. Generate View DDL - Generate VIEW DDL

    Enter `1` on the Command line to create the JCL and DDL used to generate the schema.
11. Enter the DDL input data set name, for example, `HLQ .SIEBEL.INST.SCHEMA`, and then press Enter.
12. Press Enter again and an SQL data set named `HLQ .SIEBEL.INST.SCHEMA.SCH` is created.
13. Press F3 to exit after the job completes.
14. On the `DSLIST` screen, enter `e` before the `HLQ .SIEBEL.INST.SCHEMA.SCH` data set, and press Enter.

    > **Note:** If you use a tool other than the DSNTEP2 tool to create objects, specify the program and plan names of the tool you are using in the SIEBSQLA member, for example:
    >
    > ```
    > DSN SYSTEM(Q109)
    > RUN PROGRAM(MYTEP2) PLAN(MYTEP2) PARMS(' /SQLTERM(/) ')
    > END
    > ```

15. In the `HLQ .SIEBEL.INST.SCHEMA.SCH` data set, edit the following JCL members in the order shown. Run the GRJCL member last because it grants permissions to the objects (database, table spaces and tables) created when you run the other three members.
    - DBJCL

- TSJCL
- TBJCL
- GRJCL

To edit and run each JCL member, do the following:

a. Enter e before the JCL member, and press Enter.
b. Edit the job card parameters in each member, for example, specify the appropriate CLASS and MSGCLASS names:

```
// D2PUNL0 JOB ACCNT#,CLASS=S,MSGCLASS=X,MSGLEVEL=(1,1),
// TIME=1440,REGION=0M,NOTIFY=&SYSUID,LINES=9999
//*TYPRUN=SCAN
```

c. Enter an asterisk (*) before the TYPRUN parameter to comment out this command.
d. Enter `submit` on the Command line and press Enter to submit the job.

   After each job is completed, verify that the return code is `0` (zero) to make sure that no errors occurred and that the job ran successfully.

   The Siebel DDL is now generated.

16. To generate the View DDL file, follow the instruction to edit the HLQ `.SIEBEL.INST.EXEC` data set described in Step 8.
17. Select 2 on the Install Main Menu screen to generate the View DDL.
18. Enter the DDL file name, for example, `HLQ .SIEBEL.INST.DDLVIEW`, to specify the DDL input data set, then press Enter.
19. Press Enter to generate a DDLVIEW SQL data set named `HLQ .SIEBEL.INST.DDLVIEW.VIE`.
20. Press F3 to exit after the job completes.
21. On the `DSLIST` screen, enter `e` before `HLQ .SIEBEL.INST.DDLVIEW.VIE` data set, and press Enter.

> **Note:** If you use a tool to create objects other than the DSNTEP2 tool, specify the program and plan names of the tool you are using in the SIEBSQLA member, for example:

```
DSN SYSTEM(Q109) RUN PROGRAM( MYTEP2 ) PLAN( MYTEP2 ) PARMS(' /SQLTERM(/) ') END
```

22. In the `HLQ .SIEBEL.INST.DDLVIEW.VIE` data set, edit the VIEJCL member:

a. Enter e at the start of the VIEJCL member, and press Enter.
b. Edit job card parameters in each member, for example, specify the appropriate CLASS and MSGCLASS names:

```
// D2PUNL0 JOB ACCNT#,CLASS=S,MSGCLASS=X,MSGLEVEL=(1,1),
// TIME=1440,REGION=0M,NOTIFY=&SYSUID,LINES=9999
//* TYPRUN=SCAN
```

c. Enter an asterisk (*) before TYPRUN parameter to comment out this statement.
d. Enter an asterisk (*) before GRANT statements; GRANT statements on views are not required with DB2 for z/OS. If you do not comment out these statements, error messages are returned when you run the VIEJCL member; you can ignore these messages.
e. Enter `submit` on the Command line and press Enter to submit the job. Press Enter again.

   After each job is completed, verify that the return code is `0` (zero), to make sure that no errors occurred and that the job ran successfully.

23. Return to the midtier to continue the Siebel Schema installation by verifying the schema that you have just created.

    The schema.sql file is overwritten during the schema validation process. Therefore, if you would like to save the schema.sql file with the complete Siebel Schema DDL that was generated for the Siebel Schema, rename schema.sql (located in the DBSRVR_ROOT `\db2390\dboutput\install` directory on Windows or DBSRVR_ROOT `/db2390/dboutput/install` directory on UNIX) after you apply schema.sql on the mainframe.

Setting Up the SQL Delimiter on DB2

Because the file schema.sql uses a slash as a delimiter, you must set up the slash as the delimiter in the tool (SPUFI or DSNTEP2) you are using to apply the DDL. Siebel CRM provides files with slashes for delimiters because schema.sql might contain triggers for table partitioning that use semicolons. For example:

```
CREATE TRIGGER Q202999.PTH0393 NO CASCADE BEFORE INSERT ON Q202999.S_ACT_EMP
REFERENCING NEW AS N FOR EACH ROW MODE DB2SQL WHEN (N.EMP_ID IS NOT NULL) BEGIN
ATOMIC SET N.PARTITION_COLUMN=RIGHT(N.EMP_ID, 2); END /
```

Note: If you are not using table space partitioning, you can replace slashes with semicolons.

DSNTEP2

If you run the DDL in batch mode, use the DSNTEP2 tool. Perform the following procedure to set the delimiter in DSNTEP2.

To set the delimiter in DSNTEP2

- Use the `SQLTERM(/)` parameter to set the delimiter to a slash mark. The following example illustrates how this appears in the `SYSTSIN DD` statement.

  Note: In the following example, `Q202` is the subsystem name.

  ```
  //SYSTSIN DD * DSN SYSTEM(Q202) RUN PROGRAM(DSNTEP2) PLAN(DSNTEP71) PARMS('/SQLTERM(/)') -
   LIB('DSN710.RUNLIB.LOAD') END
  ```

SPUFI

If you use SPUFI, perform the following procedure to set the delimiter.

To set the SQL delimiter in SPUFI

1. Set CHANGE DEFAULTS to YES.
2. When the CURRENT SPUFI DEFAULTS panel appears, change the SQL TERMINATOR to a slash mark.

If you are not using triggers, you can change the slashes back to semicolons. If you use the TSO editor, do this by entering the following:

```
CHANGE X'61' X'5E' ALL
```

Deploying User-Defined Functions

Two user-defined functions (UDFs) are provided in the Siebel application: nextseq and exrate. The nextseq() UDF generates sequential values for Enterprise Integration Manager (EIM) export and the exrate() UDF is used for currency aggregation functions. After the schema is validated, you must install the UDFs using the following instructions. These instructions are applicable to both Windows and UNIX operating systems.

Note: You must have Workload Manager (WLM) installed and configured to run the nextseq and exrate user-defined functions.

To deploy user-defined functions

1. Navigate to the output directory (DBSRVR_ROOT `\db2390\dboutput\install` on Windows or DBSRVR_ROOT `/db2390/dboutput/install` on UNIX) and modify the udfftp.txt FTP file as follows:
   a. Change the &IP address to the IP address or domain name of your mainframe.
   b. Change the &username to a valid TSO user ID.
   c. Change the SIEBELQ1 high level DSN qualifier to your DSN qualifier.
   d. Save the changes to the file.
2. Double-click DBSRVR_ROOT `\db2390\dboutput\install`\udfftp.bat (Windows) to execute the batch FTP script to send files to the mainframe, or issue the following command from UNIX:

   ```
   ftp -vn <udfftp.txt> udfftp.log
   ```

   The script sends the following files to the mainframe from the DBSRVR_ROOT `\db2390\storproc\zos` directory on Windows or the DBSRVR_ROOT `/db2390/storproc/zos` directory on UNIX:
   - SIEBEL.UDF.RECEIVE.JCL
   - SIEBEL.UDF.LOAD.XMIT
   - SIEBEL.UDF.CNTL.XMIT
   - SIEBEL.UDF.DDL.XMIT
   - SIEBEL.UDF.DBRM.XMIT
3. On the mainframe, navigate to the USER_ID `.UDF.RECEIVE.JCL` data set and edit the data set by typing `e` beside it, then press Enter.
4. Press Enter again.
5. Modify the job card parameters according to the instructions included in the JCL member.
6. To submit the job, type submit on the command line, and press Enter. The files you sent to the mainframe in Step 2 are received in a readable format.
7. Edit the HLQ `.SIEBEL.UDF.DDL` file by typing `e` before the data set name, and pressing Enter.
   a. Edit EXRATE by entering **e** before the function name and pressing Enter. Set the Collid value in the EXRATE function to your table owner name and set the WLM environment variable to the name of your application environment.
   b. Edit NEXTSEQ by entering **e** before the function name and pressing Enter. Set the WLM environment variable to the name of your application environment.
8. Press F3 twice.
9. Edit the HLQ `.SIEBEL.UDF.CNTL` data set by typing **e** before the data set name and pressing Enter.

10. In the HLQ `.SIEBEL.UDF.CNTL` data set, edit and run the following JCL members in the order shown:
    - IEBCOPY
    - CREATUDF
    - BIND
    - WLMRFSH

    The WLMRFSH (refresh) job must be the last job run.

    Edit and submit each JCL member in turn as follows:

    a. Edit the member by entering **e** before the member name and pressing Enter.
    b. Edit the job card parameters according to the instructions in the member.
    c. Submit the job by entering `submit` on the command line, and pressing Enter.

    Make sure that the return code for each job equals 0.

Return to the midtier to continue the Siebel Schema installation by populating the schema with seed data.

Process of Performing a Custom Installation

This process is a step in *Roadmap for Installing the Siebel Database*.

You can use the Database Configuration Wizard option, Customized Install, to configure the Siebel database storage layout for the Siebel application installation.

To perform a custom installation, perform the following tasks:

- *Preparing a Storage Control File*
- *Performing a Custom Installation*

Preparing a Storage Control File

This task is a step in *Process of Performing a Custom Installation*.

When you select the standard Siebel Schema installation method, the Database Configuration Wizard creates a storage control file based on the information that you provide in response to prompts.

When you select the custom installation method, you must first create the storage control file that is used as the basis of your Siebel Schema custom installation. You can choose from the following options:

- Use an existing storage control file in the custom installation.

  You can specify the name of an existing storage control file in a custom installation.

- Modify one of the Siebel-provided storage control file templates and use it in the custom installation. You can edit the storage control file template directly, or you can use the Siebel Database Storage Configurator (dbconf.xls) to modify the template. For more information, see *Modifying a Storage Control File Using the Database Storage Configurator*.
- Extract a storage control file based on the layout of the existing Siebel Schema using the Extract Storage File option of the Database Configuration Wizard. For more information, see *Extracting a Storage Control File from the DB2 Catalog*.

Extracting a Storage Control File from the DB2 Catalog

You can create a storage control file using the Siebel extract utility. This utility extracts the storage layout of your source database from the DB2 catalog and creates a new storage control file that you can use, for example, in performing a custom installation or in cloning an existing Siebel database. You can use the extract utility at any time.

> **Note:** You must validate the storage control file after you extract it and after you modify it. For information on how to validate a storage control file, see *Validating an Extracted or Modified Storage Control File*.

Follow the instructions in the following procedure to extract a storage control file under Windows and UNIX.

To extract the storage control file

1. Launch the Database Configuration Wizard and follow the steps in *Performing a Standard Installation* until the Siebel Database Operation screen is displayed.
2. On the Siebel Database Operation screen, select Run Database Utilities, and click Next.
3. From the Database Utility Selection screen, select Configure Database, and click Next.
4. The following options are available on the Database Configuration Options screen:
   - **Extract Storage File.** Extracts a storage control file based on the existing Siebel Schema layout.
   - **Validate Storage File.** Validates that a specified template and a specified schema will work on the DB2 host.

   Select the Extract Storage File option, and click Next.
5. The following Extract options are available:
   - **Extract from Catalog.** This option extracts the storage layout of your source database from the DB2 catalog. The output is a new storage layout.
   - **Extract from Catalog and Merge with Template.** Extracts the storage layout of your source database and merges it with a Siebel template storage control file. This preserves your schema layout. If a database object is specified in both the existing database and the template, the layout of the existing database is extracted as the default. If there are conflicts that cannot be resolved, the extract process fails. You must correct any errors and run the Extract process again.

   Select the appropriate Extract option, and click Next.
6. If you have installed more than one language pack onto the Siebel Server, the Base Language screen is displayed. Specify which language is the primary (base) language for the Siebel database, and click Next.
7. On the ODBC Data Source Name screen, you can either select the default name of the ODBC data source or specify a different data source name.

   > **Note:** The Siebel Server installation process automatically creates the data source, using the format EnterpriseName_DSN.

8. On the Siebel Database User Name screen, type your source database user name, and then click Next.
9. On the Siebel Database Password screen, type the password associated with the source database user name, confirm the password by typing it again, and then click Next.
10. On the Siebel Schema Qualifier screen, type the schema qualifier for your existing database, and then click Next.

    The schema qualifier is also an authorization ID. The schema qualifier must start with a letter and cannot contain special characters.
11. Identify the database encoding scheme; either Unicode or Non-Unicode.

    If you selected the UNICODE Database option, proceed to Step 13.

    > **CAUTION:** Choose the correct option for your database to prevent installing the wrong data types. The database cannot create Unicode data types on a non-Unicode page setting, so check this setting carefully before choosing the option.

12. If you indicated that your database uses a non-Unicode encoding scheme, specify the code page encoding scheme you use, either ASCII or EBCDIC, and click Next.

    For more information on choosing the database code page, see *Considerations in Choosing the Database CCSID*.
13. On the Default Database Name screen, enter the default name for Siebel databases (for example, SIDB), and click Next.
14. On the Default Table Space screen, enter the default name for Siebel table spaces (for example, SIEBTS), and click Next.
15. On the Default Storage Group screen, enter the default storage group name for Siebel table spaces, for example, SYSDEFLT. Click Next.
16. Enter the default storage group name for Siebel indexes, for example, SYSDEFLT. Click Next.
17. On the Default Buffer Pool Name screen, specify the default name of the buffer pool assigned to table spaces or accept the default, BP1. Your DBA must have previously activated the buffer pool and granted access to it.
18. Enter the default name of the 4 KB buffer pool assigned to indexes, or accept the default name, BP2. This buffer pool must be activated and the DBA must have granted access to it. Click Next.
19. In the Storage Control File screen, type the name you want to assign to the new storage control file generated, for example DBSRVR_ROOT `\db2390\` my_storage_file.ctl (Windows) or DBSRVR_ROOT/ `db2390/` my_storage_file.ctl (UNIX). Click Next.
20. On the Log Output Directory screen, indicate the directory where you want the log files to be generated. The default directory is SIEBSRVR_ROOT `\log\dbconfig_extract_catalog_mf` (Windows) or SIEBSRVR_ROOT `/log/dbconfig_extract_catalog_mf` (UNIX).
21. Save the configuration information you have entered and launch the Siebel Upgrade wizard as described in the following topics:
    - *Running the Database Configuration Wizard on Microsoft Windows*
    - *Running the Database Configuration Wizard on UNIX*
22. Click OK.

    The Upgrade Wizard connects to the z/OS host, extracts the storage control file from the DB2 catalog, and copies it to the file you specified.
23. Click Exit to exit the Database Configuration Wizard.

    You must now validate the extracted storage control file as described in *Validating an Extracted or Modified Storage Control File*.

Validating an Extracted or Modified Storage Control File

You must validate a storage control file after you extract it or after you modify it to ensure it has been correctly generated and will work with the Siebel Schema as defined in the schema.ddl file.

To validate a storage control file

1. Launch the Database Configuration Wizard and follow the steps in *Performing a Standard Installation* until the Siebel Database Operation screen is displayed (Step 6).
2. On the Siebel Database Operation screen, select Run Database Utilities, and click Next.
3. Choose the Configure Database option, and click Next.
4. On the Database Configuration Options screen, select the Validate Storage File option, and click Next. This option validates that a specified storage control file and a specified schema will work on the DB2 host.
5. Select one of the following options:
   - **Batch - Generate Unload/Load.** Select this option to validate the storage control file that you are going to use when upgrading to a new release of Siebel CRM.
   - **Foreground - Generate Insert/Select.** Select this option to validate the storage control file that you are going to use when migrating repositories or synchronizing Siebel Repository definitions with the Siebel Schema.
6. Identify the appropriate database encoding method, either Unicode or non-Unicode, and click Next.

   > **CAUTION:** Choose the correct option for your database to prevent installing the wrong data types. The database cannot create Unicode data types on a non-Unicode page setting, so check this setting carefully before choosing the option.

   If you selected the UNICODE Database option, proceed to Step 8.
7. If you indicated that your database uses a non-Unicode encoding scheme, specify the code page encoding scheme for your Siebel database, either ASCII or EBCDIC.

   For more information on choosing the database code page, see *Considerations in Choosing the Database CCSID*.
8. If you have installed more than one language pack onto the Siebel Server, the Base Language screen is displayed. Specify which language is the primary (base) language for the Siebel database, and click Next.
9. On the ODBC Data Source screen, accept the default name of the ODBC data source, or enter a different data source name.

   > **Note:** The Siebel Server installation process automatically creates the data source, using the format EnterpriseName_DSN.

   To continue, click Next.
10. On the Siebel Database User Name screen, indicate the source database user name. The user name must be an ID authorized to log in to the Siebel database and must have authorization to set the CURRENT SQLID.

    To continue, click Next.
11. On the Siebel Database Password screen, enter the password associated with the source database user name. Reenter the password to confirm it, then click Next.

12. On the Siebel Database Schema Qualifier screen, enter the identifier that designates the Siebel Schema for your database. This is also an authorization ID. The schema qualifier must start with a letter, can be up to 8 characters in length, and cannot contain special characters.

    To continue, click Next.
13. On the Security Group ID/Grantee screen, type the group ID for which schema access is being granted, for example, SSEROLE.
14. On the Schema File screen, specify the name and directory path of the schema file against which the extracted storage control file is to be validated. If your Siebel schema has been modified, use schema.ddl. For the Siebel preconfigured schema, use ddl.ctl.
15. On the Storage control file screen, enter the directory path and filename of the file you extracted (as described in *Extracting a Storage Control File from the DB2 Catalog*) and which you want to validate.

    To continue, click Next.
16. On the Log Output Directory screen, indicate the directory where you want the log files to be generated. The default directory is SIEBSRVR_ROOT `\log\dbconfig_validate_mf\output` (Windows) or SIEBSRVR_ROOT `/log/dbconfig_validate_mf/output` (UNIX).
17. Save the configuration information you have entered and launch the Siebel Upgrade wizard as described in the following topics:
    - *Running the Database Configuration Wizard on Microsoft Windows*
    - *Running the Database Configuration Wizard on UNIX*
18. Click OK to start the validation process.
19. When the validation process completes successfully, click Exit to exit the Database Configuration Wizard.

    If the validation process fails, errors are noted in the log files in the directory you specified in Step 16. Correct the errors noted and run the validation process again.

    If you chose the Batch - Generate Unload/Load validate option, the validation process generates the following additional files in the log output directory. Review these files to determine the cause of validation errors:
    - load.ldc
    - schema.db.sql
    - schema.grt.sql
    - schema.nuind.sql
    - schema.oind.sql
    - schema.sql
    - schema.tbl.sql
    - schema.tbsp.sql
    - schema.uind.sql
    - unload.ldc

Performing a Custom Installation

This task is a step in *Process of Performing a Custom Installation*.

This topic describes how to perform a custom Siebel database installation using the Database Configuration Wizard.

When you select the Install Database option on the Database Configuration Wizard, you can choose the installation mechanism to use: either Run DDL Automatically or Generate DDL Into Files. If you select the Run DDL Automatically option, performing the steps in the following procedure completes the custom Siebel database installation.

If you select the Generate DDL Into Files option, after performing the procedure in this topic, you must also complete the tasks described in *Completing the Siebel Schema Installation Using Generated DDL* to complete your custom Siebel database installation.

To perform a custom installation

1. Launch the Database Configuration Wizard and follow the steps described in *Performing a Standard Installation* until the Select Installation Type screen appears (Step 17).
2. Select the Customized Install option, and click Next.
3. In the Storage Control File screen, enter the name of the customized storage control file that you want to use to configure storage on the DB2 host. You can use the Browse button to navigate to this file.

   To continue, click Next.
4. In the Select Installation Mechanism screen, indicate which installation mechanism you want to use:
   - Generate DDL into Files

     Select this option to generate the DDL required to create the Siebel Schema into a file. You must later use Siebel-provided scripts, your own FTP, or a similar file transfer program to transport the DDL (schema.sql and ddlview.sql) to the DB2 host, where the DBA executes it, using customary methods, for example, SPUFI.

     To continue, click Next and proceed to Step 5.
   - Run DDL Automatically

     Select this mechanism to apply the DDL and DML required to create the Siebel Schema directly against the database.

     The Siebel application uses ODBC to apply all necessary steps in this installation method.

     To continue, click Next and proceed to Step 7.
5. On the DDL Commit Frequency screen, choose the number of DDL statements to be run before a COMMIT statement is issued. Click Next.
6. In the Output Directory screen, specify the name of the directory where you want to save the Siebel Schema DDL files that are generated. The default directory is DBSRVR_ROOT `\db2390\dboutput\install` (Windows) or DBSRVR_ROOT `/db2390/dboutput/install` (UNIX).
7. On the Log Output Directory screen, indicate the directory where you want the install log files to be generated, and click Next.

   By default, the files are created in SIEBSRVR_ROOT `\log\install_mf` (Windows) or SIEBSRVR_ROOT `/log/install_mf` (UNIX).
8. Save the configuration information you have entered and launch the Siebel Upgrade Wizard as described in the following topics:
   - *Running the Database Configuration Wizard on Microsoft Windows*
   - *Running the Database Configuration Wizard on UNIX*
9. Press OK and the Upgrade Wizard is launched.

- If you selected the Run DDL Automatically installation option, the Siebel Upgrade Wizard applies the DDL commands to create the Siebel Schema directly against the database on the z/OS host. When you receive a message stating that the configuration was applied successfully, click OK.

  This step completes the Run DDL Automatically installation option.
- If you selected the Generate DDL Into Files installation option, click OK to continue, and the Siebel Upgrade Wizard generates the following files into the output directory that you designated in Step 6.

  - schema.sql and ddlview.sql files. These contain the DDL to create the Siebel tables and indexes.
  - instftp.bat. Use this batch file to run the Siebel-provided scripts, your own FTP, or a similar file transfer program to transport the DDL (schema.sql and ddlview.sql) to the DB2 host.
  - instftp.txt. Use this file to transfer files to the mainframe.
  - jobinstl.txt. Contains JCL and REXX execs to execute the DDL.

  To complete the installation process, you must now perform the steps in *Completing the Siebel Schema Installation Using Generated DDL* to apply the schema.sql and ddlview.sql files on the DB2 host.

When you have successfully completed the installation, you are ready to import the Siebel Repository. For further information, see *Importing the Siebel Repository*.

About the Siebel Log Files

The Upgrade Wizard writes logs that provide detailed information on the installation and configuration processes you run, and they also list all errors. The Upgrade Wizard writes the logs for a process to one of the following directories by default:

- Windows: `SIEBSRVR_ROOT\LOG\process`
- UNIX: `SIEBSRVR_ROOT/log/process`

where process is the name of the process you have run, for example, `install_mf` (install a Siebel database) or `imprep` (import repository).

> **Note:** You can select a different log directory name from the Log Output Directory screen when you run the Database Configuration Wizard.

The process directory contains the following subdirectories:

- **Output.** Directory containing the process log files
- **State.** Directory containing the state.log file

The output and state directories are automatically archived on subsequent runs of a process that completes successfully. (The names of subsequent log directories are appended with `_1`, `_2`, and so on.) To preserve disk space, periodically delete or save log directories to another location.

About the State Log File

Each installation and database configuration process consists of a series of steps, each of which must complete successfully. If the Upgrade Wizard cannot complete a step, it marks the step as incomplete in the state.log file and exits.

You must correct the error and then run the Upgrade Wizard again. When you rerun the Upgrade Wizard, it refers to the state log and resumes at the incomplete step that contained the error.

About Process Log Files

You can identify errors you encounter during an upgrade by reviewing the process log files, named Upgwiz.log (Windows) or srvrupgwiz1.log (UNIX), in the output directory.

The name of the log file increments for subsequent log files that are created if the Siebel Upgrade Wizard is run again for the same process.

Reviewing the Log Files for Errors

This task is a step in *Roadmap for Installing the Siebel Database*.

Review the Siebel database installation and configuration log files to verify that the process you ran completed correctly, and to identify errors that must be resolved. The log files might include errors that are expected and benign. You must compare any error messages found in the log files to the sample error messages listed in the errors file, and correct any non-benign errors. For additional information about the log files, see *About the Siebel Log Files*.

To manually review the log files for unacceptable errors

1. Print the errors file. The errors file lists the benign and expected errors you might find in the log files; you can ignore these errors. The errors file is located in the installation subdirectory:

   Windows: `DBSRVR_ROOT\DB2390\errors.rtf or errors.htm`

   UNIX: `DBSRVR_ROOT/DB2390/errors.txt`
2. Sort the log files in the following directory by date.

   Windows: `SIEBSRVR_ROOT\LOG\process\output`

   UNIX: `SIEBSRVR_ROOT/LOG/process/process`
3. Open each log file, starting with the earliest, and search for errors. Starting with the earliest log file can shorten your research time.

   Log files are identified by the `.log` extension. Errors are either tagged with the word error or enclosed in square brackets [...].
4. For each error found, compare the error description against the list of acceptable errors documented in the errors file.

   Identify errors as follows:

   - If you find the error in the errors file, errors of that type are acceptable and no action is required. Continue to review the errors found in the log file.
   - If a log file is not listed in the errors file, there are no acceptable error messages for that log file.
   - If you find an error that is not listed in the errors file, it is unacceptable. You must correct the condition that caused the error before you run the Siebel Upgrade Wizard again.

5. Repeat Step 4 for each log file.

   For help with determining the cause of an error, create a service request (SR) on My Oracle Support.

   When reviewing error messages, be aware that error numbers can change after the installation of a new driver version. Compare the error descriptions to find out which are acceptable errors for the z/OS platform.

   > **CAUTION:** Although you are unlikely to encounter errors other than those listed in the errors file (see Step 1), it is critical that you review the error messages. Certain errors, such as a failure to create indexes, can result in performance problems or anomalous behavior in Siebel CRM.

About Identifying Errors Returned by Applications

The process log files produced by the Siebel Upgrade Wizard (upgwiz.log on Windows and srvrupgwiz.log on UNIX) do not contain information on errors returned by applications. Review the end of the log file for details about the latest failure. If the step that failed was not a native SQL step (which would be listed in the log file), then it occurred as part of an external utility for which you need to review a corresponding log file, identified by the `/L` parameter.

To find additional operation-specific log files, open upgwiz.log or srvrupgwiz.log using a text editor and search for a ".log" string or a "/L" string. The log file shows the names of log files produced during the installation operation; for example, if the configuration of the Siebel database fails, the upgwiz.log file contains the following information:

```
GenericLog GenericError 1 2011-02-28 15:51:24 (err=1) was returned by application
(C:\siebel\ses\siebsrvr\bin\ddlimp.exe ...)
GenericLog GenericError 1 2011-02-28 15:51:24 Execute file action failed
(err=Launching the application)
```

In an example such as the previous one, the ddlimp.log file contains a detailed error message:

```
DDLIMP-ERR-1071: Unable to build ddl statement "add col" (Open unloadtbl.jcl)
```

Rerunning the Installation

If you need to rerun the Siebel Schema installation from the beginning, either after completing an installation run or after starting but not finishing the run, enter a new log output directory name.

You have to rerun the installation when you generate a new DDL file for the Siebel Schema based on a new or modified storage control file.

8 Importing the Repository and Performing Postinstallation Tasks

Importing the Repository and Performing Postinstallation Tasks

This chapter describes how to import the Siebel Repository and describes other postinstallation procedures that you perform to complete the Siebel database installation and configuration process.This chapter contains the following topics:

- *Process of Completing the Siebel Database Implementation*
- *Importing the Siebel Repository*
- *Granting Table Privileges*
- *Validating the Siebel Schema*
- *Populating the Siebel File System*
- *Installing License Keys*
- *Installing Multilingual Seed Data*
- *Importing a New Language to Your Repository*

Process of Completing the Siebel Database Implementation

After completing the Siebel Schema installation, there are still a number of tasks you must perform before your Siebel CRM implementation is completed. These tasks are listed as follows:

1. *Importing the Siebel Repository*.
2. Configure the Siebel Server.

   For information on this task, see *Siebel Installation Guide* .
3. Perform the relevant postinstallation tasks (some are mandatory and some optional):
   - *Granting Table Privileges*
   - *Validating the Siebel Schema*
   - *Populating the Siebel File System*
   - *Installing License Keys*
   - (Optional) *Installing Multilingual Seed Data*
   - (Optional) *Importing a New Language to Your Repository*

Importing the Siebel Repository

This task is a step in *Process of Completing the Siebel Database Implementation* .

Importing the Siebel Repository is the final stage in the Siebel Schema installation process. When you import the Siebel Repository, you populate all the repository tables in the Siebel database with Innovation Pack 2017 application objects.

Regardless of how many Siebel CRM applications you are using (for example, Siebel Sales, Siebel Service, Siebel Marketing), you load data into the repository tables only once for each installation.

You can also export the Siebel Repository into a platform-independent file that can be sent to Global Customer Support for analysis in case of problems that cannot be diagnosed by telephone or email. For help with problems relating to your Siebel application, create a service request (SR) on My Oracle Support.

To import the Siebel Repository

1. Launch the Database Configuration Wizard and follow the steps in *Performing a Standard Installation* until the Siebel Database Operation screen is displayed (*Performing a Standard Installation*).
2. Select the Import/Export Repository option, and click Next.
3. The following options are available on the Select Repository Operation screen:
   - **Import Repository.** Used to import the Siebel Repository for the first time with a base language.
   - **Add language to an existing Repository.** If you have already imported your Siebel Repository and its base language, select this option to add another language to the repository. For information, see *Importing a New Language to Your Repository*.
   - **Export Repository.** Exports the Siebel Repository into a platform-independent file that can be sent to Global Customer Support for analysis, if needed.

     > **Note:** You can also use the repository export option to replicate repositories.

   Select the Import Repository option, and click Next.
4. On the Import Selection screen, indicate whether you want to import a standard Innovation Pack 2017 or a customized repository, by clicking the appropriate radio button.

   > **Note:** Select Import Custom Repository when you are importing a multilingual repository from a test or development environment. This imports all languages to your target repository.

   To continue, click Next.
5. On the Language Selection screen, select the base language in which you want to run the database.
6. On the ODBC Data Source Name screen, enter the name of the ODBC data source, or enter the database alias you prefer to use for the data source. If you specify a database alias, the alias must have been previously registered within DB2 Connect.

   The Siebel Server installation process automatically creates the data source, using the syntax: EnterpriseName DSN. To continue, click Next.
7. In the Siebel Database User Name screen, enter the database user name, then click Next.

   The Database User Name is the ID used to log into the Siebel database. This user ID must be part of the security group with authorization to set the current SQLID to the schema name.

8. In the Siebel Database Password screen, enter the password associated with the database user name, then click Next to continue.
9. On the Siebel Schema Qualifier screen, enter the ID that identifies the Siebel Schema owner, and click Next.

   The Import Repository Name screen appears.
10. On the Import Repository Name screen, enter the designated name of the repository you want to import, for example, Siebel Repository, then click Next.
11. Enter the file and path name of the repository file you want to import, then click Next.
12. On the Log Output Directory screen, specify where you want the import repository log files to be generated.

    By default, the files are created in SIEBSRVR_ROOT `\log\imprep` (Windows) or SIEBSRVR_ROOT `/log/imprep` (UNIX).
13. Save the configuration information you have entered and launch the Siebel Upgrade wizard as described in the following topics:
    - *Running the Database Configuration Wizard on Microsoft Windows*
    - *Running the Database Configuration Wizard on UNIX*
14. Click OK to begin the repository import.

    A window appears, displaying information about the repository import activities.

    If the repository import is successful, the Upgrade Wizard displays a message that the repository has been imported. Otherwise, review the log files. For more details on logs, see *About the Siebel Log Files* and *Rerunning the Installation*.

    > **Note:** If the Siebel Repository import fails midway through the process, you must clean up the data from the failed import by using Siebel Tools to delete the repository.

Granting Table Privileges

This task is a step in the *Process of Completing the Siebel Database Implementation* .

To grant the Siebel group ID additional, required privileges on EIM tables, the database administrator must edit and execute the grantstat.sql file. This procedure is described in this topic.

To edit and run the grantstat.sql script

1. Navigate to the grantstat.sql script which is located in the DBSRVR_ROOT `\db2390` directory (Windows) or the DBSRVR_ROOT/ `db2390` directory (UNIX).
2. Edit grantstat.sql by replacing &1 with the Siebel Schema Qualifier ID that you recorded in *Deployment Planning Worksheet*.
3. Execute the grantstat.sql script, using the method you prefer.

Validating the Siebel Schema

This task is a step in the *Process of Completing the Siebel Database Implementation* .

After you install the Siebel database and import the repository, you must compare the physical database schema with the repository to make sure that there are no inconsistencies between them. Use the Siebel Server utility, dbchck.exe (Windows) or dbchck (UNIX), located in the bin subdirectory of your Siebel Server installation directory, to make this comparison.

You can use the dbchck utility to validate data relationships, including foreign keys and the list of values. You can also use this utility when you have made changes to the extensibility of your Siebel database.

The most popular mode in which to run dbchck is using the option `/dict /all`. This option provides a comprehensive log file that lists all of the discrepancies between the logical data model defined in the repository you specify and the physical database schema in the Siebel database you run the utility against.

To validate the Siebel Schema

1. From the SIEBSRVR_ROOT `\bin` directory (Windows) or the SIEBSRVR_ROOT/ `bin` directory (UNIX), locate dbchck.exe (Windows) or dbchck (UNIX).
2. (Windows only) Delete the dictionary cache file (diccache.dat) before running dbchck, and verify that there are no EIM or Siebel Remote operations running.

   The dbchck utility creates a new diccache.dat file before carrying out the integrity check. By deleting the existing diccache.dat file before starting dbchck, you ensure that dbchck validates against the Siebel Repository you specify.

   > **Note:** If you stop any EIM or Siebel Remote processes, you can restart them after dbchck has run.

3. Source environment variables as appropriate for your platform, using either siebenv.bat (Windows) or siebenv.sh or siebenv.csh (for UNIX).

   Environment variable scripts are located in the SIEBSRVR_ROOT\ `bin` directory (Windows) or the SIEBSRVR_ROOT `/bin` directory (UNIX).
4. Source the database profile.
5. Run dbchck using the following syntax:

```
dbchck /u SADMIN /p password /t SIEBTO /r "Your Siebel Repository Name" /l dbchck.log /dict /all /s ODBC
 data source
```

where:

- password is the login password to the database.
- Your Siebel Repository Name is the repository you want to compare against the physical data model; in the installation, the default value is:

  ```
  "Siebel Repository"
  ```

  > **Note:** You must specify your repository name within quotation marks ("") after the repository name parameter (`/r`).

- ODBC data source is the ODBC data source applicable to the repository.
- SIEBTO is the Siebel Schema Qualifier.

> **Note:** To view all the dbchck parameters with their descriptions, use option `/h`.

Any discrepancies found appear on the screen. Detailed information is written to a log file in the `SIEBSRVR_ROOT\bin` directory (Windows) or the `SIEBSRVR_ROOT/bin` directory (UNIX).

> **Note:** The log file name is the name you specified after the log file parameter (`/l`). In the previous example, the file is named dbchck.log.

6. Review the log file generated as a result of running this script. Any discrepancies are flagged as failures. Investigate all discrepancies.

Discrepancies in the dbchck Log

After completion of the dbchck.exe script, the dbchck.log file generated might contain discrepancies noted as failures.

Some errors noted in the dbchck.log file are not acceptable and must be addressed. An example of such an error follows:

```
Dictionary column FIN_PERIOD_ID not in physical schema! Dictionary different from physical schema for column
dictionary: REVISED_COST numeric(22,7) null physical: REVISED_COST decimal(10,0) null
```

This error is caused by a mismatch in the data type definition of the column in the logical and physical schema. The definition of a column in the logical and physical schema must be synchronized; you can synchronize the logical definition with the physical schema using Siebel Tools. After the synchronization is done, run dbchck.exe again.

> **Note:** LONG columns, defined as LONG VARCHAR columns in the Siebel Repository, are physically created as VARCHAR columns by DB2 for z/OS. The dbchck utility does not generate errors for LONG columns in these circumstances; this is expected behavior on the DB2 for z/OS platform. For additional information, see *About Long Columns and Siebel Utilities*.

For more information about using the dbchck utility, see the section *Checking the Repository* in the *Siebel Enterprise Installation Manager Administration Guide*.

Populating the Siebel File System

This task is a step in the *Process of Completing the Siebel Database Implementation* .

Specific files required to run the Siebel File System, such as correspondence templates and Siebel Marketing files, are provided with the Database Configuration Utilities software. A directory called DBSRVR_ROOT\ `files` (Windows) or DBSRVR_ROOT `/files` (UNIX) is created automatically when you install the Database Configuration Utilities.

Populate the appropriate subdirectory of the Siebel File System with these file attachments after installing the Database Configuration Utilities and the Siebel database and before running the Siebel Web Client.

To populate the Siebel File System directory

1. Copy the files from the DBSRVR_ROOT\ `files` (Windows) or DBSRVR_ROOT `/files` (UNIX) directory to the att subdirectory of the Siebel File System.
2. Verify that the files are in the correct directory.

Installing License Keys

This task is a step in the *Process of Completing the Siebel Database Implementation* .

After you have installed the Siebel database and performed the mandatory postinstallation tasks, you must install the license keys you received with Siebel CRM 17.0 or later to enable you to use the new release.

This topic describes how to install the Siebel license keys so that users can access the Siebel application.

To add new license keys

1. Start the new-release version of Siebel Tools.
2. Connect to the Siebel database as the Siebel Administrator.

   You are prompted to enter your license key number the first time you log on to Siebel Tools.
3. Add your new license key information and click OK.

For additional information on installing license keys, see *Siebel Installation Guide* .

Installing Multilingual Seed Data

This task is a step in the *Process of Completing the Siebel Database Implementation* .

If your organization deploys internationally and, therefore, requires data to be in multiple languages, you must install multilingual seed data (for example, lists of views, responsibilities, or system preferences). You install this seed data by adding new language packs to your database after you install the base language for your database. These language packs populate the List of Values (LOV) with seed data in the new language.

You must successfully install seed data in your base language before you can add seed data in other languages to your database.

> **Note:** You cannot add secondary languages to the Siebel database for an Enterprise Server unless you have already installed the relevant language pack on the associated Siebel Server.

To install multilingual seed data

1. Install the appropriate Language Pack on the Siebel Server. Follow the instructions for adding a new language to an existing instance as described in *Siebel Installation Guide* .
2. When installation is completed, launch the Database Configuration Wizard by following the instructions under *Performing a Standard Installation*.
3. When the Select Installation Operation screen appears (*Performing a Standard Installation*), select Add a language to an existing Siebel Database, and then click Next.

   > **Note:** To add seed data in a new language to your database, you must have already imported your repository in its base language.

   The Base Language screen appears and displays the languages you have installed for your Siebel database.
4. In the Base Language screen, specify which of the installed languages is the one in which you want to primarily run your database. (This is your base language.)

   To continue, click Next.

   > **Note:** The Language Selection screen appears only if you have installed the files for multiple Siebel Language Packs. Installation of multiple Language Packs can occur either during your initial installation of Siebel CRM or at a later time.

5. In the Language Selection screen, specify the new language you are adding for this database.

   To continue, click Next.
6. On the ODBC Data Source Name screen, enter the ODBC Data Source Name. The Siebel Server installation process automatically creates the data source, using the format EnterpriseName_DSN. Accept the default name of the ODBC data source or enter the database alias you prefer to use for the data source.

   > **Note:** Database aliases must be registered within DB2 Connect. If you define database aliases, you must add the `TXNISOLATION` parameter to the database alias entry in the db2cli.ini file and set it to have a value of 1. For information on setting values in the db2cli.ini file, refer to the relevant IBM documentation.

7. On the Siebel Database User Name screen, enter the Database User Name.

   Type the ID (for example, SADMIN) used to log into the Siebel database. This user ID must be part of the security group (secondary authorization group) with authorization to set the current SQLID to the schema name.
8. On the Siebel Database Password screen, enter the password for the ID used to log into the Siebel database.

   Retype the password to confirm it, then click Next.

9. On the Siebel Schema Qualifier screen, enter the character ID that identifies the Siebel Schema owner, for example, SIEBTO.

   > **Note:** This ID can be up to eight characters in length, must be in uppercase, must start with a letter, and cannot include any special characters. This value is the SQL Authorization ID stored in the CURRENT SQLID special register.

10. On the Security Group ID/Grantee screen, type the group ID for which schema access is being granted, for example, SSEROLE. Click Next.

    For more information about the group authorization ID privileges, see *Security Concepts for a DB2 for z/OS Environment*.

11. The Repository Name screen appears. Type the name of your Siebel Repository or accept the displayed default name.

    To continue, click Next.

12. On the Log Output Directory screen, enter the name of the directory where you want the log files to be generated, and click Next.

    By default, the files are created in SIEBSRVR_ROOT `\log\install_lang` (Windows) or SIEBSRVR_ROOT `/log/install_lang` (UNIX).

13. Save the configuration information you have entered and launch the Siebel Upgrade wizard as described in the following topics:

    - *Running the Database Configuration Wizard on Microsoft Windows*
    - *Running the Database Configuration Wizard on UNIX*

    The Siebel Upgrade Wizard is launched.

14. To begin, click OK.

    A window appears, displaying information about installation activities. A message appears when the installation is completed.

    To verify that the installation was successful, review the log files as described in *About the Siebel Log Files*.

15. When you have finished adding a language to your database, you need to import the Siebel Repository for the language you have just added. For information on completing this task, see *Importing a New Language to Your Repository*.

16. After you have added a language to the Siebel database, your seed data is multilingual so you must enable the multilingual list of values (MLOV) capability within Siebel CRM. You must also enable individual LOVs associated with the language. For information on multilingual deployments, see *Siebel Global Deployment Guide* and *Configuring Siebel Business Applications* .

Importing a New Language to Your Repository

This task is a step in the *Process of Completing the Siebel Database Implementation* .

After you successfully import your Siebel Repository in its base language, you can add additional languages to the repository. By adding a new language to your repository, you populate rows of localizable information, which allows Siebel CRM to better operate in the new language.

Regardless of how many Siebel CRM applications you are using (such as Siebel Sales, Siebel Service, Siebel Marketing), you perform this step only once for each language you want to import.

> **Note:** You can only add an additional language to your repository if it has already been deployed and added to your database. For further information on deploying additional languages, see *Siebel Installation Guide* . For information on adding an additional language to your database, see *Installing Multilingual Seed Data*.

To import a repository in a secondary language

1. After installing multilingual seed data as described in *Installing Multilingual Seed Data*, launch the Database Configuration Wizard.
2. Follow the steps in *Importing the Siebel Repository* to Step 3 (Select Repository Operation).
3. Select the Add Language to an existing Repository option and click Next.
4. In the Language Selection screen, select the new repository language you are adding, and click Next.
5. On the ODBC Data Source Name screen, enter the name of the ODBC data source, or enter the database alias you prefer to use for the data source. If you specify a database alias, the alias must have been previously registered within DB2 Connect.
   The Siebel Server installation process automatically creates the data source, using the syntax EnterpriseName_DSN.
   To continue, click Next.
6. In the Siebel Database User Name screen, enter the database user name, then click Next.
   The Database User Name is the ID used to log into the Siebel database. This user ID must be part of the security group with authorization to set the current SQLID to the schema name.
7. In the Siebel Database Password screen, enter the password associated with the database user name. Reenter the password to confirm it, then click Next.
8. On the Siebel Schema Qualifier screen, enter the ID that identifies the Siebel Schema owner, for example, SIEBTO, and click Next.
9. On the Import Repository Name screen, enter the designated name for this Siebel Repository or accept the default.
10. Enter the localized Repository file name.
    If you are importing a secondary language repository on top of your base-language repository, either accept the default installation path and file name for this repository or type another valid installation path.
11. On the Log Output Directory screen, specify where you want log files to be generated, then click Next.
    By default, the files are created in SIEBSRVR_ROOT `\log\imprep_lang`(Windows) or SIEBSRVR_ROOT `/log/imprep_lang` (UNIX).
12. Save the configuration information you have entered and launch the Siebel Upgrade wizard as described in the following topics:
    - *Running the Database Configuration Wizard on Microsoft Windows*
    - *Running the Database Configuration Wizard on UNIX*
13. To begin, click OK.
    A window appears, displaying information about installation activities. A message appears when the installation is completed.
    To verify that the installation was successful, review the log files. When you import a repository with a new language, it creates the following log files:

- imprep_prim.log
- imprep_lang.log

For further information on log files, see *About the Siebel Log Files*.

14. Click Exit to exit the Database Configuration Utility.

9 Customizing a Development Environment

Customizing a Development Environment

This chapter describes procedures for customizing development environments and migrating those customizations to user-acceptance or production environments on DB2 for z/OS. This chapter includes the following topics:

About Customizing Your Development Environment

Customization of Siebel CRM within your development environment can involve:

- Converting nonpartitioned tables to partitioned tables
- Changing views, business objects, applets, and tables
- Adding new columns to existing tables and adding new tables
- Modifying workflow policies and workflow processes

These tasks are documented in *Using Siebel Tools* and *Siebel Business Process Framework: Workflow Guide* . However, several customization procedures and issues are specific to DB2 for z/OS; these are described in this chapter.

It is assumed that you perform your development on either of the following:

- DB2 on a Windows or UNIX computer
- DB2 for z/OS on a partition of your z/OS computer reserved for development

In many cases, changes to user-acceptance and production (server) databases are made by a database administrator working within a change-management system. Therefore, this guide includes procedures for generating Data Definition Language (DDL) files that are later applied to databases.

About Using Siebel Tools in a DB2 for z/OS Environment

There are certain considerations and procedures to follow in Siebel Tools when developing applications that run against databases on DB2 for z/OS. These include:

- *Siebel Tools Configuration File Parameters*
- *Setting Database Options*
- *Storage Control File Names*
- *About Inactivating Unused Indexes*
- *About Reducing VARCHAR Field Lengths*

Siebel Tools Configuration File Parameters

The following parameters in the `[ServerDataSrc]` section of the Siebel Tools application configuration file (tools.cfg) must be set for DB2 for z/OS, as shown in the following table.

Configuration File Parameter	Value
TableOwner	Schema qualifier
MaxCursorSize	-1
PrefetchSize	-1

Setting Database Options

When developing applications on any platform (for example, DB2 for Windows) that will eventually be deployed on DB2 for z/OS, you must set the appropriate database options in Siebel Tools. Setting these options enables all Siebel Tools features for DB2 for z/OS and validates Siebel objects for the z/OS operating system:

- CHAR columns with a length greater than 1 are allowed.
- Users can eliminate unused indexes.
- Users can reduce the size of VARCHAR fields.

To set database options for DB2 for z/OS

1. Start Siebel Tools.
2. From the Tools menu, choose View, and then the Options menu item.

   The Development Tools Options dialog appears.

3. On the Database tab, select the Developing for deployment on DB2 for zSeries check box.

Storage Control File Names

When working in Siebel Tools, you must specify the complete paths for storage control file names in the Apply dialog box when applying schema changes to a database. (In previous releases, the storage control file was referred to as the table groupings file.)

About Inactivating Unused Indexes

The standard Siebel data model has a large number of indexes that can degrade performance on DB2 for z/OS. This performance degradation can occur because DB2, unlike Oracle and Microsoft SQL Server, stores NULL as one of the indexed values. Each row with a NULL value for an indexed column gets an entry of its record identifier (RID) in the DB2 index.

Because many Siebel tables are related to a large number of other tables, there are many indexes on foreign keys. Where these related tables are not used, DB2 assigns the indexes for the foreign keys one entry: NULL. This key entry has a long RID chain comprised of all the RIDS for every row in the table. There is maintenance overhead each time a row is inserted and even more overhead when it is deleted. Therefore, users of DB2 for z/OS might want to deactivate unused indexes. Do not deactivate unused indexes in Siebel Tools. To deactivate indexes, contact your Oracle sales representative for Oracle Advanced Customer Services to request assistance from Oracle's Application Expert Services.

About Reducing VARCHAR Field Lengths

DB2 for z/OS pads VARCHAR fields to their maximum lengths in indexes. To increase flexibility, Siebel CRM makes extensive use of VARCHAR columns. This use of VARCHAR columns can cause performance degradation for z/OS customers who have a large number of rows in certain tables, such as S_CONTACT.

The total length of an index equals the combined lengths of all index columns. To reduce the index length, you reduce the length of participating columns. To make better use of index space, users of DB2 for z/OS can reduce VARCHAR lengths, in cases where there is no application impact, using Siebel Tools.

Analyze your business needs at installation or upgrade to determine whether it is necessary to reduce VARCHAR field lengths. Some examples are the following:

- FST_NAME and LAST_NAME, VARCHAR(50) columns in S_CONTACT. These belong to many indexes.
- ADDR, a VARCHAR(200) column in S_ADDR_PER.
- NAME, a VARCHAR(100) column in S_ORG_EXT.

Be aware that you also must make the same length reductions to columns denormalized from those you have reduced. One example is S_PER_RESP.PER_FST_NAME, which is denormalized from S_CONTACT.FST_NAME.

A denormalized column duplicates the data in a column in another (base) table for performance reasons. The table and column names of the duplicated column are specified in the Denormalization Path property of the Column object definition of the denormalized column. For more information on columns, see *Using Siebel Tools* .

About Siebel LONG Columns on DB2 for z/OS

This topic describes how LONG columns in the Siebel Schema are implemented on the DB2 for z/OS platform. Siebel CRM uses LONG columns on RDBMS tables to store arbitrarily long character data up to 16,350 characters. Examples include scripts, emails, notes, and descriptions. On all RDBMS platforms, except DB2 for z/OS, the size of LONG columns is fixed and is independent of the size of the table space in which the table is defined.

On the z/OS platform, a LONG column in the logical Siebel Schema is indicated as a LONG VARCHAR column in the Siebel Repository. As of DB2 Version 9.1 for z/OS, LONG VARCHAR columns cannot be created with the LONG attribute in the DB2 catalog. Instead DB2 for z/OS creates these columns as VARCHAR columns that are set to the maximum size possible. The size of these VARCHAR columns varies according to two factors:

- The buffer pool and page size of the table space in which the table is defined.
- The combined byte size of all other columns in the table.

You can estimate the size of the LONG column by calculating the buffer pool size of the table space minus the size of all other columns; the LONG column takes up the remainder of the buffer pool size defined for the table space. For example, a table containing one LONG column, defined in a 16-KB table space, with a combined byte size for all other columns of 5 KB, results in the creation of a VARCHAR column having a size of about 11 KB. For more information on calculating the size of columns, refer to the vendor documentation on the IBM Web site.

About Long Columns and Siebel Utilities

The Siebel dbchck utility compares the physical database schema with its logical definition in the Siebel Repository and generates errors for any inconsistencies between them. However, the dbchck utility does not generate errors for LONG columns, which are identified as LONG VARCHAR columns in the Siebel Repository and physically created by DB2 as VARCHAR columns, because inconsistencies between the logical and physical definitions of these columns is expected behavior on the DB2 for z/OS platform.

When you run other Siebel utilities, for example, Synchronize Schema Definition (ddlsync), the utilities generate LONG DDL syntax for these LONG columns if the tables containing the columns need to be re-created. However, when the table is re-created, the LONG syntax is converted by DB2 and the column is again physically created as a VARCHAR column set to the maximum size possible.

How Siebel Tables with LONG Columns Are Stored

On the z/OS platform, LONG columns can be up to 16 KB in length. If Siebel CRM created all LONG columns on DB2 for z/OS with a length of 16,350 characters, this would effectively force all tables to a 32-KB table space, thereby increasing buffer pool storage requirements (each table space is assigned a buffer pool of the same size).

To minimize the number of tables created in 32-KB table spaces, during the Siebel installation process, tables with LONG columns are created in an 8-KB table space by default. In Siebel CRM, LONG columns have an assumed minimum logical length of 4096 bytes, which forces the table to be created in an 8-KB table space. However, some of the larger tables with LONG columns are created in 16-KB table spaces. In addition, the S_SERVICE_SCRPT table is defined in a 32-KB table space because the LONG column in this table needs as much space as possible.

Note: Although Siebel CRM sets the LONG column to have a logical length of 4046 bytes, the physical length is not specified in the `CREATE TABLE` statements for the LONG column. So the resulting length of a LONG column is still set by DB2 for z/OS to a VARCHAR of the maximum length possible, that is, all the space not used by other columns.

About Moving Tables With LONG columns to Larger Table Spaces

By creating most tables with LONG columns in an 8-KB table space by default, almost all aspects of the Siebel application function without problems. However, depending on your business needs and the amount and type of data a LONG column is to contain, you might choose to define a Siebel table containing a LONG column in a larger or smaller table space.

For example, if your implementation involves a usage scenario in which very large notes, descriptions, emails, or other items up to 16,350 characters in length are stored, you can resolve this situation by moving the table to a larger table space, either 16 KB or 32 KB. Be aware, however, that if you move a table to a 32 KB table space, the following problems can occur:

- The LONG column can become very large, which increases the storage requirements for buffer pools and can result in wasted space because the Siebel application usage of LONG columns is limited to 16 KB. If additional data is stored in a LONG column, using a tool such as SPUFI, only the first 16,350 bytes of data is retrieved by the Siebel application. Therefore, it is recommended that before executing the DDL that creates a table with a LONG column in a 32-KB buffer pool, you change the physical column definition to VARCHAR (16,350).
- Large LONG columns in 32 KB table spaces can cause SQL SELECT statements to fail because the result set of a SELECT statement containing a LONG column can exceed the 32 KB in-memory sort limit of DB2 for z/OS. When DB2 for z/OS performs an in-memory sort, the full length of the LONG column is added to the total length of the result set. As a result, you might experience SQL0670N errors.

  If you encounter DB2 sort limit problems of this type, redefine the LONG column as a CLOB column and move it to a smaller table space. For information about converting LONG columns to CLOB columns, see *Converting LONG VARCHAR Columns to CLOB Columns*.

Determining the Table Space Size for Tables with LONG Columns

To avoid LONG columns becoming unnecessarily large, define tables containing LONG columns in table spaces and buffer pools that are an appropriate size. This involves considering factors such as the type of data that the LONG column stores, and the maximum amount of data that the column is likely to contain. The following procedure describes how to estimate the appropriate table space size for a table with a LONG column.

To estimate the appropriate table space size for a table with a LONG column

1. Enter the following query to determine your current minimum length requirements for an existing LONG column:

```
SELECT MAX(LENGTH(column_name))
FROM tableowner.tablename
```

where:

- column_name is the name of the LONG column
- tableowner.tablename is the name of the table owner and the table containing the LONG column

2. Using the value returned by the query as a guideline, adjust the column length to take into account future needs.
3. On the basis of your calculations in Step 1 and Step 2, use the guidelines in the following table to determine the size of the table space appropriate for the table containing the LONG column.

> **Note:** The values shown are general guidelines only; the actual table space size required for a specific table might vary depending on table-specific variables.

Length of Data in LONG Column	Table Space Size
Less than 3404 bytes	4 KB
Less than 7486 bytes	8 KB
Less than 15,678 bytes	16 KB
15,678 to 16,350 bytes	32 KB However, if SELECT queries are failing because of the in-memory sort limit of DB2 for z/OS, convert the LONG column to a CLOB.

4. Move the table to a smaller or larger table space and buffer pool as required.

   For information on this task, see *Moving Tables Between Table Spaces*.

Moving Tables Between Table Spaces

You can move tables containing LONG columns to larger or smaller table spaces as appropriate for your environment. Circumstances in which you might consider moving tables between table spaces include the following:

- If a table containing a LONG column is defined in a 32-KB table space and if SELECT queries are failing because the row length of the table exceeds the 32-KB limit, then the column must be converted to a CLOB and the table can be placed in a smaller table space.
- DB2 for z/OS creates LONG columns as VARCHAR columns that are set to the maximum size possible. It is recommended that, where possible, you change the physical definition of the LONG column to a VARCHAR column of a fixed length, and move the table containing the column to a smaller table space.

  In determining the size of the VARCHAR column, take into account that if the column does not exceed the buffer pool size limit, the column size can be easily increased at a later time using an ALTER statement, which is

an additive schema change. However, if the column size must be decreased at a later time, you must drop the table and then re-create it, which is a non-additive schema change that involves shutting down the production database.

The following procedure outlines the steps in moving a table with a LONG column to a larger or smaller table space.

To move a table to a table space of a different size

1. Determine the appropriate table space size for the table using the guidelines in *Determining the Table Space Size for Tables with LONG Columns*.
2. Unload the data in the table.
3. Delete the table using the SQL DROP command.
4. Re-create the table in a table space of a different size.

   If the table is the only table defined in the table space, when re-creating the table you can specify the same table space name, provided that you specify a different buffer pool size. If the table is defined in a table space with several other tables, when re-creating the table, create it in a new table space where the table is the only table in the table space.
5. Load the data back into the table.

Example of Moving a Table to a Table Space of a Different Size

This topic gives one example of moving a table to a table space of a different size. You might use this feature differently, depending on your business model. In this example, the LONG column in the table does not exceed 4000 bytes in length so the table can be defined at the physical schema level as a VARCHAR in an 8-KB table space.

To move a table with a LONG column to a different table space

1. Unload the data from the table, casting the LONG column as a VARCHAR column with a maximum length of 4000 bytes.

   Defining a LONG column as a VARCHAR data type with a defined length means that even if you define the table containing the column in a larger table space, the length of the column is limited in size; this is beneficial when performing database upgrades.

   ```
   SELECT
   CLASS_ID,
   COMMENTS,
   …
   NAME,
   PROCEDURE_NAME,
   REPOSITORY_ID,
   ROW_ID,
   SSE_FLG,
   SST_FLG,
   SSV_FLG,
   USER_AGENT_CD,
   CAST(column_name AS VarChar(4000)) AS column_name
   FROM tableowner.tablename WITH UR;
   ```

   where:

   - column_name is the name of the LONG column
   - tableowner.tablename is the name of the table owner and table containing the LONG column
2. Use the SQL DROP command to remove the original table, then re-create the table in an appropriately sized table space, in this example, an 8-KB table space.

3. Load the data back into the table.

About Siebel Tables and CLOB Columns

This topic describes how character large object (CLOB) data types are implemented on the z/OS platform.

When you install Siebel CRM, a number of Siebel application objects are defined as CLOB data types in the Siebel Repository; such objects can have up to 128 KB of data for a single data element in a table row. Other Siebel tables have columns that are defined as LONG columns in the Siebel Schema, and as LONG VARCHARs in the Siebel Repository, but are converted to CLOB columns on z/OS. Siebel objects that are defined as LONG columns but stored as CLOBs can have up to 16,350 bytes of data for a single data element in a table row.

Siebel Tables with LONG Columns That Are Created as CLOBs

The following Siebel tables have columns defined as LONG that are converted to CLOB columns on z/OS (Clobs is set to Yes in the storage control file definition of the table). Siebel CRM uses CLOB columns for these tables to avoid the in-memory sort limit of DB2 for z/OS.

These tables belong to the Siebel Repository and are not read or updated frequently so the performance issues associated with using CLOB columns, described in *Issues in Using CLOB Columns Instead of LONG VARCHAR Columns*, are not relevant:

- S_BITMAP_DATA
- S_DMND_CRTN_PRG
- S_EVT_MAIL
- S_NOTE
- S_NOTE_ACCNT
- S_NOTE_CON
- S_NOTE_OPTY
- S_SCHMST_DBSCPT
- S_SCHMSTEP_SCPT
- S_SERVICE_SCRPT

If appropriate for your environment, you can enable CLOB columns for Siebel tables other than those listed. For information on this task, see *Converting LONG VARCHAR Columns to CLOB Columns*.

Siebel Tables Defined with CLOB Columns

The following Siebel tables have columns that are defined as CLOB columns in the Siebel Repository. Activity on these tables is quite high, but these tables cannot be stored in 32-KB table spaces because they have very large columns (greater than 128 KB) and so could encounter the in-memory sort limit of DB2 for z/OS:

- EIM_AUDIT_ITEM
- EIM_AUDIT_READ

- S_AUDIT_ITEM
- S_AUDIT_READ
- S_BR_GBL_BINARY
- S_BR_MODULE_BIN
- S_DOCK_TXN_LOG
- S_TU_LOG
- S_TU_LOG_X_01
- S_TU_LOG_X_02
- S_TU_LOG_X_03
- S_TU_LOG_X_04
- S_TU_LOG_X_05
- S_WEBCHNL_SES
- S_WEBCHNL_SNSVC

If appropriate for your environment, you can also define new extension columns for tables as CLOB data types. For information on this task, see *About Defining New Extension Columns as CLOB Data Types*.

About Defining New Extension Columns as CLOB Data Types

If you create an extension column for a table, you can define the column as a CLOB data type by typing the value CLOB in the Physical Type field of the column in Siebel Tools (although CLOB is not one of the options in the Physical Type field drop-down list). For information on creating extension columns for tables, see *Roadmap for Creating Custom Extensions to the Siebel Schema*.

It is recommended (though not required) that you also enable CLOB columns for the table containing the extension column in the storage control file as described in *Converting LONG VARCHAR Columns to CLOB Columns*.

Converting LONG VARCHAR Columns to CLOB Columns

Siebel application objects defined as LONG columns in the Siebel Schema are defined as LONG VARCHAR data types in the Siebel Repository. If you choose, you can store these columns as character large objects (CLOBs) on DB2 for z/OS.

The default setting for these objects in the Siebel Schema is LONG because of the performance and storage characteristics of CLOBs in a DB2 for z/OS environment. These characteristics are described in *Issues in Using CLOB Columns Instead of LONG VARCHAR Columns*. However, it is recommended that you store LONG VARCHAR objects in the Siebel Repository as CLOBs in the following circumstances:

- If you have Siebel Marketing, change the LONG column in the S_NOTE table to a CLOB.
- If you write your own scripts, it is recommended that you convert LONG columns that contain scripts to CLOB columns to safeguard against space limitations that are inherent to the structure of z/OS.

> **Note:** If you choose to store LONG VARCHAR objects as CLOB objects, the logical and physical schema definitions of the object are not synchronized. As a result, when you run the dbchck utility, errors are generated for the relevant columns; these can be ignored.

The Siebel installation, upgrade, and migration processes create all auxiliary objects necessary to support CLOBs so that their use is transparent. However, CLOBs are not enabled by default. To store Siebel application objects defined in the Siebel Repository as LONG VARCHAR data types as CLOB data types, perform the following procedure.

To convert LONG VARCHAR columns in a table to CLOB columns

1. Edit the storage control file as described in *About Modifying Storage Control Files*.
2. Locate the table object for which you want to enable CLOB columns, and set the value of Clobs to YES.
   All LONG VARCHAR columns defined for the table in the Siebel Repository are created as CLOB columns in the physical Siebel Schema.

Issues in Using CLOB Columns Instead of LONG VARCHAR Columns

If you use a CLOB column instead of a LONG VARCHAR column, you do not run into the in-memory sort limit of DB2 for z/OS. This memory limitation does not occur because DB2 for z/OS does not account for the actual length of the CLOB column in estimating the total row width to be sorted in memory. However, there are a number of issues in using CLOB columns as follows:

- Using a CLOB column has an impact on performance.
  Fetching data from CLOB columns requires extra network flows between the client and the z/OS host, which can impact response time. Use CLOB data in form applets but be cautious about using CLOB data in a Siebel list applet if you want to keep the volume of data in network flows to reasonable levels. Writing data to CLOB columns is also slower than writing to LONG VARCHAR columns so for tables that are frequently used, moving to CLOB columns is not an option.
- Using a CLOB column has an impact on storage (DASD).
  Using CLOB columns generally increases the amount of storage (DASD) needed, because space is allocated for the total width of the CLOB column even if the CLOB column contains only one byte of data.
- Compression is not supported for LOB table spaces.
- Perhaps the most important trade-off of using CLOBs relates to recovery considerations. Using CLOBs with LOG YES provides a point of forward recovery. However, there is a cost associated with logging the contents of a CLOB column. You need to consider this additional overhead before deciding to use CLOB columns.

Converting Nonpartitioned Tables to Partitioned Tables

Siebel CRM supports two scenarios for converting nonpartitioned tables to partitioned tables.

> **Note:** In the following scenarios, source table refers to your current nonpartitioned table, and target table refers to the destination partitioned table.

- Source table and target table are defined in different table spaces.
  If the source table and target table are defined *in different table spaces*, and the target table space is a new table space that does not yet exist in your current database, no special action is required. Run the two utilities

available in the Database Configuration Wizard (Migrate Repository and Synchronize Schema Definition) to automatically rebuild the table in the new partitioned table space and remove the original table from the database.

- Source table and target table are defined in the same table space.
  Database operations do not support rebuilding a table space from nonpartitioned to partitioned. Therefore, if the source table and target table are defined in the same table space (for example, if you used the dbconf.xls spreadsheet to convert an existing table space from nonpartitioned to partitioned), you must perform one of the following procedures (as appropriate for your configuration) to convert the nonpartitioned table to a partitioned table.

To convert to a nonpartitioned table for a partitioned table space configuration

1. Use your preferred database tool to manually move every nonpartitioned table (every table that you intend to convert) to a temporary table space, and then use the SQL DROP command to remove the original table space.
2. Use the Siebel Database Storage Configurator (dbconf.xls) to modify the storage control file, changing the mode of the table spaces to partitioned.
   For further information, see *About Modifying Storage Control Files*.
3. Run the Database Configuration Wizard and select the Migrate Repository option. This process automatically rebuilds the source tables in the new partitioned table space and removes the original nonpartitioned table.
4. Manually remove the temporary table space that you created in Step 1 from the database.

To convert nonpartitioned tables to partitioned tables using alternate tools

- You can use alternate tools (for example, BMC Change Manager or IBM Compare Utility) to convert nonpartitioned tables to partitioned table spaces. In this case, after applying all changes, change the schema version by executing the following SQL:

```
update S_APP_VER
set CUSTOM_SCHEMA_VER = char(integer(CUSTOM_SCHEMA_VER) + 1)
```

This SQL is stored in ddlview.sql and can be executed outside of the Siebel application.

About Creating Custom Extensions to the Siebel Schema

There are several different possible scenarios for creating Siebel Schema custom extensions:

- Creating a small extension column, so the table fits into its existing table space
- Creating a large extension column, so the table has to be redefined to a larger page size and therefore a larger table space
- Creating an extension table

If planning custom extensions to the Siebel Schema on the z/OS platform, consider the following:

- If a LONG column is defined for a table, attempting to add an extension column to the table causes an error. This is because a LONG column uses all the space in a table space that is not used by the non-LONG columns,

leaving no space available for an extension column. In this case, use an extension table instead. For further information, see *About Siebel LONG Columns on DB2 for z/OS*.

- You can create a new extension column of type CLOB for a table by typing the value CLOB in the Physical Type field of the column in Siebel Tools, although this option is not available in the drop-down list.
- Creating tables with Identity type columns or extension columns of Identity type is not supported.
- The process of creating a small extension column is the same on z/OS as on other platforms and is documented in *Using Siebel Tools* . To create a large extension column or an extension table on the z/OS platform, you must specify a 16-KB or 32-KB table space:
  - If you are developing on DB2 for Windows or UNIX, you must enter a 16-KB or 32-KB table space in the Apply Schema screen (see Step 4.)
  - If you are developing on DB2 for z/OS, you must edit the storage control file to specify the database and table space in which the new extension column or table is to reside.

For further information on creating custom extensions to the Siebel Schema, see *Roadmap for Creating Custom Extensions to the Siebel Schema*. For information on estimating storage needs, see *About Creating Custom Extensions to the Siebel Schema*.

Roadmap for Creating Custom Extensions to the Siebel Schema

To extend the Siebel Schema by creating extension columns or tables, you need to perform the following tasks:

1. Plan the new custom object, for example, determine the type of object you need to create, the name of the object, and its size.
2. Add a storage definition for the new object to the storage control file of the Siebel Schema being extended.

   For information on this task, see *Amending the Storage Control File for New Schema Objects*.
3. Create the new object in Siebel Tools; this adds the object definition to the Siebel Repository thereby updating the logical schema definition in the development environment.

   The process of adding a new object in Siebel Tools is the same on z/OS as on other platforms. However, you need to make sure that you select the check box on the Database tab of the Development Tools Options dialog under the View menu. This option validates the object's compliance with DB2 for z/OS sizing conventions.
4. Apply the physical schema extensions to the development database, specifying the storage control file you updated in Step 2. You can apply the changes directly or generate the DDL into a file which can be applied on the z/OS host later.

   If you are developing on DB2 for z/OS, follow the procedures in *Migrating Customizations from Development to the Target Database*.

   Some organizations do not allow direct database extension. Siebel Tools does not allow the Apply/ DDL button to be disabled but the Apply process fails if developers do not have appropriate database privileges (such as CREATEDBA and DBADM). If you do not want developers to extend tables directly, you can control this using database privileges. In this case, developers follow the procedures in *Migrating Customizations from Development to the Target Database* to apply schema extensions to their development databases.
5. If you chose to generate the DDL into an output file, review the file, then have your DBA execute the file on the z/OS host.

6. In your development environment, update and test the configuration changes that apply to the extensions you made.
7. Apply the schema extensions on the production database on the z/OS host.

   For information on this task, see *Process of Applying Schema Extensions to the Target Database*.

Amending the Storage Control File for New Schema Objects

If you need to create new database objects as a result of changes made to the Siebel Schema in Siebel Tools, you must create a storage definition for the new objects in your storage control file. If you do not create storage control file definitions for new objects, the required objects are created using values derived from the Defaults object in the storage control file when you apply the schema extensions to the development database using Siebel Tools.

This task is a step in *Roadmap for Creating Custom Extensions to the Siebel Schema*.

> **Note:** If you have different storage control files for your development environment and production environment, you must add the new object definitions to both storage control files.

Use the following procedure to add new object definitions to a storage control file.

To amend the storage control file for new schema objects

1. Using the Extract from catalog option on the Database Configuration Wizard, extract the storage control file from the Siebel Schema that is being extended.

   For information on this task, see *Extracting a Storage Control File from the DB2 Catalog*.
2. Add the new object definitions to the extracted storage control file. For example, if you create a new extension table, you must create a table object definition, and object definitions for the database and table space in which the table is to be placed.

   You can edit the storage control file using a text editor or you can use the Siebel Database Storage Configurator to edit the file. For additional information, see *Using the Siebel Database Storage Configurator*.

   You can add new object definitions anywhere in the storage control file, for example, you can append them to the end of the file. The new object definitions are read from the file when you apply the physical schema extensions to the development database. Once the new objects are physically created in the Siebel Schema, the next time the Extract from catalog task is run to produce a new storage control file, these objects are correctly grouped by type in the file.

   > **Note:** The Extract from catalog process collects the definition for only Siebel Schema database and table space objects that are associated with a table.

   The following example shows object definitions for a new extension table, named X_EXTENT, which is created in a table space, named S0600100, which is in a database, named SIDB6001:

```
[Object 6001]
Type = Database
Name = SIDB6001
LockSize = Page
[Object 6002]
Type = Tablespace
Name = S0600100
Database = SIDB6001
LockSize = Page
```

```
Bufferpool = BP16K1
Define = No
Partitions = 0
[Object 6003]
Type = Table
Name = X_EXTENT
Database = SIDB6001
Tablespace= S0600100
CLOBS = NO
```

When specifying the object number, for example [Object 6001], enter the next number in sequence after the last object number listed in the existing storage control file.

Spooling DDL Generated by the Apply Button

The Siebel environment variable, SIEBEL_GENERATE_DDL, allows you to record into a file the DDL that is generated when you click the Apply button in Siebel Tools.

To spool the DDL generated by the Apply button to a file

1. Before running Siebel Tools, set the environment variable from the command line or using Windows environment parameters:
   - If using the command line, set `SIEBEL_GENERATE_DDL` to `Y`.
   - If using the Windows environment parameters, do the following:
     - From the Start menu, choose Control Panel, and then the System option.
     - Click the Advanced tab, then click Environment Variables.
     - In the System Variables box, click New.
     - The New System Variable window appears.
     - Enter `SIEBEL_GENERATE_DDL` in the Variable name field and `Y` in the Variable value field, and click OK.
     - Continue to click OK until you are out of the System Variable window.
2. Open Siebel Tools, and click the Apply button.
   The DDL is generated to the file output.sql in the `bin` subdirectory of the Siebel Tools installation directory, such as `C:\Siebel\Tools_1`.

Process of Applying Schema Extensions to the Target Database

After testing schema extensions that you created in the development environment, you can migrate the changes to the target user-acceptance or production database. To apply schema extensions to the target database, perform the following tasks:

1. *Preparing the Target Database*
2. *Migrating Customizations from Development to the Target Database*
3. *Applying Schema Changes to the Target Database*.

This process is a step in *Roadmap for Creating Custom Extensions to the Siebel Schema*.

Preparing the Target Database

This topic describes how to prepare a Siebel production database before Siebel Schema extensions are applied.

This task is a step in *Process of Applying Schema Extensions to the Target Database*.

Complete the following actions before migrating the changes to the target database:

- Prepare the storage control file (for example, my_storage_file.ctl).

  If you have created any new extension tables, edit the storage control file to specify the database and table spaces where the new extension tables are to reside. For information on this task, see *Amending the Storage Control File for New Schema Objects*.
- Ask all mobile users to synchronize.
- Make sure all connected clients are disconnected from the Siebel database.
- When all mobile user transactions have been merged and routed, stop the Siebel Server.
- Perform a full backup of the database.

> **Note:** If you are changing the data type or length of custom extension columns that already contain data in the target database, export that data before making the schema changes. After making the changes, import the data back into the target database.

Migrating Customizations from Development to the Target Database

This topic describes how to migrate customizations from a development to a target database.

When migrating customizations from your development source to your target database on the DB2 host, you can use one of two modes of execution:

- **Run DDL Automatically.** Select this mode to apply the DDL and DML required to create the Siebel Schema directly against the database, using an ODBC connection.
- **Generate DDL Into Files.** Select this mode to generate the DDL required to create the Siebel Schema into files for transfer to the DB2 host.

This topic describes the DDL-generation mode of execution that is generally used under a change-management system.

This task is a step in *Process of Applying Schema Extensions to the Target Database*.

> **Note:** Before you migrate your customizations from your development database to your target database, validate your storage control file as described in *Validating an Extracted or Modified Storage Control File*.

If your development environment is on DB2 for z/OS, your DBA can clone the development database to the target database. For more information, see *Cloning a DB2 for z/OS Database*.

Siebel CRM does not support customized database triggers. If you have created customized triggers on your Siebel base tables, disable them before migrating the database schema. You can re-create the triggers after the migration is finished.

To migrate the schema

1. Launch the Database Configuration Wizard and follow the steps in *Performing a Standard Installation* until the Siebel Database Operation screen is displayed (Step 6).
2. Select the Migrate Repository option, and click Next.
3. In the Source Repository Selection Screen, choose one of the following options, and click Next:
   - **Read source repository directly from the database.** If you select this option, proceed to Step 5.
   - **Read source repository from a previously exported file.** If you select this option, the Repository File Selection screen appears. Proceed to Step 4.
4. If you selected the Read source repository from a previously exported file option, enter the name of the source repository file you want to use, and click Next.
5. Indicate whether the target database will be online or offline when the repository migration process runs against the target database, and click Next.
6. Select one of the following options, and click Next:
   - **There are new schema changes to be applied.** If you select this option, during the migration process, schema changes defined in the new repository are applied to the physical target database. The target enterprise must be offline.
   - **There are no new schema changes to be applied.** If you select this option, schema changes defined in the new repository are not applied to the physical target database during the migration process.
7. On the Language Selection screen, indicate the language in which the target database runs, and click Next. (This screen is displayed only if you have more than one language deployed.)
   - If you selected the Read source repository from a previously exported file option in Step 3, proceed to Step 12.
   - If you selected the Read source repository directly from the database option in Step 3, proceed to Step 8.
8. Enter the ODBC Data Source Name to use to connect to the Siebel database, and click Next.
9. Enter the Database User Name for the database in which the source repository resides, and click Next.
10. Enter the password associated with the Database User Name. Re-enter the password to confirm, and click Next.
11. Enter the Siebel Schema Qualifier of the database in which the source repository resides. This name is a character ID that identifies the Siebel Schema owner, for example, SIEBTO.
12. On the Source Database Repository Name screen, enter the name of the repository you are migrating, then click Next.

    You are not prompted to enter this value if you chose the Read source repository from a previously exported file option in Step 3.
13. On the Target Database Repository Name screen, enter the name you want to assign to the repository after it is migrated, then click Next.
14. In the Target RDBMS Platform screen, select IBM DB2 UDB for z/OS, and then click Next.
15. Specify whether the target database you are migrating to is Unicode or non- Unicode, and click Next.
16. Enter the target database ODBC data source name, then click Next.
17. Enter the target database DB2 subsystem name, then click Next.
18. In the Target Database User Name screen, enter the Target Database Username, then click Next.
19. In the password screen, enter the password associated with the Target Database Username. Confirm the password by typing it again, then click Next.
20. In the Target Schema Qualifier screen, type the target schema qualifier in uppercase, and then click Next.
21. In the Target Security Group ID/Grantee screen, type the target security group authorization ID, then click Next.

If you indicated that there are no new schema changes to be applied (Step 6), proceed to Step 25.

22. In the Migrate Repository Mechanism screen, select the Generate DDL into Files option, and then click Next.
23. On the DDL Commit Frequency screen, choose the number of DDL statements that can be run before a COMMIT statement is issued, and click Next.
24. On the Output Directory screen, enter the DDL output directory. The default directory is DBSRVR_ROOT `\db2390\dboutput\` dev2prod (Windows) or DBSRVR_ROOT/ `db2390/dboutput/` dev2prod (UNIX).
25. On the Storage Control File screen, enter the name and directory path of the storage control file to be used in the migration, for example, DBSRVR_ROOT `\db2390\` my_storage_file.ctl (Windows) or DBSRVR_ROOT/ `db2390/` my_storage_file.ctl (UNIX).

    Click Next.
26. On the Log Output Directory screen, specify where log files generated during the migration are to be created, for example, SIEBSRVR_ROOT `\log\dev2prod_mf` (Windows) or SIEBSRVR_ROOT `/log/dev2prod_mf` (UNIX), and click Next to continue.
27. Save the configuration information you have entered and launch the Siebel Upgrade wizard as described in the following relevant topic:
    - *Running the Database Configuration Wizard on Microsoft Windows*
    - *Running the Database Configuration Wizard on UNIX*
28. Click OK to continue.

    The Siebel Upgrade Wizard creates a number of files in the directory you specified in Step 24. These files contain the SQL and unload and load control cards used to perform the repository migration from development to production on the DB2 host.

    A file, ftpsync.txt, is also created; use this file to transfer the migration files to z/OS where they are applied. For information on this process, see *Applying Schema Changes to the Target Database*.
29. Click Exit to exit the Database Configuration Wizard.

Applying Schema Changes to the Target Database

After generating the DDL into files, you must use Oracle-provided scripts, your own FTP, or a similar file transfer program to transport the DDL files to the z/OS host, where you execute it using customary methods, for example, SPUFI.

This task is a step in *Process of Applying Schema Extensions to the Target Database*.

To apply schema changes to the target database using Oracle-provided scripts

1. Navigate to the output directory you specified in Step 24 and open the ftpsync.txt file.
2. Edit the ftpsync.txt file using the following information:
   a. Change &IP to the IP address of your DB2 host, for example `ZM01`.
   b. Change &Username to your own user name, for example `SADMIN`.
   c. Change all occurrences of `SIEBELQ1` to your own high-level qualifier (`HLQ`), for example, `SADMIN`.
   d. Save the file.

      The following is an example of the ftpsync.txt file:

      ```
      open &IP user &Username
      ```

```
quote site cylinders primary=1 secondary=1

quote site recfm=fb lrecl=80 blksize=0
quote site cylinders primary=1 secondary=1

send &directoryPath1/jobsync.txt 'HLQ.SIEBEL.V00'
quote site cylinders primary=10 secondary=2

send &directoryPath1/unload.ldc 'HLQ.SIEBEL.V01'
quote site cylinders primary=10 secondary=2

send &directoryPath1/load.ldc 'HLQ.SIEBEL.V02'
quote site cylinders primary=5 secondary=2

send &directoryPath1/schema.sql 'HLQ.SIEBEL.V03'
quote site cylinders primary=1 secondary=1

send &directoryPath1/bumpver.sql 'HLQ.SIEBEL.V04'
```

3. Execute ftpsync.txt to send the DDL files to the host, as follows:
   - If you are using Windows, double-click jobsync.bat.
   - If you are using UNIX, execute Unix_jobsync.bat.

     The Unix_jobsync.bat file contains the following code:

     ```
     mv ./schema.sql schema.sql.old
     cat ./schema.db.sql ./schema.tbsp.sql ./schema.tbl.sql ./schema.grt.sql ./
     schema.uind.sql ./schema.oind.sql ./schema.nuind.sql > schema.sql
     echo User
     ftp -vn < ftpsync.txt > ftpsync.log
     ```
4. Enter the password associated with the user name you specified in the ftpsync.txt file, and press any key to continue.
5. The log file ftpsync.log is created in the DBSRVR_ROOT `\db2390\dboutput\dev2prod` (Windows) or DBSRVR_ROOT `/db2390/dboutput/dev2prod` (UNIX) directory.

   The log contains information on the file transfer.
6. When the transfer is successfully completed, log on to the mainframe, and edit the DSNHLQ.SIEBEL.V00D2P data set by changing the job card and DSNHLQ.
7. Submit the job and, when it is completed, make sure that the return code is 0.

   A new partitioned data set (PDS) is created, DSNHLQ.SIEBEL.D2P.EXEC.
8. Execute the following REXX exec to apply the DDL generated for the schema migration process.

   ```
   Execute clist DSN
    HLQ.SIEBEL.D2P.EXEC(SBLD2P)
   ```
9. Verify the schema changes.

Synchronizing Siebel Repository Definitions and the Physical Siebel Schema

To synchronize the Siebel Repository database definitions and the existing physical Siebel database, use the Synchronize Schema Definition option of the Database Configuration Wizard.

When you synchronize repository definitions and the existing Siebel Schema, the following data sources are accessed:

- Siebel Repository
- Storage control file
- DB2 catalog

When you run the Synchronize Schema Definition option, be aware of the following:

- Custom columns in the Siebel Schema that are not in the Siebel Repository are not deleted
- Custom indexes in the Siebel Schema that are not in the Siebel Repository are deleted

Note: It is recommended that you use the SQL DROP command to remove EIM tables before synchronizing the Siebel logical and physical schemas to ensure that the synchronization process does not fail when processing EIM tables.

To synchronize Siebel schema definitions

1. Launch the Database Configuration Wizard and follow the steps in *Performing a Standard Installation* until the Siebel Database Operation screen (Step 6) is displayed.
2. Select the Run Database Utilities option, and click Next.
3. Select the Synchronize Schema Definition option, and click Next.
4. Identify the appropriate database encoding method, and click Next.
    - UNICODE Database
    - Non-UNICODE Database

    If you selected the UNICODE Database option, proceed to Step 7.
5. Specify the code page encoding scheme for your Siebel database, either ASCII or EBCDIC, and click Next.
6. On the Siebel Tools Directory and Tools DSN screen, enter the directory where Siebel Tools is installed. Oracle reccomends that you do not select the BIN folder.
   This step is required when schema changes are made so that the changes are published into the Runtime Repository.
7. On the Language Selection screen, select the language in which the database runs, and click Next.
8. Enter the ODBC data source name to use to connect to the database, and click Next
9. Enter the source database DB2 subsystem name, then click Next.
10. Enter the database user name. Click Next.
11. Enter the password associated with the database user name, then enter the password again to confirm it. Click Next.
12. Enter the Siebel schema qualifier in uppercase, and click Next.
13. Enter the name of the security group ID/Grantee, and click Next.
14. Specify the Repository Synchronization Mechanism, in this case, select the Generate DDL into Files option, and click Next.
15. On the Commit Frequency screen, choose the number of DDL statements that can be run before a COMMIT statement is issued, and click Next.
16. Enter the name of the output directory where the DDL is to be created, then click Next.
    By default, the DDL output directory is DBSRVR_ROOT `\db2390\dboutput\ddlsync` (Windows) or DBSRVR_ROOT `/db2390/dboutput/ddlsync` (UNIX). Click Next.
17. Enter the path and name of the storage control file to use in the synchronization process. Alternatively, use the Browse button to locate this file, then click Next.
18. Enter the name of the Repository with which the existing physical Siebel database is to be synchronized. Click Next.

19. Enter the name of the directory where the log files are to be created, and click Next.

    By default, the files are created in SIEBSRVR_ROOT `\log\ddlsync_mf` (Windows) or SIEBSRVR_ROOT `/log/ddlsync_mf` (UNIX).
20. Save the configuration information you have entered and launch the Siebel Upgrade wizard as described in the following topics:
    - *Running the Database Configuration Wizard on Microsoft Windows*
    - *Running the Database Configuration Wizard on UNIX*

    The Wizard generates the DDL required to synchronize the Siebel database and the Siebel Repository.

    a. If you selected the Run DDL Automatically installation option, the Siebel Upgrade Wizard applies the DDL commands to synchronize the Siebel database with the Repository directly against the database on the z/OS host. When you receive a message stating that the configuration was applied successfully, click OK. This step completes the Run DDL Automatically synchronization option.
    b. If you selected the Generate DDL Into Files installation option, the Siebel Upgrade Wizard generates the following files into the output directory that you designated in Step 19 after the synchronize schema definition configuration is completed:
       - schema.sql
       - ddlview.sql

       Ask your DBA to apply these files to synchronize the Siebel Repository database definitions with the existing physical Siebel database.

Cloning a DB2 for z/OS Database

You can clone your existing Siebel database to a target database. The DB2 system cloning process involves the following steps:

1. *Extracting a Storage Control File from the DB2 Catalog*

   Extract a storage control file from the existing Siebel Schema.
2. *Generating a DDL File from a Storage Control File*

   Use the extracted storage control file to generate a DDL file, then apply the DDL file to the database to create a clone of the Siebel Schema.

Generating a DDL File from a Storage Control File

Use the following procedure to generate DDL from a storage control file.

To generate a DDL file

1. Launch the Database Configuration Wizard and follow the steps in *Performing a Standard Installation* until the Select Installation Type screen is displayed (Step 17).
2. Select the Customized Install option, and then click Next.
3. Specify the storage control file you want to use to create the DDL, and click Next.
4. In the Select Installation Mechanism screen, select the Generate DDL Into Files option, and click Next.

5. On the DDL Commit Frequency screen, choose the number of DDL statements that can be run before a COMMIT statement is issued, and click Next.
6. In the Output Directory screen, specify the location of the directory in which the DDL is to be created, for example, DBSRVR_ROOT `\db2390\dboutput\install` (Windows) or DBSRVR_ROOT/ `db2390/dboutput/install` (UNIX). Click Next.
7. In the Log Output screen, specify the directory where log files are to be created or accept the default. Click Next to continue.

   By default, the files are created in SIEBSRVR_ROOT `\log\install_mf` (Windows) or SIEBSRVR_ROOT `/log/install_mf` (UNIX).
8. Save the configuration information you have entered and launch the Siebel Upgrade wizard as described in the following topics:
   - *Running the Database Configuration Wizard on Microsoft Windows*
   - *Running the Database Configuration Wizard on UNIX*
9. Press Ok and the Wizard generates DDL from the storage control file you specified in Step 3 into the output directory that you designated in Step 6.

   When the DDL files are created, the following message appears.

   ```
   Files schema.sql and ddlview.sql with the Siebel Schema modifications have been
   generated in the DDL Output Directory. Please select Yes to exit now and apply
   the files
   ```
10. Select Yes.

    The schema.sql and ddlview.sql files are created.
11. Apply the DDL files using either an optional process provided by Siebel CRM or any other tools used in your company for applying DDL.

    If you want to clone the Siebel Schema into more than one database, copy the generated DDL files, replace the schema qualifier in the DDL files, and then save the files under a new name.

About Data Migration

After you have successfully migrated your source schema to your target database, you must copy data from your development environment. Such data might include:

- Modified files, such as the Web templates, image files, and cascading style sheets
- Transactional data, such as accounts, contacts, and opportunities
- Setup data, such as employees, positions, and responsibilities
- Program data, specifically Assignment Manager rules and Workflow processes and policies, and personalization rules and expressions

For information on copying such data, see *Migrating Data Using Siebel Enterprise Integration Manager* and *Developing and Deploying Siebel Business Applications* .

Customizing Applications Using Assignment and Workflow Rules

You can customize applications using Siebel Assignment Manager and Siebel Workflow, as well as by using Siebel Tools.

You can use Siebel Assignment Manager to create business rules that automatically assign entities, such as opportunities, service requests, or activities, to the most qualified individuals. For more information, see *Siebel Assignment Manager Administration Guide* .

Siebel Workflow provides a graphical interface for designing and implementing business processes and user interactions. Workflow processes define the steps to automate business processes, such as sending email. Workflow policies trigger processes when they detect certain conditions, for example, an opportunity being assigned. For more information, see *Siebel Business Process Framework: Workflow Guide* .

To activate assignment or workflow rules you must run the following Siebel Server components:

- Generate Triggers (GenTrig).

  Allows you to create database triggers. GenTrig writes database triggers to a SQL file, such as TRIGGER.SQL. On the DB2 host, the SQL file is executed manually by a DBA. To execute TRIGGER.SQL manually, run GenTrig with the EXEC parameter set to FALSE.

  > **Note:** For DB2, GenTrig does not log into the database as the table owner. Instead, it logs in as the privileged user. All triggers generated are qualified with the schema qualifier. When starting the GenTrig component, users are prompted for the schema qualifier.

- Workflow Monitor.

  Uses the database triggers to identify the records that might match policy conditions.

For information on activating rules, see *Developing and Deploying Siebel Business Applications* .

10 Maintaining Siebel CRM Applications on DB2 for z/OS

Maintaining Siebel CRM Applications on DB2 for z/OS

This chapter describes maintenance tasks that you can perform to improve the performance of Siebel CRM applications on DB2 for z/OS after you have completed the Siebel CRM installation. This chapter includes the following topics:

DB2 Statistics for Siebel CRM

It is recommended that you apply statistics to Siebel CRM. Update statistics on table spaces that contain EIM interface tables and base tables (including table spaces that contain extension tables and repository tables) when there has been a change of 20 percent or more in the row distribution. It is usually not necessary to update statistics on all of the table spaces, only on those containing tables that have changed.

The RUNSTATS utility and the DSTATS utility are recommended for updating statistics.

About the RUNSTATS Utility

The RUNSTATS utility can be used to collect distribution and frequency statistics for all types of columns (indexed and nonindexed) and for user-defined groups of columns. You can use RUNSTATs to collect statistics on the most and least frequently occurring values, and to collect cardinal values for groups of columns.

Refer to the DB2 catalog history tables to determine when you need to reorganize a table or when you need to update the statistics for a table. When you execute RUNSTATS, you can specify options that allow you to update the catalog history tables without updating the statistics in the catalog tables. This allows you to populate the history tables without affecting existing access paths.

You can run the rstat390 utility from your UNIX or Windows midtier computer to execute RUNSTATS on the DB2 host. The rstat390 script is located within the SIEBSRVR_ROOT `\bin` directory (Windows) or the SIEBSRVR_ROOT `/bin` directory (UNIX). For rstat390 help, use option `/h`.

Update statistics only when there is little activity on the Siebel database, such as after midnight. If you run this utility while users are accessing and updating the Siebel database, lock contention can occur. When this happens, an error message is generated, for example, the following error message is generated from rstat390:

```
ODBC error S1000 in SQLExecDirect: [IBM][CLI Driver][DB2/6000] SQL2310N The utility
could not generate statistics. Error "-911" was returned.
```

This error does not indicate that your database has been harmed; however, you do have to rerun the RUNSTATS job for any table for which this type of error is generated, because statistics were not updated for that table.

You can execute RUNSTATS on an active system if you specify shrlevel change as an option. This option allows concurrent access while the RUNSTATS utility executes. You can also execute RUNSTATS directly from the DB2 host. For additional information on the options available with the RUNSTATS utility, see the relevant IBM documentation.

When to Use the DSTATS Utility

You can execute RUNSTATS following the reorganization of any fragmented table spaces and indexes. For more information on this topic, see *About Reorganizing Table Spaces, Partitions, and Indexes*.

If you are experiencing slow queries, however, run the DSTATS utility. Slow queries can result when the optimizer chooses an inefficient access path as a result of data skew on nonleading indexed columns or nonindexed columns. In such cases, IBM recommends running the DSTATS utility to collect column distribution statistics on these columns; this can lead to significant improvements in the query time.

About Reorganizing Table Spaces, Partitions, and Indexes

It is recommended that you reorganize the table spaces, partitions, and indexes that have a tendency to become fragmented.

No strict guidelines can be offered as to which table spaces, partitions, and indexes might require reorganization due to the variety in application and customer operation variables at any given customer site. However, database administrators must pay attention to the status of large or heavily used table spaces and indexes, because fragmentation of these objects can affect performance significantly.

Avoid using online REORG of Siebel Repository table spaces. Schedule reorganization of Siebel Repository table spaces during application downtime to avoid adversely affecting the Siebel application.

After you reorganize table spaces and indexes, it is recommended that you execute rstat390 (from the midtier) or RUNSTATS (on the DB2 host). For more information on how to run this utility, see *About the RUNSTATS Utility*.

About Cursor Close

Cursor close allows you to regulate system resource utilization by Siebel clients in a z/OS environment by setting the MaxCursorSize cursor configuration parameter. This parameter specifies the number of database rows a Siebel client cursor can get from a user-entered query or a Siebel operation that generates a SQL query.

Note: There are two cursor configuration parameters: MaxCursorSize and PreFetchSize. When using Siebel CRM on DB2 for z/OS, however, only the value you specify for the MaxCursorSize parameter is used; values specified for the PreFetchSize parameter are ignored.

The MaxCursorSize cursor configuration parameter is read at startup from the following places:

- Enterprise profile configuration parameters on the Siebel Server for Siebel Web Clients.
- Application configuration (.cfg) file for Siebel Developer Web Clients and Siebel Mobile Web Clients.

The Siebel client connector closes the SQL cursor after the number of rows specified by the MaxCursorSize cursor close parameter have been retrieved.

You can choose whether to use cursor close in restricted or unrestricted modes.

Unrestricted Mode

When the MaxCursorSize cursor configuration parameter is not specified or is set to a value of -1, the Siebel application opens a cursor on the user's behalf and retrieves blocks of records until the query result set is exhausted or until the user cancels the display of remaining records. When a user remains on a list applet with multiple screens of data, for example, Contact List Applet, the application continues to hold the cursor.

The impact on database and network resources when a query returns a large result set can be significant.

Restricted Mode

During normal operation (restricted mode), with the MaxCursorSize cursor configuration parameter set appropriately, the Siebel application retrieves up to the MaxCursorSize number of rows from the Siebel database.

For example, if MaxCursorSize is specified as 128, the Siebel application retrieves up to 128 rows from the database for a single operation or query. The Siebel application then closes the cursor, releasing the thread for other users to access. The user can then scroll through the retrieved rows, because the rows are cached on the Siebel Server within the Application Object Manager.

If a user tries to scroll to records beyond the value of MaxCursorSize (128 in the example), the cursor close alert appears:

```
There were more rows than could be returned. Please refine your query to bring back
fewer rows.
```

Suppose, for example, that you navigate to the My Opportunities view, which is sorted by the Name column. Seventy-five rows qualify for this query. The initial query returns 128 rows. When you try to scroll to the 129th opportunity, the cursor close alert appears. You then have to choose Edit, Query, and then Refine Query. Type '>' in the Name field, and then execute the query to retrieve additional records.

Cursor Close Parameter Values on DB2 for z/OS

By default, on other DBMS platforms, Siebel CRM sets both MaxCursorSize and PreFetchSize to a value of -1, which causes the cursors to remain open to retrieve additional data.

On DB2 for z/OS, setting the value of the MaxCursorSize parameter to -1 can cause a high number of DB2 Database Active Threads (DBAT); this can potentially lead to DB2 DBM1 address-space task abnormal ends, caused by DBM1 virtual storage constraints. To alleviate this problem, it is recommended that z/OS users set the MaxCursorSize parameter to a positive value. (Siebel CRM on DB2 for z/OS ignores values specified for the PreFetchSize parameter.)

The minimum acceptable value that you can specify for the MaxCursorSize cursor close parameter is 32; the default value is 128. This value allows a reasonable number of rows to be retrieved without holding open the cursor. After the cursor is closed, the change is committed and the Database Active Thread (DBAT) becomes inactive.

> **Note:** The default value of the MaxCursorSize parameter for Siebel Tools is -1; because the number of Siebel Tools users is relatively small, the number of active connections at any time is unlikely to exceed the DB2 for z/OS limit.

It is recommended that your corporate IT department and Siebel CRM administrator collaborate to determine the value for the cursor close parameter that is most appropriate for each class of users connecting to DB2 for z/OS. Guidelines are outlined in the following table; these are not mandated, but they provide a balance between system resource utilization, user responsiveness, and productivity. However, your enterprise might have unique requirements beyond the scope of this chapter.

Cursor Close Ignored

Many specialized Siebel business components, such as Application Administration, List Management, all charts, all picklists (List Of Values), and all calendar views have been modified to ignore the cursor configuration parameter setting. Ignoring the settings allows these functions to operate correctly.

Modifying the Cursor Close Alert Message

You can modify the message that is displayed to the user when there are more rows than can be returned in a query.

To modify the cursor close alert message

1. In your Siebel application, navigate to the Administration-Application screen, then the System Preferences view.
   The System Preferences view appears.
2. In the System Preference Name column, select the SSASqlErrRsltsDiscarded parameter.
3. Edit the value in the System Preference Value column to how you want the cursor close alert message to display.
   The maximum message length is 200 characters.
4. Restart the Siebel application.

Changing Cursor Configuration Parameter Values

This topic describes how to change the cursor configuration parameter values. You can change the cursor configuration parameter for Siebel clients in the following locations:

- Application configuration (.cfg) files for the Siebel Developer Web Client and Siebel Mobile Web Client.
- From the command line for the Siebel Web Client.
  Because the Siebel Web Client does not have an application configuration file, it uses the configuration parameter values specified for the Enterprise Server; these values are applied to all server components and the Web Client.

- Application configuration (.cfg) file for Siebel Tools.
- At the business component level in Siebel Tools.

Setting the Cursor Configuration Parameter for the Siebel Web Client and Siebel Servers

For the Siebel Web Client, the MaxCursorSize parameter value is set from the command line at the Siebel Enterprise Server level. As a result, you cannot set this parameter separately for each class of users. You must set it to the same value for all users, typically 128.

For additional information on modifying enterprise server parameters and using the Siebel Server Manager command-line interface (srvrmgr) program, see *Siebel System Administration Guide* .

To set the cursor configuration parameter value for the Siebel Web Client

1. List the DSMaxCursorSize cursor configuration parameter to check its current value. From the Siebel Server Manager command-line interface, enter the following:

   ```
   srvrmgr > list advanced param DSMaxCursorSize for named subsystem ServerDataSrc
   ```

2. Change the value of the DSMaxCursorSize parameter by entering the following from the command line:

   ```
   srvrmgr > change parameter DSMaxCursorSize=value for named subsystem
   ServerDataSrc
   ```

   where value is the new value you want to specify for the DSMaxCursorSize parameter.
3. Stop and restart the Siebel Server for the changes to take effect.

Siebel Scripting Considerations

This section outlines best practices for managing cursor modes in Siebel scripts when querying DB2 on z/OS and OS/390.

Follow these best practices for scripted queries when the underlying database is DB2 on z/OS and OS/390:

- Use the **ForwardBackward Cursor Mode** when your business component query is expected to return only a single record for example, when querying pick lists or searching a business component by its Id field set the cursor mode to ForwardBackward.
- Use the **ForwardOnly Cursor Mode** when the query result set is expected to return multiple records, use ForwardOnly cursor mode. In these cases, always traverse to the end of the result set using the NextRecord method to ensure the cursor is properly closed.

Setting the Cursor Configuration Parameter for the Siebel Developer Web Client and Siebel Mobile Web Client

For the Siebel Developer Web Client and the Siebel Mobile Web Client, you can set the MaxCursorSize cursor configuration parameter for different classes of users in application configuration files. The following table lists the user classes and the recommended MaxCursorSize parameter value for each class.

> **Note:** You do not need to set a cursor configuration parameter value for Siebel Mobile Web Client users who connect exclusively to the local database and update by synchronizing with the corporate Siebel database. These users do not have a cursor configuration parameter in the `[Local]` sections of their application configuration (.cfg) files.

Take special care with the training of Siebel Mobile Web Client users who access both their local database and the Siebel database. This training must include explaining the difference in operation between the two platforms, setting appropriate expectations for receiving cursor close alerts when connected to the Siebel database, and showing how to formulate queries for common cursor close alert scenarios.

User Class	Description	Recommended Value
Siebel administrator	Accesses the database to perform system updates. Might need to access large amounts of data.	-1
Siebel application developer	Performs development and customization of the Siebel application using Siebel Tools.	-1
Special	Sporadically submits queries or executes operations returning a large number of rows. Special users cannot be effective if their queries are limited by the cursor configuration parameter value. Examples include: • No requery formulation is available to access the next set of data. • Operation requires the complete result set. • Data returned must be internally consistent. When a requery is issued, the returned data set might have changed because of other database activity.	-1
Typical	Majority of users. Examples include call center agents and sales representatives.	128

For information on application configuration files, see *Siebel System Administration Guide* .

To set the cursor configuration parameter for the Siebel Developer and Siebel Mobile Web Clients

1. Create a backup copy of the application configuration (.cfg) file.
2. Open the original .cfg file in a text editor.
3. Set the MaxCursorSize parameter in the `[ServerDataSrc]` section of the .cfg file.
4. Save your changes.

Using Siebel Tools to Bypass the Number of Rows Returned

There might be cases where you do not want to change the general limitation concerning the number of rows returned (default 128), but do want to have an unlimited number of rows available for certain operations.

To do this, leave the parameter value for MaxCursorSize in the .cfg file set to the default number (or any number you choose), but use Siebel Tools to set values for the specific business components for which you want unrestricted operation.

To ignore the cursor close limitation, change the properties for the MaxCursorSize parameter for these business components to -1.

Database Connection Pooling

Siebel database Connection Pooling is a feature that allows database connections to be shared and reused, so reducing the number of DB2 database active threads (DBATs) that are required. This feature saves the overhead of creating and maintaining database connections for each user session and allows the number of concurrent user sessions to exceed the number of database connections. Database Connection Pooling is not enabled by default.

About Database Connection Pooling

With Database Connection Pooling, a user session is not tied to a database connection. The Application Object Manager component creates database connections as required. When a new connection is needed (a user session is started), the Application Object Manager creates an initial connection for user authentication. If there are connections in the connection pool, this initial connection is dropped; if the shared connection pool is empty, this initial connection is added to the pool.

When the user issues a query, it is run on any free database connection in the pool and the connection is blocked until the statement has finished running. When the query completes, the connection is returned to the pool and becomes available to run any other user queries. If a number of users issue fast-running queries, they can share the same connection without blocking.

If the user issues a query and all the existing connections are busy, then a new connection is created, provided the pool has less than the maximum number of connections allowed. The value specified for the MaxSharedDbConns server parameter determines the maximum number of database connections that can be created.

When the maximum number of connections is reached, the Application Object Manager randomly checks all existing connections for their availability and assigns the query to the first free connection that it finds. The Application Object Manager keeps track of connection utilization and determines the minimum number of connections that is required depending on statement load.

For additional information on Database Connection Pooling, see *Siebel Performance Tuning Guide* .

> **Note:** In the current release of Siebel CRM, you can cancel user-initiated query operations even if Connection Pooling is turned on.

About Connection Pooling and Cursor Close

Connection Pooling can give you flexibility in increasing the number of rows returned from a query or in keeping cursors open by setting the DSMaxCursorSize parameter to -1. However, consider the latter option only if it is essential to your business needs. Setting the DSMaxCursorSize parameter to the -1 setting requires careful DB2 buffer pool tuning and the use of DB2 storage contraction DSNZPARMs. For further information, see *About Cursor Close* and *DSNZPARM Parameter Settings for Siebel CRM*.

You must also review the unlimited cursor close values in combination with other factors, such as:

- DB2 DBM1 virtual storage
- Transaction rate of arrival

- Average length of a transaction
- Maximum number of connected threads
- Number of expected long-running complex queries

Dynamic SQL in the Siebel Application

Traditional applications are typically based on static, hard-coded SQL. However, Siebel CRM generates dynamic SQL statements at run time, using the Siebel Application Object Manager and the program configuration in use by the customer. As a result, SQL statements can be as individual as the deployments in which they occur and, in some cases, dynamic SQL can become very complex, for example, joins with 30-40 tables are not uncommon.

To reduce the complexity of generated SQL (such as the number of joins, the number of columns in SELECT statements, or the number of tables in FROM statements), consider eliminating columns from the standard Siebel applets that you do not require for your business needs, especially in the most frequently used screens and views. Also keep this consideration in mind when you customize the Siebel application by creating new objects.

About Spooling SQL in the Siebel Application

Using spooling with the Siebel application can be useful for several reasons:

- It allows you to identify the SQL statement being executed when an SQL error code is returned.
- It can help you in assessing the time taken by the various SQL statements when you encounter performance issues.
- When data appears to be missing from a view, you can identify the SQL used to retrieve that data.

You can then run the SQL manually, either from the midtier computer, for example using the DB2 Command Center, or on the DB2 for z/OS host, for example using tools such as SPUFI, to see what data is being returned.

To minimize the amount of SQL tracing produced, and, therefore, the size of the log files, limit the tests to the minimum number of screen changes needed to allow the issue to be reproduced. If possible, spool SQL on a Developer Web Client as you have more control over the testing which makes it easier to minimize the amount of SQL traced.

> **Note:** The z/OS-specific parameters of SQL statements are not displayed in the spool files. Add appropriate values for the z/OS parameters to the spooled SQL to ensure that the output of any Explain statements you run on the spooled SQL is not misleading.

Turning on SQL Spooling for the Application Object Manager and the Database Connector

You can turn on component event logging for the Developer and Mobile Web Clients and the server component. You can spool the SQL generated by the Siebel Application Object Manager and by the database connector but do not turn on spooling for both. If you do, the amount of data returned will be confusing.

To turn on SQL spooling for the Application Object Manager

- To turn on SQL spooling for the Application Object Manager, set the component event Application Object Manager SQL Log to 4.

To turn on SQL spooling for the database connector

- To turn on SQL spooling for the database connector, set the component event SQL Parse and Execute to 5.

For some components, the SQL Profiling event can be used to identify the most poorly performing queries.

For Developer Web Clients, you can also set these component events as environment variables. They work in addition to the spooling options.

About Tracing SQL Generated by the Siebel Application

The SQL that the Siebel application generates is sent to the DB2 host on the z/OS platform through ODBC and the DB2 Call Level Interface (CLI) before the SQL command is processed by DB2. To effectively troubleshoot issues, you might have to trace the relevant SQL statement through one or all of these layers. For help selecting the most appropriate tracing for the situation, create a service request (SR) on My Oracle Support.

Steps to initiate tracing are described in the following topics:

- *Enabling DB2 CLI Tracing*
- *About Enabling ODBC Tracing*
- *Enabling SQL Tracing for Siebel Database Utilities*
- *About SQL Tracing on the DB2 Host*

Enabling DB2 CLI Tracing

You can enable DB2 Call Level Interface (CLI) tracing by adding the following lines to the db2cli.ini file (this procedure applies to Windows and UNIX).

To enable DB2 CLI tracing

1. Add a section to the db2cli.ini file labeled `[COMMON]`.
2. Add the following entries to the `[COMMON]` section.

```
[COMMON]
Trace=1
TracePathName=C:\TRACE
TraceFlush=1
TraceComm=1
```

For each thread, one trace file is created in the path specified in TracePathName (ensure you have already created the path). On UNIX systems, the DB2 process owner must have the appropriate permissions to write to the directory specified in TracePathName.

Only review the trace files created at the time of problem re-creation.

Each DB2 process generates its own trace file. The events in the DB2 CLI trace are specific to DB2 clients.

About Enabling ODBC Tracing

You can enable ODBC tracing by following the instructions in 475526.1 (Article ID) on My Oracle Support. This document was formerly published as Siebel FAQ 1343. ODBC tracing produces results similar to the CLI trace, but focuses on the ODBC API calls. You cannot see the data being passed in the SQLExecute calls with the ODBC trace but you can see it in the DB2 CLI traces.

Enabling SQL Tracing for Siebel Database Utilities

You can enable SQL tracing by setting the DBUTIL_LOG_EVENTS environment variable before you run any of the Siebel database utilities, for example, the ddlimp, dataimp, ddlsync, or repimexp utilities. These utilities are invoked by the Siebel Migration application, therefore the environment must be set up before running the migration plan. Depending on the operating system you are using, DBUTIL_LOG_EVENTS should be set as the System Environment variable (on Windows) or exported to the siebenv.sh shell script (on UNIX).

To enable SQL tracing for Siebel database utilities

1. On Windows operating systems, define a System Environment variable as shown in the following table.

Variable Name	Value
DBUTIL_LOG_EVENTS	SQLParseAndExecute=5,SQLDBUtilityLog=3

2. On UNIX operating systems, export the environment variable in the siebenv.sh shell script as follows:

```
export DBUTIL_LOG_EVENTS="SQLParseAndExecute=5,SQLDBUtilityLog=3"
```

You must restart the Siebel Server for the change to take effect.

Note: The DBUTIL_LOG_EVENTS environment variable should only be set for debugging purposes, as it can generate large log files and negatively affect performance.

About SQL Tracing on the DB2 Host

Determine the tracing to set on the DB2 host in conjunction with the DBA. Refer to the documentation for the DB2 performance monitoring tool you use to determine how to enable the tracing.

Long-Running Queries and the RLF Error Message

You can set the DB2 Resource Limit Facility (RLF) to cancel long-running queries. For more information on using RLF, refer to the appropriate IBM documentation.

When performing long-running queries, users might receive the following RLF error message:

```
An error has occurred executing a query. Query did not execute successfully because
it exceeded the resource limits set on the database server.
Please simplify your query or report this problem to your systems administrator.
```

This error occurs when an SQL statement is canceled by the DB2 Governor, because it exceeds the resource limits set on the database server.

If this message results from the execution of a Query by Example (QBE), users can try to simplify the query, or contact their systems administrator and describe what they were doing in the application when the error occurred.

Using the odbcsql Utility to Submit SQL Statements

The Siebel Server installation program installs the odbcsql utility in the SIEBSRVR_ROOT `\bin` directory (Windows) or the SIEBSRVR_ROOT/ `bin` directory (UNIX) of the Siebel Server. Siebel CRM uses this program to perform operations on the DB2 database whenever these operations are performed from the midtier, that is, from the Siebel Server.

You can use odbcsql to submit SQL statements, for example, to obtain the DB2 storage group names for data and indexes which you have to specify during the Siebel database installation process.

Note: You can also use other utilities that allow you to query and return result sets to the client.

You can also use odbcsql to test your ODBC data source after you install the Siebel Server.

To execute odbcsql

- Navigate to the SIEBSRVR_ROOT `\bin` directory (Windows) or the SIEBSRVR_ROOT/ `bin` directory (UNIX) and enter the following from the command prompt:

  ```
  odbcsql /s database_alias /u username /p password
  ```

  for example:

  ```
  odbcsql /s siebsrvr_siebel /u sadmin /p dgt^js12*gzs
  ```

  where:

  - `/s` is the source ODBC DSN for the connection
  - `/u` is the user ID
  - `/p` is the password

  If the odbcsql connection is successful, you can enter queries against the DB2 catalog tables. For example, the following query returns a list of all DB2 storage groups:

  ```
  SELECT NAME FROM SYSIBM.SYSSTOGROUP;
  ```

  To exit odbcsql, enter EXIT. For additional information on the utility, enter the following at a command prompt:

  ```
  odbcsql /help
  ```

The odbcsql utility is also useful in troubleshooting postinstallation connectivity problems that arise in the ODBC layer of your installation.

Enabling DB2 Dynamic Statement Caching

Siebel CRM uses the DB2 global statement caching capability. Global statement caching allows dynamic SQL statements that are reexecuted to bypass the PREPARE phase after the first execution. Dynamic statements are prepared once, stored on a global statement cache, then reused many times. The Siebel application also maximizes the reuse of statement caching.

To enable global statement caching either

- Set the CACHEDYN parameter to YES in the macro DSN6SPRM when generating DSNZPARMs.
- Set CACHE DYNAMIC SQL to YES when installing DB2.

About Managing the Dynamic Statement Cache

Because DB2 uses the Environmental Descriptor Manager (EDM) component to manage the dynamic statement cache, it is important that you set EDM-related system parameters to appropriate values. Prior to DB2 for z/OS Version 8, the cached SQL statements were stored in either the primary EDM Pool, or in a data space. In DB2 for z/OS Version 8 and later releases, the cached SQL statements are always stored in a dedicated storage pool above the 2-GB bar. For guidance on setting the EDM system parameters, see your IBM documentation.

Note: Monitor and adjust the EDM Pool and the related cache storage areas based on your workload.

System parameters associated with dynamic statement caching are described in the following table.

Parameter	Value	Description
`CACHEDYN`	`YES`	Enables dynamic statement caching.
`CONTSTOR`	`YES`	Contracts working storage for connections periodically (every 50 commits or when storage use reaches 2 MB).
`EDMPOOL`	`32768`	EDM pool storage size in KB.
`EDMSTMTC`	`102400`	Global statement cache size, in KB, used by EDM in DB2 for z/OS Version 8 and later releases.
`MINSTOR`	`YES`	Minimizes thread storage usage.

About Invalidating Cached Statements

After DB2 prepares a SQL statement and places it in the global statement cache, it is reused. If you are performing SQL tuning and you want to force DB2 to prepare and reoptimize a cached statement, the global SQL statement cache must be invalidated.

To invalidate an entry in the global statement cache for a specific SQL statement, one of the tables referenced by the SQL statement or the table catalog statistics must be altered in some way.

You can invalidate a dynamically cached statement using one of the following methods:

- Use the RUNSTATS utility to automatically invalidate cached statements that refer to objects against which RUNSTATS is executed. You can run the RUNSTATS utility on any table or table space referenced in the SQL statement.

  You can use the REPORT NO UPDATE NONE option to invalidate the cache entries without incurring the overhead of executing all the RUNSTATS logic.
- Use the DB2 command `STOP OBJECT` or the SQL DROP, ALTER, or REVOKE statements (for example, `REVOKE ALL FROM PUBLIC`) on any object related to the plan to invalidate both global and local caches. While some ALTER statements require that an object is stopped, others do not; for example, STOP OBJECT is not required for DROP or REVOKE.
- Use the command `ALTER TABLE` tablename `AUDIT NONE` to purge statements that reference the named table from the cache.
- Use a CREATE INDEX statement on tables used in the DB2 access path.
- Stop the DB2 subsystem and restart it.

Tracing the Source of a Query

Administrators can identify the terminals from which queries are generated using the Session ID. The session ID is composed of three parts: ClientIP_ClientProcessID_SerialNumber. The following is an example of a session ID:

```
172.20.80.25_324_1.
```

Eventually, the Application Object Manager and the DB2 host share the same string which can be used to identify the current connection.

After the session ID is displayed at the Server Manager, you can also check it on the database side.

To verify session ID on the DB2 host

1. At the Option prompt, enter the location of the DB2 Primary Option Menu.

   The DB2 Primary Option Menu appears.
2. At the Command prompt, enter D to access the DB2 Defaults Panel.
3. At the Command prompt, enter the name of the subsystem that you are connecting to.
4. Press F3 to go back to the DB2 Primary Option Menu.
5. At the Command prompt, enter the appropriate selection to go to the DB2 Command window.

6. At the DB2 Command window, enter the following:

```
-DISPLAY THREAD(*) DETAIL
```

   Note: The hyphen before DISPLAY is required.

7. Press Enter to display current active threads.

   The session ID string is displayed in the application name field. If there are many active threads shown, identify your thread by matching the corresponding host name and userid.

About Coordinated Universal Time and DB2 for z/OS

Siebel Global Deployment Guide contains information on using Coordinated Universal Time (UTC). In Siebel CRM, the UTC System Preference default value is TRUE. If you do not want to implement UTC, you must change this value to FALSE.

When installing and configuring UTC on DB2 for z/OS, consider the following:

- The Siebel UTC implementation only works if the system time that the database software uses is in UTC.

  Any system or database parameter that implements a time zone different to UTC must remain at zero. The use of any other value is not supported, as it effectively returns the database configuration to a local time zone.

  As an example, the z/OS system parameter, TIMEZONE, can be set in the CLOCKxx member of SYS1.PARMLIB. For a correct Siebel UTC implementation, the value of this parameter must be zero, as shown:

```
TIMEZONE W.00.00.00 /* GMT */
```

- The UTC upgrader is not supported on DB2 for z/OS.

  Configure UTC using a fresh database install only.

11 Migrating a Siebel Database to Unicode Format

Migrating a Siebel Database to Unicode Format

This chapter describes how to migrate your Siebel database from an ASCII or EBCDIC encoding format to a Unicode encoding format.

This chapter consists of the following topics:

About Migrating a Siebel Database to Unicode Format

You can migrate your Siebel database from either an ASCII or EBCDIC encoding system to a Unicode encoding system if the following conditions are met:

- You are using a Innovation Pack 2017 with DB2 Version for z/OS.
- You are migrating to a DB2 version for z/OS subsystem.

When you migrate your Siebel schema from an ASCII or EBCDIC encoding system to a Unicode encoding system, all of the source tables that are marked as active in the repository are migrated to new target tables which are in Unicode format. Tables that are defined as inactive in the repository are not built in the Unicode target schema.

Migrating to Unicode involves the following steps:

1. Siebel CRM 17.0 and later supports Unicode UTF-16 encoding, that is, it uses a double-byte character encoding scheme. To migrate to Unicode, Siebel database utilities first convert the existing schema to a schema that supports UTF-16.
2. Siebel CRM database utilities then build SQL CREATE schema control cards and JCL templates, which are transferred to the z/OS host and applied to create the target database schema.
3. Finally, data is migrated from the source ASCII or EBCDIC database to the target (Unicode) database using Cross Loader SQL control cards; these are built using options available on the Siebel Unicode Migration Option Menu on the z/OS host.

CAUTION: Migrating to Unicode is more complex than simply importing your existing data into a Unicode database. Failure to execute the migration correctly can result in serious data corruption or unrecoverable data loss. For this reason, the participation of Oracle's Application Expert Services is mandatory if you perform a Unicode migration. Contact your Oracle sales representative for Oracle Advanced Customer Services to request assistance from Oracle's Application Expert Services.

About Unicode Storage Requirements

The use of the Unicode UTF-16 encoding format will increase the storage required by your Siebel database. During the Unicode migration process, data types in the Siebel ASCII or EBCDIC database are mapped from single-byte to double-byte characters, as shown in the following table.

Data Types on ASCII and EBCDIC Databases on Z/OS	Data Types on Unicode Databases on Z/OS
Char	Graphic
Varchar	Vargraphic
Longvarchar	Longvargraphic
Clob	DBClob

Moving from single-byte to double-byte characters causes some Siebel table spaces to exceed their page sizes and causes some tables to exceed the 32-KB limit; these tables will require LOB columns in the Unicode database.

To help you estimate the increase in storage that can occur when you migrate from a code page to a Unicode database, The following table shows the number of table spaces and the storage required for a sample SIA EBCDIC database, and the storage and table spaces required by the same database when it has been migrated to Unicode.

Note: The figures in the following table are provided as a guideline only. The increase in storage required by your actual Unicode database might be greater or lesser than indicated for the sample database.

Siebel SIA Database	No of 4-KB Table Spaces	No of 8-KB Table Spaces	No of 16-KB Table Spaces	No of 32-KB Table Spaces	Total Space Required
EBCDIC	4229 (2,759,008 KB)	331 (312,112 KB)	171 (62,272 KB)	105 (52,160 KB)	3,185,552 KB
Unicode	3941 (2,507,584 KB)	405 (327,776 KB)	345 (358,992 KB)	247 (119,712 KB)	3,314,064 KB

Siebel CRM uses COMPRESS set to YES as the default for table space definitions; setting the COMPRESS parameter to YES reduces the storage requirements for table spaces.

About the NOT PADDED Clause

To minimize any increase in storage when you migrate your database to Unicode, indexes in the Siebel Schema are defined with the NOT PADDED clause by default when you convert your storage control file to Unicode using the Siebel Database Storage Configurator as described in *Converting the Storage Control File to Unicode*.

When the NOT PADDED clause is specified:

- Varying-length columns of an index are not padded to their maximum length; DB2 stores only the key data, which reduces the storage required by the index key.
- VARGRAPHIC columns of an existing index are stored as varying-length columns.
  If the NOT PADDED clause is not specified, VARGRAPHIC columns are stored as fixed-length columns.
  You can specify that indexes are to be NOT PADDED by default by specifying the PADIX DSNZPARM. You can also use the PADDED or NOT PADDED clauses in the CREATE INDEX or ALTER INDEX statements to specify whether an individual index is padded or not.

Differences Between ASCII, EBCDIC and Unicode Sort Order

The EBCDIC, ASCII, and Unicode encoding systems each use a different sort order for numbers, upper case alpha characters, lower case alpha characters, and special characters. As a result, if you load a partitioned table based on these characters, rows can be assigned to different partitions in the target table than they were in the original ASCII or EBCDIC source table, which might lead to unbalanced partitions in your database. You must determine if the partitioning keys need to be reevaluated when you migrate to Unicode.

Roadmap for Migrating the Siebel Database to Unicode

To migrate a Siebel CRM database from an ASCII or EBCDIC encoding system to a Unicode encoding system, complete the following tasks and processes:

1. Perform the Unicode migration prerequisite tasks described in *Requirements for Migrating the Siebel Database to Unicode* .
2. Use the Unicode Migration option of the Database Configuration Wizard and the Upgrade Wizard to generate the schema-related files for the target (Unicode) database.
   For information on this task, see *Generating the Unicode Migration Files*.
3. Transfer the Unicode migration files to the z/OS host and set up the host environment as described in *Process of Preparing the z/OS Host Environment*.
4. *Process of Performing the Database Unicode Migration*.

Requirements for Migrating the Siebel Database to Unicode

The tasks described in this topic are a step in the *Roadmap for Migrating the Siebel Database to Unicode*.

Before you can migrate your Siebel CRM ASCII or EBCDIC database to a Unicode database, you must perform the prerequisite tasks described in this topic.

1. Install Innovation Pack 2017.
2. Ensure that the target environment you are migrating to is a DB2 for z/OS subsystem. The Unicode data conversion process fails if these minimum requirements are not met.
3. Verify that the Group codes in the Siebel Repository and the DB2 Catalog are consistent.

   Review the Group names in the schema.ddl file for your existing ASCII or EBCDIC database and verify that they match the table space names in the storage control file for your existing database. If they match, proceed to Step 4. If they do not match, do the following:

   a. Run the following SQL statement to update S_TABLE with the group codes from the catalog:

      ```
      UPDATE source_tableowner_name.S_TABLE A
      SET A.GROUP_CD = (SELECT B.TSNAME
      FROM SYSIBM.SYSTABLES B
      WHERE A.NAME = B.NAME AND B.CREATOR = 'source_tableowner_name')
      ```

      where source_tableowner_name is your Siebel schema qualifier name.

   b. Remove the diccache.dat file from your SIEBSRVR_ROOT `\bin` directory (Windows) or the SIEBSRVR_ROOT/ `bin` directory (UNIX).
4. Extract the storage control file for your existing Siebel ASCII or EBCDIC database.

   The extracted file contains definitions of the physical layout of your existing Siebel database and is used to generate the DDL to build the Unicode database schema.

   For information on extracting a storage control file using the Database Configuration Utility, see *Extracting a Storage Control File from the DB2 Catalog*. Ensure that you specify the following values:

   - **Extract method.** Select the Extract from Catalog option, not the Extract from Catalog and Merge with Template option
   - **Database values.** Specify database parameter values for your existing Siebel ASCII or EBCDIC database
   - **Storage Control File.** You can assign any name to the storage control file that is extracted, for example, storage_ascii.ctl or storage_ebcdic.ctl
5. Extract the schema.ddl file for your existing Siebel ASCII or EBCDIC database.

   Ensure that your existing ASCII or EBCDIC Siebel database and repository are synchronized, then extract the schema.ddl file for your database. The schema.ddl file contains the logical definitions for all Siebel tables

and indexes for the current version of your schema and is used as an input parameter by the Database Configuration Wizard when you select the Unicode Migration option.

a. Synchronize your Siebel database and repository using the Database Configuration Utility.
For information on this task, see *Synchronizing Siebel Repository Definitions and the Physical Siebel Schema*.
Ensure that you specify the following values:
   - **Run Database Utilities option.** Select the Synchronize Schema Definition option of the Database Configuration Wizard
   - **Storage Control File.** Specify the name of the storage control file that you extracted in Step 4, for example, storage_ascii.ctl or storage_ebcdic.ctl

The Synchronize Schema Definition process generates the synchronized schema.ddl file and places it in the DBSRVR_ROOT `\db2390` (Windows) or DBSRVR_ROOT `/db2390 (UNIX)` directory.

b. Rename the schema.ddl file to one of the following, depending on the code page of your existing database:
   - ddl_ascii.ctl
   - ddl_ebcdic.ctl

6. Convert the storage control file to Unicode.
Convert the storage control file you extracted in Step 4 to Unicode format using the Siebel Database Storage Configurator (dbconf.xls). The converted storage control file is then used as an input parameter by the Database Configuration Wizard when you select the Migrate to Unicode option. For information on completing this task, see *Converting the Storage Control File to Unicode*.

Converting the Storage Control File to Unicode

As part of the process of migrating your ASCII or EBCDIC database to a Unicode format, you must convert the physical layout of your source database so that it supports Unicode. This involves expanding the size of the columns in the schema tables; the extent to which a column must be expanded is determined by the column type and length.

Once you have converted the storage control file to Unicode, you must validate the converted file. This topic outlines both tasks.

To convert the source storage control file to Unicode and validate it

1. Open the Siebel Database Storage Configurator as described in *About Modifying Storage Control Files*.
2. Click the Import tab, navigate to the directory containing your existing Siebel schema storage control file (the file you extracted in Step 4), and double-click this file to import it.
3. When the import process is completed, a message appears stating that you have imported the storage control file successfully. Click OK.
4. When the message, `Please enter default values for your system` appears, accept the default values and click Set.
5. You are prompted to indicate whether you want to import row lengths. Click Yes.
6. Navigate to the directory containing the renamed schema.ddl file for your schema (for example, ddl_ascii.ctl or ddl_ebcdic.ctl) and double-click this file to import it (this is the file you extracted and renamed in Step 5).
The Siebel Database Storage Configurator reads the ddl_codepage.ctl file to determine the type and length of each table column in the schema.
7. When the ddl_codepage.ctl file is imported successfully, click OK.

8. When the message `Would you like to import row counts` is displayed, select No.
9. Click the Structures tab, then the Database tab.
10. (Optional) Change the database name prefix of all the databases listed (for example, SIDB) to the database name prefix you want to use in your Unicode schema (for example, SIEB). The database name prefix must be the same for all database objects in the Siebel schema.
11. Click Functions, and then Convert to Unicode.
12. Click the Convert button.

    If a message is displayed stating that the spreadsheet has already been converted to Unicode, click the Structures tab and change entries in the CCSID column from Unicode to ASCII, then select the Convert to Unicode tab again.
13. When the storage control file has been converted, click the Home tab and then click the Export button.
14. Rename your converted storage control file, for example, *storage_uni.ctl*, and save it. The Siebel Database Storage Configurator validates the file.
15. Click OK and a storage control file is created for your Siebel database with an expanded Unicode schema.
16. If errors occur while the Unicode storage control file is being exported, a message is displayed and the values that need to be reviewed are highlighted.

    Make any corrections that are required before exporting the file again. For further information on this task, see *Modifying a Storage Control File Using the Database Storage Configurator*.
17. Validate the converted Unicode storage control file using the Database Configuration Utility Validate Storage File option.

    For information on validating a storage control file using the Database Configuration Utility, see *Validating an Extracted or Modified Storage Control File*.

    Ensure that you specify the following values:

    - **Configure Database option.** Select the Validate Storage File option
    - **Batch or Foreground Mode.** Choose the Batch - Generate Unload/Load option
    - **Database values.** Specify database parameter values for your existing Siebel ASCII or EBCDIC database
    - **Schema File.** Specify the ddl_codepage.ctl file you extracted in *Requirements for Migrating the Siebel Database to Unicode*
    - **Storage Control File.** Specify the storage control file you have just converted to Unicode, for example, storage_uni.ctl
18. Depending on the outcome of the validation process, do one of the following:

    - If the validation process completes successfully, proceed to *Generating the Unicode Migration Files*.
    - If the validation process fails because of buffer pool errors, follow the procedure in *Recovering from Buffer Pool Errors*, then run the validation process again.

      Errors generated during the validation process are written to a log file in the SIEBSRVR_ROOT `\log \dbconfig_validate_mf\output` directory (Windows) or the SIEBSRVR_ROOT/ `log/dbconfig_validate_mf/ output` directory (UNIX).

Recovering from Buffer Pool Errors

If the validation of the converted Unicode storage control file fails because the buffer pool sizes specified in the storage control file are too small, you must correct the errors in the storage control file and run the validation process again. This topic describes how to correct buffer pool errors.

To recover from buffer pool errors

1. Open the Siebel Database Storage Configurator as described in *About Modifying Storage Control Files*.
2. Import the storage control file you previously converted to Unicode (storage_uni.ctl) as described in *Converting the Storage Control File to Unicode*.

   > **Note:** When importing the Unicode storage control file, ensure you specify No when you are prompted to select whether or not you want to import row lengths.

3. Click Functions, and then Tools.
4. Click the Repair BP Validation button.
5. Navigate to the directory containing the log file listing the buffer pool errors and open this file. The default directory is SIEBSRVR_ROOT `\log\dbconfig_validate_mf\output` (Windows) or SIEBSRVR_ROOT `/log/dbconfig_validate_mf/output` (UNIX).
6. The Siebel Database Storage Configurator updates the buffer pool sizes. When this process completes successfully, click OK.
7. Export the updated file, save it, and validate it again as described in *Converting the Storage Control File to Unicode*.

   > **Note:** Ensure that you save the file with the same name it had when you imported it.

Generating the Unicode Migration Files

When you have extracted the original schema ddl.ctl file and converted the storage control file to Unicode for your ASCII or EBCDIC database, run the Database Configuration Utilities to generate the new Unicode schema files at the midtier.

The Database Configuration Wizard builds the new schema by reading the current schema and making the necessary adjustments to data types to convert the schema to a Unicode format. The utility also builds the load and unload control cards required to perform the migration process on the z/OS host.

This task is a step in *Roadmap for Migrating the Siebel Database to Unicode*.

To generate the Unicode schema files

1. Launch the Database Configuration Wizard and follow the steps in *Performing a Standard Installation* until the Siebel Database Operation screen is displayed Step 6).
2. From the Siebel Database Operation screen, select Run Database Utilities, and then click Next.
3. Select the Unicode Migration option, and click Next.

4. Choose the mode you want to use for the Unicode migration process.
   - **Batch - Generate Unload/Load.** Allows you to generate the Unicode migration files in batch mode. Select this option.
   - **Foreground - Generate Insert/Select.** Allows you to generate the Unicode migration files in foreground mode.
5. If you have installed more than one language pack onto the Siebel Server, the Base Language screen is displayed. Specify which language is the primary (base) language for the Siebel database, and click Next.
6. Enter the ODBC Data Source Name to connect to the target Unicode database. Click Next.
7. Enter the database user name for the target Unicode database. Click Next.
8. Enter the password associated with the database user name for the target Unicode database. Re-enter the password to confirm, then click Next.
9. Enter the Security Group ID/Grantee name, for example, `SSEROLE`, then click Next.
10. Enter the schema qualifier for the target database, and click Next.
11. On the Host/LPAR name where Target database resides screen, enter the name of the host and logical partition that will contain the new Unicode database to be migrated, for example ZM01. Click Next.
12. Enter your TSO account ID, then click Next.
    This account ID must have the authorization to allocate and create data sets on the z/OS host.
13. Enter the dataset high-level qualifier you want to use for the z/OS Unicode migration data sets, then click Next. Follow your organization's naming standards.
14. Enter the name of the directory where you want to save the migration DDL files generated. The default directory is DBSRVR_ROOT `\db2390\dboutput\unicode_migration` (Windows) or DBSRVR_ROOT `/db2390/dboutput/unicode_migration` (UNIX).
    Click Next.
15. Specify the schema.ddl file you generated in Step 5 (ddl_codepage.ctl), then click Next.
16. Enter the path and name of the storage control file you converted to Unicode (for example, storage_uni.ctl) as described in *Converting the Storage Control File to Unicode*.
17. Specify the name of the Log Output Directory and click Next.
    By default, the files are created in SIEBSRVR_ROOT `\log\unicode_migration_mf` (Windows) or SIEBSRVR_ROOT `/log/unicode_migration_mf` (UNIX).
18. Save the configuration information you have entered and launch the Siebel Upgrade wizard as described in the following topics:
    - *Running the Database Configuration Wizard on Microsoft Windows*
    - *Running the Database Configuration Wizard on UNIX*
19. Click OK.
    The Upgrade Wizard creates a number of files in the output directory you specified.
20. Apply these files on your z/OS host as described in *Transferring Migration Upgrade Files from the Midtier to the z/OS Host*.

Process of Preparing the z/OS Host Environment

This process is a step in the *Roadmap for Migrating the Siebel Database to Unicode*.

After generating the files required to perform the Unicode migration on the midtier, you must move these files to the z/OS host where they are used to create the data sets required to perform the Unicode migration.

To prepare the z/OS environment for the Unicode migration, perform the following tasks.

1. *Transferring Migration Upgrade Files from the Midtier to the z/OS Host*
2. *Creating and Allocating the Setup Data Sets*
3. *Defining Environment Variables for the Unicode Migration*

Transferring Migration Upgrade Files from the Midtier to the z/OS Host

When you run the Unicode Migration option of the Database Configuration Wizard, files are generated in the midtier output directory that you specified. You must now transfer these files from the midtier to the z/OS host where they can be executed.

This task is a step in *Process of Preparing the z/OS Host Environment*.

To transfer the Unicode migration files to the DB2 host

1. Navigate to the output directory you specified in Step 14:
   Windows: The default output directory is DBSRVR_ROOT `\db2390\dboutput\unicode_migration`
   UNIX: The default output directory is DBSRVR_ROOT `/db2390/dboutput/unicode_migration`
2. Double-click Uni_Jobsync.bat (Windows) or issue the following command from UNIX:
   ```
   ftp -vn < Uni_ftpsync.txt
   ```
3. When prompted to do so, enter the password for the TSO user name that you specified in *Generating the Unicode Migration Files*, then press any key to continue.
   The files are sent to the z/OS host and the log file, Uni_ftpsync.log, is created in your log output directory.
4. Verify that all the files referenced in the Uni_ftpsync.txt file in your Unicode migration output directory have been transferred to z/OS data sets.

Creating and Allocating the Setup Data Sets

After transferring the Unicode migration files to the z/OS host, you must generate and allocate the data sets used in the migration process. The following procedure describes how to do this.

This task is a step in *Process of Preparing the z/OS Host Environment*.

To create and allocate the Unicode migration data sets

1. Log on to the target z/OS host where the Unicode database is to be migrated. Ensure that the Procedure parameter on the logon panel is set to DB2REL8.
2. To create the Unicode migration data sets, navigate to the DSNHLQ.`SIEBEL.UNI.JOB0` data set.
3. Go to Edit mode on the data set and modify the job card, if necessary. Submit the job by typing`submit` on the command line, and pressing Enter.
4. Verify that the job ran successfully (RC is 0) and that it created the DSNHLQ.`SIEBEL.UNI.EXEC` data set. This data set contains the executable code for the Unicode migration.
5. To allocate the DSNHLQ.`SIEBEL.UNI.EXEC` REXX exec library on the ISPF Primary Option menu, select option 6 to go to the TSO command line processor panel.
6. Issue the following command to allocate the CLIST/EXEC library:

```
Alloc f(sysexec) da('DSNHLQ.SIEBEL.UNI.EXEC') shr reuse
```

7. Press Enter. The EXEC library is allocated.
   The CLIST/EXEC DSNHLQ.SIEBEL.UNI.EXEC library is used to execute all REXX execs. The panel and programs are located in this library.

Tip: The TSO command allocation for the EXEC library must be reissued with each TSO logon. Add this allocation to your personal logon CLIST to avoid having to reallocate the DSNHLQ `.SIEBEL.UNI.EXEC` library with each new TSO logon.

Defining Environment Variables for the Unicode Migration

After creating the z/OS data sets required to run the Unicode migration, you have to set up the z/OS system environment variables appropriately for the migration process. The following procedure describes how to perform this task.

This task is a step in *Process of Preparing the z/OS Host Environment*.

To set up the migration environment on the z/OS host

1. Display the Siebel Unicode Migration Option Menu panel (panel ID is UNIMIG8P) by navigating to the TSO command line processor panel, then entering the following command:
   ```
   UNIMIG82
   ```
2. To define the environment variables that will be used for the Unicode migration process, select option 1: Environment Setup and press Enter.
3. The Unicode Migration Environment Setup Menu appears. The panel ID is SBLSSETP.
4. Select option 1: Set System Environment Variables, and press Enter. The System Variable Definitions panel is displayed (panel id SBLSETVP).
5. Enter values in the following fields, or amend any of the values that are displayed, as appropriate for your environment.
   - **Source DDL From: Create Target On:**
     Enter information in the following fields to identify your source and target databases:
     - **Host/LPAR Name.** Enter the source and target MVS subsystem LPAR identifiers. For a subsystem to subsystem migration, the MVS source and target LPAR names can be the same or different, depending on where the DB2 subsystems reside. For a same subsystem migration, the MVS source and target LPAR names must be the same.
     - **DB2 Subsystem.** Enter the DB2 subsystem identifier (SSID) for the source and target databases. For a subsystem to subsystem migration, the source and target SSIDs must be different. For a same subsystem migration, the source and target SSIDs must be the same.
     - **Table Owner.** Enter the character IDs that identify the source and target Siebel Schema table owners. The source and target table owner names must be different.
   - **Source DB2 Libraries: Target DB2 Libraries**
     The default DB2 library names are displayed for your source and target databases. Amend these values as necessary.

6. Press PF3 and the values you entered are saved in the EXEC library.
7. On the Unicode Migration Environment Setup Menu select option 2: Unpack JOB1, and press Enter.
8. Submit the JCL in the data set DSNHL`Q`.`SIEBEL`.`UNI`.`JOB1`.
9. Verify that the job ran successfully (RC is 0) and that the following data set libraries were created:
    - DSNHLQ`.UNI`.`INSTALL`.`JCL`
    - DSNHLQ`.UNI`.`JCLLIB`
    - DSNHLQ`.UNI`.`LOAD`
    - DSNHLQ`.UNI`.`PROC`
10. On the Unicode Migration Environment Setup Menu select option 3: Set Jobcard Variables, and press Enter. The Jobname Prefix/Parm Definitions Jobcard Parameters panel appears; the panel ID is SBLJPXP.
11. Specify values for the following fields:
    - **Accntg.** Verify that you are using the correct Accounting class.
    - **Notify.** You can change the NOTIFY value from &SYSUID to your TSO ID or leave it as &SYSUID.

      > **Note:** You can remove the notify parameter from the job card if you choose.

    - **Scheduling.** Enter either 1: Siebel Scheduled mode or 2: Vendor Scheduled mode. For information about choosing a scheduler, see *Siebel Database Upgrade Guide for DB2 for z/OS* .
12. Enter a three-character job name prefix for the following Unicode migration job types:
    - Cross Loader
    - Rebuild Indexes
    - RunStats

    It is recommended that the three-character prefix is unique as this makes it easier to find your jobs in the queue but this is not a requirement. The remaining five characters of the job name are defined by Oracle and are unique across all Unicode migration jobs.
13. Press Enter after entering the job name prefix and parameter definitions. The JCL template files are updated. Messages indicate when each step is completed.
14. On the Unicode Migration Environment Setup Menu, select option 4: Verify target environment, and press Enter.

    A process is run to determine whether or not any objects exist in the target environment. If objects are found, you must delete them before proceeding.

    > **Note:** Running option 4: Verify target environment, might not locate all objects in a target environment. It is recommended that you perform additional checks to verify that your target environment is empty.

15. On the Unicode Migration Environment Setup Menu, select option 5: Build Log Table, and press Enter.

    This option creates and loads the temporary logging table for the Unicode migration jobs.
16. Submit the JCL in the data set DSNHLQ `.SIEBEL.UNI.INSTALL.JCL(CREATLOG)` to create the logging table.

    The DSNHLQ `.SIEBEL.SBLLOG.LOADFILE` file is also created; this file loads the initial set of jobs that were transferred to the z/OS host.
17. Verify that the job ran successfully (RC is 0). You can review the output in SDSF or another job output facility.

    The Environment Setup is now completed.
18. Press PF3 to return to the Siebel Unicode Migration Option Menu panel.

You can now proceed to build the target schema. For information on this task, see *Building the Target Schema*.

Process of Performing the Database Unicode Migration

This process is a step in the *Roadmap for Migrating the Siebel Database to Unicode*.

Once you have prepared your z/OS host system, you can migrate your ASCII or EBCDIC Siebel schema to a Unicode format by performing the following tasks in the sequence in which they are listed:

1. *Building the Target Schema*
2. *Building the Unicode Migration Jobs*
3. *Submitting the Unicode Migration Jobs*

Building the Target Schema

Complete the following procedure to build the target (Unicode) schema. When you have completed this task, you can then migrate data from the ASCII or EBCDIC schema to the Unicode schema.

This task is a step in the *Process of Performing the Database Unicode Migration*.

To build the target schema

1. On the Siebel Unicode Migration Option Menu, select option 2. Pre-Migration. The Unicode Pre-Migration Menu panel is displayed. The panel ID is SBLUPREP.
2. Select option 1. Build Target Schema to create the Unicode schema on the target database.
3. Submit the JCL in the data set DSNHLQ `.SIEBEL.UNI.INSTALL.JCL(CRESCHEM)`.

   The Build Target Schema process creates the Unicode databases, table spaces, and tables in the target DB2 subsystem.
4. On the Unicode Pre-Migration Menu, select option 2. Verify Target Schema, and press Enter.

   The Unicode migration tool reads table information from the source and target schemas, and compares the schemas to ensure that they contain the same tables.

   Differences between the schemas are logged in the `DSNHLQ .SIEBEL.UNI.DIFFRENT` data set. Review this file and resolve the cause of any differences noted.

   > **Note:** Objects in the source schema that are defined as inactive in the repository are not built in the target schema.

   The build target schema process is now completed.

Building the Unicode Migration Jobs

Once you have built your Unicode schema, you must build the appropriate DB2 Cross Loader, index rebuild, and Runstats jobs that are required to migrate your source database to your Unicode database. This topic describes how to build the data migration jobs.

This task is a step in the *Process of Performing the Database Unicode Migration*.

To build the Unicode migration jobs

1. On the Siebel Unicode Migration Option Menu, select option 2. Pre-Migration, and press Enter.
2. On the Unicode Pre-Migration Menu, select option 3. Build Jobs, and press Enter. The Unicode Pre-Migration Build Job Menu is displayed. The panel ID is SBLBLDJP.
3. Select option 1. Build Cross Loader Jobs, and press Enter.
4. Submit the JCL in the data set DSNHLQ `.SIEBEL.UNI.INSTALL.JCLLIB(SBLJCL23)` to generate the data migration jobs.
5. After submitting the job, enter cancel on the command line or press PF3 to save changes.
6. Verify that the job ran successfully (RC is 0), and that a data set, DSNHLQ `.SIEBEL.DDLIMP.UNLOAD.CNTL`, was created.

   This data set contains the Cross Loader cards and jobs that copy the data from the source to the target database.
7. On the Unicode Pre-Migration Build Job Menu, select option 2. Build Index Rebuild Jobs, and press Enter.

   A message appears, asking you to confirm that the DSNHLQ `.SIEBEL.PROC(ISPBAT)` (ISPF batch procedure) is installed.
8. Enter Y to confirm that the ISPF batch proc is installed, and press Enter.
9. When the following message appears, specify the number of indexes to be included in each rebuild job, and press Enter:

   ```
   NUMBER OF INDEXES PER REBUILD JOB.
   ```

   The maximum number of indexes that can be included in a job is 10.

   Consider your objective before choosing a maximum number of indexes for each job. Increasing this number results in fewer jobs but requires more memory and sort work. Reducing this number results in more jobs, which reduces resource requirements but causes fewer indexes to be built in parallel.
10. Submit the JCL in the data set, `DSNHLQ .SIEBEL.UNI.INSTALL.JCLLIB(SBLJCL24)`.

    > **Note:** This job builds the Index Rebuild jobs for unique, non-unique, and obsolete indexes and can take an extended period of time to complete (up to an hour or more).

11. Verify that the job ran successfully (RC is 0) and that the following data sets were created:
    - `DSNHLQ .SIEBEL.UNI.RBLDOIX.JCL`
    - `DSNHLQ .SIEBEL.UNI.RBLDOIX.SQL`
    - `DSNHLQ .SIEBEL.UNI.RBLDSIX.JCL`
    - `DSNHLQ .SIEBEL.UNI.RBLDSIX.SQL`
12. On the Unicode Pre-Migration Build Job Menu, select option 3. Build Runstat Jobs, and press Enter.

    The Upgrade RUNSTATS panel is displayed (the panel ID is SBLRSP).
13. Read the information relating to the RUNSTATS jobs on the Upgrade Runstats panel, then press Enter.
14. Submit the JCL in the data set`DSNHLQ .SIEBEL.UNI.INSTALL.JCLLIB(SBLRS)` to start the RUNSTATS job generation process. When the process is completed, the Runstat jobs are placed in the `DSNHLQ .SIEBEL.UNI.RUNST` data set.

15. Press PF3 twice to return to the Unicode Pre-Migration Menu.

    Note: Before proceeding to Step 16 to load the log table, ensure that all of the Build jobs have finished running and that they completed successfully (RC is 0).

16. Select option 4. Load Target Schema to load the Siebel logging table with the Unicode migration jobs to be executed against the target schema. Press Enter.
17. Submit the job using the JCL in data set DSNHLQ `.SIEBEL.UNI.INSTALL.JCL(LOADLOG2)`.
18. Verify that the job ran successfully, and that additional log entries for Cross Loader, index rebuild, and runstats jobs have been loaded to the log table (TMP_SBLLOG_UNI).

    The process of building the Unicode migration jobs is completed.

Submitting the Unicode Migration Jobs

When you have built your Unicode schema and generated the migration jobs, you can migrate data from the ASCII or EBCDIC Siebel schema to the new Unicode schema by submitting the appropriate DB2 Cross Loader jobs. Then you must create indexes for the new Unicode tables. This topic describes how to perform both of these tasks, and how to generate statistics for the new Unicode schema.

This task is a step in the *Process of Performing the Database Unicode Migration*.

Note: Ensure that all of the pre-migration jobs have completed successfully before you submit the Unicode migration jobs described in this topic.

To submit the Unicode migration jobs

1. On the Siebel Unicode Migration Option Menu, select option 3. Migration, and press Enter. The Unicode Migration Menu is displayed (the panel ID is SBLUMIGP). The options on the Unicode Migration Menu are used to submit the jobs that perform the schema Unicode migration.

   Submit the jobs in the sequence in which they are listed on the menu. The indexes are built after the Cross Loader jobs are completed for performance reasons. If the indexes are built before the Cross Loader jobs are run, the load jobs take much longer.
2. Select option 1. Submit Cross Loader Jobs, and press Enter. The Cross Loader jobs migrate the data from the source to the target Unicode database.
3. Submit the JCL in the data set `DSNHLQ .SIEBEL.UNI.INSTALL.JCL(SUBXLOAD)`.

   If one of the Cross Loader jobs ends abnormally during the data migration process, restart the job as described in *Resubmitting Cross Loader Jobs*.

   When all the jobs are successfully executed, the Unicode tables are loaded with data from the source schema.
4. To create unique, nonunique, and obsolete indexes for the loaded Unicode tables, select option 2. Build Indexes, and press Enter.
5. Submit the JCL in the data set `DSNHLQ .SIEBEL.UNI.INSTALL.JCL(CRESCHIX)`.
6. Verify that the job ran successfully (RC is 0 or RC is 4).

   Indexes are created on the tables in the Unicode schema.
7. To populate the indexes, select option 3. Submit Rebuild Index Jobs, and press Enter.
8. Submit the JCL in the data set `DSNHLQ .SIEBEL.UNI.INSTALL.JCL(CRESCHEM)`.
9. Verify that the job(s) ran successfully (RC is 0 or RC is 4).

The indexes on the Unicode schema are rebuilt.

10. To collect statistics for the table spaces in the new Unicode schema, select option 4. Submit RunStat Jobs, and press Enter.
11. Submit the JCL in the data set `DSNHLQ .SIEBEL.UNI.INSTALL.JCL(SUBRUNST).`

    Verify that the job(s) ran successfully (RC is 0 or RC is 4). RUNSTATS jobs are run on all table spaces in the Unicode schema.
12. Press PF3 to save changes and return to the Siebel Unicode Migration Option Menu.

    The Unicode migration process is now completed.

Resubmitting Cross Loader Jobs

This topic describes how to resubmit Cross Loader jobs that fail during the Unicode migration process.

Each Cross Loader job consists of two steps: a main step, and a verification step.

The start of the main step, which executes the Cross Loader job, is indicated as follows:

```
//XLD000
```

where `000` is the step number.

The main step is followed by the verification step, which has the following format:

```
//XLD00000 INCLUDE MEMBER=.
```

where //XLD00000 is the XLOADER card member number.

If one of the Cross Loader data migration jobs ends abnormally, correct the problem that caused the job to fail and then resubmit the job. Do not, however, rerun steps in the job that have already executed successfully; doing so causes the job to fail again, creates duplicate data in the affected table, and causes the index rebuild jobs to fail.

To restart a Cross Loader job

1. Using IBM's SDSF or another job output facility, type SJ next to the failed job.
2. Delete the Cross Loader job steps that ran successfully.
3. Submit the job again.

Recovering from Cross Loader Column Length Errors

When running the Unicode Migration Cross Loader jobs, you might receive the following error message:

```
SUM OF INPUT COLUMN LENGTHS FOR CURSOR C1 IS TOO LARGE
```

This error message is generated when the row length of the table being unloaded has exceeded the maximum allowed by Cross Loader. If you receive this message, perform the steps in the following procedure.

To recover from Cross Loader column length error messages

1. Do one of the following:

- If the definitions for the column are the same in the source and target tables, define the LONG column as a VARCHAR column of a specified length, for example:

  ```
  CAST (LOG_DATA_LONG AS VARCHAR (16350))
  ```

- If the definitions for the column in the source and target tables are different, change the definition of the LONG column in the source table to have the same format as the corresponding column in the target table, for example, the column might be defined as a CLOB data type in the target table.

2. Rerun the job that generated the error.

Viewing the Log File

During the Unicode migration process, a log file is generated that provides information on the current status of the migration jobs. You can review the log file to determine the status of a specific job using the following procedure.

To view the migration job log

1. On the Siebel Unicode Migration Option Menu, select option 4. View Log Database Joblog, and press Enter.

   The Joblog Query panel is displayed (panel ID is SBLULOGP).
2. Specify the number of jobs that you want to display on the screen at one time by entering a value in the Rows field, then select the type of jobs you want to display by entering one of the following option numbers, and pressing Enter:

   - Option 1. Lists all of the jobs that completed successfully.
   - Option 2. Lists all of the jobs that did not complete successfully.
   - Option 3. Lists the jobs that did not run.
   - Option 4. Lists all of the jobs in the job log.

The log file is initially populated with data as part of the environment setup, and is updated as jobs are run to reflect the current status of the migration jobs.

12 Migrating Data Using Siebel Enterprise Integration Manager

Migrating Data Using Siebel Enterprise Integration Manager

This chapter describes special considerations if you are using Siebel Enterprise Integration Manager (EIM) to import, export, update, merge, or delete data within Siebel CRM on DB2 for z/OS. Before using EIM, familiarize yourself with *Siebel Enterprise Integration Manager Administration Guide* . This chapter only discusses those elements of EIM operation specific to running EIM on DB2.

This chapter consists of the following topics:

- *About Setting Up EIM for DB2*
- *How to Improve EIM Performance When Importing Data*
- *Resolving Performance Degradation During the Load Process*
- *Resolving Errors in the EIM Process*

About Setting Up EIM for DB2

There are many ways you can optimize EIM data throughput. EIM is similar in function to other DB2 applications, so use regular performance monitoring tools to monitor and fine-tune performance.

Review and tune performance for each set of business data you load with EIM, for example, Opportunities, Contacts, or Products.

Choosing an Appropriate Partitioning Scheme

To achieve the best system throughput, partition base tables and EIM tables and set up parallel EIM processes to efficiently exploit partitioned EIM table spaces.

When a table is created on a partitioned table space, the table is assigned a partitioning key composed of one or more columns. EIM tables by default use the clustering U1 index, IF_ROW_BATCH_NUM, ROW_ID. In some cases, you can obtain better performance by partitioning EIM tables based on the partitioning key of the target base table, for example, on IF_ROW_BATCH_NUM plus the partitioning key of the corresponding target table. For more information on partitioning EIM tables, see *EIM Tables and Partitioning*.

> **Note:** Do not update the values in the partitioning key because you might have to perform an unload and reload or a reorganization of the data to resolve performance issues.

EIM and Table Partitioning

The mechanism by which EIM generates the ROW_ID can result in an uneven distribution of data if you use EIM with one of the default partitioning schemes to import data into a base table. For recommendations about how to avoid uneven data distribution in a partitioned table containing data imported by EIM, see *Prepartitioned Siebel Tables*.

Considerations in Planning Data Mapping

Data mapping, an important part of preparing to run EIM, is particularly important on DB2 because of the way DB2 stores data in tables and indexes. When planning your data mapping, take into account such factors as partitioning, lookups during the load phase, and searches for online transactions.

About Deciding What Data to Import

The Siebel CRM Data Model provides for many possible business needs and configurations. Many of the tables, columns, and indexes provided with Siebel CRM might not be relevant to your business. Because DB2 stores information in all columns, including NULL columns, consider carefully what data you want to import into your Siebel database. Importing unnecessary data increases the size of tables and indexes, which can affect performance. For information on removing columns and indexes and reducing column sizes using Siebel Tools, see *Customizing a Development Environment*.

MS_IDENT Column for DB2 for z/OS

The MS_IDENT column exists in every EIM table and is specific to DB2 for z/OS. It allows DB2 for z/OS to automatically generate unique and sequential values for each row in a table. This column is the unique identity column created using the Generated Always with Identity option.

> **CAUTION:** If you load data into this column using load utilities, your import might fail. Such a failure can occur particularly when migrating data from a Siebel application on DB2 for UNIX or Windows to a Siebel application on DB2 for z/OS.

How to Presort Data During an Initial Load

When loading the EIM tables with legacy data, sort this data on the partitioning key that is used on the target base table. When sorting string data, use the same character-collating sequence used in the target database. During this load, also preassign and load batch numbers corresponding to the partitioning scheme of the target base tables when possible. This method is particularly useful for those processes, such as Accounts or Contacts, where the partitioning key is based on business data rather than on a generated key. Preassigning and loading batch numbers generally improves EIM throughput, as the data is then loaded sequentially and by partition into the target base table.

> **Tip:** To further improve performance, after the EIM tables are loaded with partitioned data, unload this data in clustering order, and then reload it.

Optimal EIM Batch Size

You can regulate EIM commit frequency in several ways, including:

- Setting `COMMIT EACH TABLE` to the value `TRUE`
- Setting `COMMIT EACH PASS` to the value `TRUE`
- Adjusting batch size

Of these, the batch size most directly influences EIM performance.

The size of EIM batches can significantly affect throughput. This issue occurs as a result of the number of commits that EIM must execute. While each commit is CPU-intensive, commits release page locks, thereby releasing memory and avoiding lock escalation and timeouts.

While every installation is different, perform initial testing with large batch sizes (except DELETE EXACT, which normally runs better with smaller batch sizes). You might want to use this large-batch testing as a starting point for your own processes and modify batch size based on results.

> **Note:** Processes that populate child tables might violate the maximum number of locks allowed or might cause contention for resources in those child tables. To avoid this, reduce the batch size of these processes.

Optimal Number of Parallel Streams

A rule to determine how many batches can run simultaneously is that the combined number of parallel streams should keep the CPU 100 percent busy. These can either be multiple EIM processes running simultaneously, or the same EIM process repeated against multiple partitions of the same table.

If you run EIM processes in parallel on a DB2 database with the default setting of `UPDATE STATISTICS` set to `TRUE` in the EIM configuration file (.ifb), deadlocks can occur when multiple EIM processes access the same interface table simultaneously. To avoid a deadlock, set `UPDATE STATISTICS` to `FALSE` in the EIM configuration file.

A similar problem might occur if parallel processes access the same child tables while updating foreign keys. Therefore, analyze your EIM mappings and processes, and do not run these processes in parallel. Alternatively, you can try altering the locking level from Page to Row.

DB2 Database Logging Performance

The number of parallel streams that the DB2 for z/OS environment can support is often limited more by the DB2 logging throughput rate than by anything else. The commit process waits for a successful write to the active log and, if there is contention on the logs, performance suffers. Possible solutions to ease bottlenecks caused by logging performance include:

- Increase the number of output buffers for writing active log data sets. The size of the output buffer can be changed using the DSNZPARM parameter, OUTBUFF.
- Increase the active log size. If logs fill up faster than they can be archived, performance suffers.
- Determine the optimal number of DB2 active logs based on peak EIM loads, plus a cushion.

- Place the active log data sets on separate dedicated volumes.
- Place archive logs on a virtual tape system (VTS), or on disk, if VTS is not available.
- Define all table spaces (base tables and interface tables) with compression, which reduces logging, and decreases I/O.
- Consider data sharing if a single subsystem cannot handle the logging tasks.
- Implement data striping.

RUNSTATS Utility

If EIM performance degrades over time, consider running RUNSTATS against the base tables to update the optimizer statistics.

If you set the parameter UPDATE STATISTICS to TRUE in your EIM configuration file (.ifb), the EIM process invokes the DSNUTILS stored procedure. This procedure allows EIM to perform RUNSTATS dynamically on temporary columns to obtain optimal statistics numbers in the DB2 catalog. You must install the IBM DSNUTILS stored procedure and enable the z/OS WLM manager.

To execute EIM with UPDATE STATISTICS set to TRUE, the user account must have EXECUTE privileges on the DSNUTILS stored procedure and SYSADM, STATS, or DBADM privileges on the tables accessed. For more information on RUNSTATS, see *DB2 Statistics for Siebel CRM*.

Note: Set UPDATE STATISTICS to TRUE only once to collect proper statistics for each given EIM process. As soon as the statistic are collected, reset UPDATE STATISTICS to FALSE for that process.

SQLPROFILE Log

The SQLPROFILE parameter in the header of your EIM configuration file (.ifb) designates the file to which EIM writes an analysis of the worst-performing SQL queries overall (by total time), and the worst-performing statements in each step of the EIM process (also by total time), for example:

```
[Siebel Interface Manager]
PROCESS = Import Products
SQLPROFILE = topsql.log
```

Review this log file after the test run of each EIM process to determine where potential bottlenecks exist in your process. Based on the results of this log, either adjust mapping and IFB parameters, or perform normal DB2 tuning (such as adding an index to improve the access path).

How to Improve EIM Performance When Importing Data

There are a number of techniques you can use to improve EIM performance during import processes, particularly during your initial load of data into Siebel CRM.

For example, to improve throughput, if certain values in your database are constant across all rows of a table, use the DEFAULT COLUMN parameter to fill these rather than loading them through the EIM tables to improve throughput. For further information on improving EIM import performance, see:

- *Tuning the EIM Import Process*
- *About Improving EIM Performance During Initial Loads*

Follow the recommended import order, as described in *Siebel Enterprise Integration Manager Administration Guide* for both initial and ongoing EIM loads.

Tuning the EIM Import Process

Review and tune your EIM process for each set of business data you load, for example, Opportunities, Contacts, or Products.

To tune each EIM import process

1. Load a single batch with the following flags set in your configuration file:
    - Error flag set to 1
    - SQL flag set to 8
    - Trace flag set to 3

    Setting these flags produces a log file containing SQL statements and information about the length of time each statement took to execute.
2. Identify SQL statements that might be taking too long to execute.
3. Using the DB2 EXPLAIN utility, populate the explain table with information on how the DB2 Optimizer is executing each of these SQL statements.

    Carefully review any changes in the default optimization level in the EIM log.
4. Based on the Access Plan and other information from the EXPLAIN output, determine the required indexes for EIM and base tables.
5. Run EIM with parameters to reach each partition and repeat single process tuning.
6. Perform .ifb file tuning for each process, and remove unnecessary foreign keys.
7. Execute the REORG utility on base tables to build a compression dictionary.
8. Perform parallel process tuning.
9. Perform buffer pool tuning.
10. Execute the STOSPACE utility on base tables. This collects storage information on DB2 objects.
11. Run the RUNSTATS utility on base tables and EIM tables.
12. Use SQL explain tools to verify access paths (required indexes).
13. Modify statistics if required.
14. Remove unused indexes on base and EIM tables using the SQL DROP command.
15. Check space and resize table spaces if necessary.

About Improving EIM Performance During Initial Loads

Some considerations relating to improving EIM performance apply only during an initial load of data into your Siebel application. These considerations are described in the following sections.

Unused, Nonunique Indexes

The initial load of data is typically a database-intensive process. Each row that is inserted into the base table requires modifications, not only to the table itself, but to all its affected indexes. However, most of these indexes are never used during an initial load process.

You can usually improve performance by determining which base table indexes are unused by EIM during the import process and removing them prior to the initial load using SQL DROP commands. You must later re-create these indexes.

Unused Triggers

Removing unused triggers can improve import performance by reducing the database system overhead used to verify trigger constraints. You can use the Generate Triggers (GenTrig) component from Siebel Server Tasks to remove the triggers and to re-create them later on. This component must be in the Enabled state. For details on how to work with Generate Triggers, see *Siebel Business Process Framework: Workflow Guide* .

> **Note:** If you are using partitioning, do not remove triggers that are used for partitioning purposes from the database. If you do, your EIM process fails, or it inserts all the data in a single partition.

Free Space Parameters

If you use your Siebel application primarily for queries, updates, and deletions, alter your table spaces and indexes to provide optimal insert and update performance.

To improve the EIM import and update SQL performance, set PCTFREE for table spaces and indexes to a value of 20 or 30 prior to the EIM initial load, and maintain these settings at 20 or 30 for subsequent loads.

Insert Performance on Base Tables

You can improve the insert performance on base tables during an initial load of data into your Siebel application by following these guidelines:

- Set PCTFREE to zero for data and sequential index.
- Set FREEPAGE to zero for data and index.
- Use Page Lock if you are inserting many rows or pages sequentially.
- For a data sharing environment, set the member cluster option to reduce space map and data page P-Lock contention.

Alter base table spaces to PCTFREE and FREEPAGE values of 20 or 30 for subsequent import processes. It is also recommended that you load or reorganize any altered items before the new values take effect. The following example demonstrates how to set these values for your table spaces and indexes. Substitute your own table space and index names for your implementation.

```
ALTER TABLESPACE SIDB0401.H0401000 PART 1 PCTFREE 20;
ALTER TABLESPACE SIDB0401.H0401000 PART 2 PCTFREE 20;

ALTER INDEX SIDB0401.S_ADDR_ORG_M6 PCTFREE 20;
ALTER INDEX SIDB0401.S_ADDR_ORG_P1 PCTFREE 20;
ALTER INDEX SIDB0401.S_ADDR_ORG_P99 PART 1 PCTFREE 20;
ALTER INDEX SIDB0401.S_ADDR_ORG_P99 PART 2 PCTFREE 20;
```

Transaction Logging

Siebel transaction logging is unnecessary during an initial EIM load, and should be disabled by setting `LOG TRANSACTIONS` to `FALSE` in your .ifb file. For more information on transaction logging in EIM, see *Siebel Enterprise Integration Manager Administration Guide* .

Resolving Performance Degradation During the Load Process

When executing the EIM load process, performance might degrade noticeably after a number of batch loads are executed. This performance degradation often results when outdated statistics are loaded on the tables. To resolve the problem, update statistics on the target base tables.

You can also improve performance during the load process by updating the S_LST_OF_VAL table. The BU_ID column in the S_LST_OF_VAL base table can have only one or very few distinct values. When this happens, the DB2 optimizer often needlessly performs a table scan through all rows in the S_LST_OF_VAL table.

To avoid this problem and speed up the query, modify the statistics data by running the following SQL statements:

```
update sysibm.sysindexes set firstkeycardf=1000 where
name='S_LST_OF_VAL_M2';
update sysibm.syscolumns set colcardf = 1000 where
tbname='S_LST_OF_VAL' and name='BU_ID';
```

Note: Depending on the data you are working with, you might have to run other SQL statements first.

Resolving Errors in the EIM Process

Perform the following steps to resolve errors that can occur during your EIM process:

- If EIM aborts with the following generic error, then the user account running EIM does not have EXECUTE privileges on the DSNUTILS stored procedure:

  ```
  DSNU060I database_alias DSNUGMAP - USER username NOT AUTHORIZED FOR RUNSTATS UTILITY ON
  ```

  For information about the required permissions, see *RUNSTATS Utility*. Alternatively, you can run EIM with the UPDATE STATISTICS parameter set to `FALSE`.

- If your EIM process fails with the following error message, then the DSNZPARM parameter RETVLCFK might be set incorrectly.

  ```
  EIM-00205 Failed To Load the Application Dictionary
  ```

  Siebel CRM requires that the parameter RETVLCFK is set to `NO`. For information on DSNZPARM settings see *DSNZPARM Parameter Settings for Siebel CRM*.

- If EIM terminates during a DELETE EXACT process involving deletion of child records, the following error indicates that the maximum allowable locks have been exceeded:

  ```
  EIM Fails - Max Locks Exceeded on DELETE EXACT
  ```

  This termination causes the child records to be orphaned, because the delete to the parent table has already been committed.

  You can avoid this error by specifying the following in your .ifb file:

  ```
  COMMIT EACH PASS = FALSE
  COMMIT EACH TABLE = FALSE
  ROLLBACK ON ERROR = TRUE
  ```

13 Deployment Planning Worksheet

Deployment Planning Worksheet

Each time you install a new version of Oracle's Siebel CRM using IBM DB2 for z/OS, you must make copies of this worksheet, complete it, and give it to each member of the deployment team. This chapter includes the following topics:

- *Team Lead Summary*
- *DB2 Connect Information*
- *Siebel Database Installation Information*

Team Lead Summary

The following table provides a template for recording team lead summary information.

Role	Name
Deployment Team Lead:	
Siebel Administrator:	
Privileged User/Siebel Database User:	
DB2 Systems Programmer (SYSADM):	
DB2 Database Administrator (DBADM):	
Security Administrator:	
z/OS System Programmer:	
Midtier System Administrator:	

DB2 Connect Information

The following table provides a template for recording DB2 connect information.

Field	Value
DB2 Host Name/IP Address:	<DB2 Host Name/IP Address>
DB2 Port Number:	<DB2 Port Number>

Siebel Database Installation Information

The following table provides a template for recording Siebel Database installation information.

Item	Description
Siebel Gateway Name:	
Enterprise Server Name:	
Siebel Server Directory:	
Database Configuration Utilities Directory:	
Database Alias:	
Siebel Schema Qualifier (Max. 8 characters):	
Security Group ID / Grantee:	
Database User Name/Password:	
Siebel Administrator User Name:	
Siebel Administrator Password:	
Siebel Administrator User Group:	
Siebel User Group ID (Max. 8 characters):	
EIM User Group ID (Max. 8 characters):	
Database Name Prefix (Max. 4 characters):	

Item	Description
Storage Control File Name:	
ODBC Data Source Name:	
Storage Group for Tablespaces/Indexes:	
4-KB Buffer Pool:	
8-KB Buffer Pool:	
16-KB Buffer Pool:	
32-KB Buffer Pool:	
Index Buffer Pool:	
Code Page/CCSID:	

Note: The Security Group ID is also known as the secondary authorization ID.