Oracle Utilities Customer Care and Billing

Database Administrator's Guide

Release 2.7.0.1.0

F12135-01

January 2019
(Updated October 2019)

ORACLE

Oracle Utilities Customer Care and Billing Database Administrator's Guide, Release 2.7.0.1.0

Contents

Preface

Welcome to the Oracle Utilities Customer Care and Billing Database Administrator's Guide.

This guide provides instructions for installing and maintaining the database for Oracle Utilities Customer Care and Billing.

The preface includes the following:

Audience

This guide is intended for database administrators who will be installing and maintaining the database for Oracle Utilities Customer Care and Billing.

Related Documents

For more information, refer to these Oracle documents:

Installation Guides and Release Notes

- *Oracle Utilities Customer Care and Billing Release Notes*
- *Oracle Utilities Customer Care and Billing Quick Install Guide*
- *Oracle Utilities Customer Care and Billing Installation Guide*
- *Oracle Utilities Customer Care and Billing Database Administrator's Guide*
- *Oracle Utilities Customer Care and Billing Optional Products Installation Guide*
- *Oracle Utilities Customer Care and Billing Licensing Information User Manual*

Administrative and Business User Guides

- *Oracle Utilities Customer Care and Billing Administrative User Guide*
- *Oracle Utilities Customer Care and Billing Business User Guide*

Supplemental Documents

- *Oracle Utilities Customer Care and Billing Server Administration Guide*
- *Oracle Utilities Customer Care and Billing Security Guide*

Updates to this Documentation

This documentation is provided with the version of the product indicated. Additional and updated information about the operations and configuration of the product is available from the Knowledge Base section of My Oracle Support (http://support.oracle.com). Please refer to My Oracle Support for more information.

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.

Convention	Meaning
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

Acronyms

The following terms are used in this document:

Term	Expanded form
CCB	Oracle Utilities Customer Care and Billing
CM	Customer Modification
DBCA	Database Configuration Assistant
OUAF	Oracle Utilities Application Framework
OSDC	Oracle Software Delivery Cloud

Chapter 1

Database Overview

This chapter provides an overview of the Oracle Utilities Customer Care and Billing database, including:

- Supported Database Platforms
- Database Maintenance Rules

Supported Database Platforms

This section defines the platforms on which Oracle Utilities Customer Care and Billing is verified to operate.

Supported Platforms Summary Table

Oracle Utilities Customer Care and Billing is supported on the following platforms:

Platform	Database Versions
AIX 7.2 TL4 (POWER 64-bit)	Oracle 12.2.0.1.+ (64-bit)
Linux 7.x (64-bit) x86_64 (64-bit)	
Solaris 11.x (SPARC 64-bit)	
HP-UX 11.31 (64-bit)	

* Oracle Utilities Customer Care and Billing is tested on both Oracle Database Enterprise Edition and Standard Edition. Some features, such as Advanced Compression and Partitioning, require the Enterprise Edition.

The following Oracle Database Server Editions are supported:

- Oracle Database Enterprise Edition
- Oracle Database Standard Edition

  Note: Oracle Database Enterprise Edition and the Partitioning and Advanced Compression options are not mandatory but recommended. Standard Edition should only be considered suitable for very small, pilot projects or development environments where scalability, performance, and database size-on-disk are not important considerations. Oracle Database Enterprise Edition, including the Advanced Compression and Partitioning options, is strongly recommended in all other situations.

Refer to My Oracle Support for additional details.

Support for Software Patches and Upgrades

Due to the ongoing nature of software improvement, vendors will issue patches and service packs for the operating systems, application servers and database servers on top of specific versions that Oracle Utilities Customer Care and Billing has been tested with.

If it is necessary to apply an upgrade, please do so in a test environment that is running on the same platform as your production environment prior to updating the Oracle Utilities Customer Care and Billing production environment.

The exceptions from this rule are Hibernate version 4.0 GA and Oracle Client version 12.1.0.2. These should not be upgraded.

Always contact Oracle Utilities Customer Care and Billing Support prior to applying vendor updates that do not guarantee backward compatibility.

Database Maintenance Rules

The database supplied with the product consists of the following elements:

- A set of users to administrate, execute and read the database schema provided.
- A set of database roles to implement security for each of the users provided.
- A tablespace and a schema containing the base database objects used by the product.

The installation instructions are outlined in the installation section of this document.

Permitted Database Changes

During and after installation of the product the following changes may be performed by the database administrator personnel on site:

- Users supplied by product may be changed according to the site standards.
- Database objects may be added to the schema according to database naming standards outlined later in this document.
- Database views and indexes may be created against base database objects. Please make sure to prefix new items with "CM" (for customer modification).
- Database storage attributes for base indexes and base tables may be changed according to site standards and hardware used.
- Tablespace names, attributes and locations may be changed according to site standards.
- Database topology (base table/index to tablespace, tablespace to data file, data file to location) may be altered according to tuning and/or site standards.
- Database triggers may be created against base database objects unless they attempt to contravene base data integrity rules.
- Database initialization and parameter settings may be altered according to site standards unless otherwise advised by Oracle Support or outlined.

Non-Permitted Database Changes

In order to maintain operability and upgradeability of the product, during and after the installation of the product, the following changes may *not* be performed by the database administration personnel on site.

Base objects must not be removed or altered in the following ways:

- Columns in base tables must not be altered, removed or added in anyway.
- Columns in Indexes must not be altered or removed.
- Tables must not be renamed or removed.
- Base views must not be renamed or removed.
- Base Triggers and Sequences must not be renamed or removed.
- Base indexes must not be altered or removed.

Chapter 2

Installing the Database

This chapter provides the instructions for installing or upgrading the Oracle Utilities Customer Care and Billing database, including:

- Installation Overview
- Installing the Oracle Database

Installation Overview

Refer to Supported Database Platforms for information about the supported platforms on which Oracle Utilities Customer Care and Billing is verified to operate.

The following types of installation are available for Oracle Utilities Customer Care and Billing:

- **Initial Install** — a database with no demo data.
- **Upgrade Install** — a database upgrade to version 2.7.0.1.0 from versions 1.5.10, 1.5.15, 1.5.20, 2.0.5, 2.1.0, 2.2.0, 2.2.0.10, 2.3.1.10, 2.4.0.0, 2.4.0.1, 2.4.0.2, 2.4.0.3, 2.5.0, 2.5.0.1, 2.5.0.2, 2.6.0.0, 2.6.0.1, and 2.7.0.0.
- **Demo Install** — a database populated with demo data.

The database installation requires a supported version of the Java Development Kit Version 8.0 and Oracle 12.2.0.1(+) 32-bit client installed on the Windows 64-bit or 32-bit desktop where the install package is staged and run.

Creating the Database

For an initial install or demo install you will create an empty database on the Unix or Windows database server on which you operate the production instance of Oracle Utilities Customer Care and Billing.

1. Create the database using the Database Configuration Assistant (DBCA).

   Refer to the article *Master Note: Overview of Database Configuration Assistant (DBCA) (Doc ID 1488770.1)* on My Oracle Support for more information. Make sure to set character set for database as AL32UTF8.

   Note: While prior versions of the product have included the cdxdba programs (cdxdba.plx for UNIX or CDXDBA.exe for Windows), this is no longer supported going forward, and the Database Configuration Assistant should be used instead.

2. Enable the mandatory software options.
   - Oracle Spatial OR Oracle Locator
   - Oracle Text
3. Run the following SQL to make sure it is successful.

   ```
   SELECT COMP_NAME,STATUS FROM DBA_REGISTRY WHERE COMP_NAME IN
   ('Spatial','Oracle Text');
   ```

4. Create the default tablespace CISTS_01 and the required users and roles.

   ```
   CREATE TABLESPACE CISTS_01 LOGGING DATAFILE '/<db_file_location>/
   oradata/<DB_NAME>/cists01.dbf' SIZE 1024M REUSE AUTOEXTEND ON NEXT
   8192K MAXSIZE UNLIMITED EXTENT MANAGEMENT LOCAL UNIFORM SIZE 1M;
   ```

5. Create a role and grant privileges to the required roles.

   ```
   CREATE ROLE CIS_USER;
   CREATE ROLE CIS_READ;

   GRANT CREATE SYNONYM to CIS_USER;
   GRANT CREATE SYNONYM to CIS_READ;
   ```

6. Create the users.

```
CREATE USER CISADM IDENTIFIED BY CISADM DEFAULT TABLESPACE CISTS_01
TEMPORARY TABLESPACE TEMP PROFILE DEFAULT;
GRANT UNLIMITED TABLESPACE TO CISADM WITH ADMIN OPTION;
GRANT SELECT ANY TABLE TO CISADM;
GRANT CREATE DATABASE LINK TO CISADM;
GRANT CONNECT TO CISADM;
GRANT RESOURCE TO CISADM;
GRANT DBA TO CISADM WITH ADMIN OPTION;
GRANT CREATE ANY SYNONYM TO CISADM;
GRANT SELECT ANY DICTIONARY TO CISADM;

CREATE USER CISUSER PROFILE DEFAULT IDENTIFIED BY CISUSER DEFAULT
TABLESPACE CISTS_01 TEMPORARY TABLESPACE TEMP;
GRANT SELECT ANY TABLE TO CISUSER;
GRANT CIS_USER TO CISUSER;
GRANT CIS_READ TO CISUSER;
GRANT CONNECT TO CISUSER;

CREATE USER CISOPR PROFILE DEFAULT IDENTIFIED BY OPRPLUS DEFAULT
TABLESPACE CISTS_01 TEMPORARY TABLESPACE TEMP;
GRANT CONNECT,RESOURCE,EXP_FULL_DATABASE TO CISOPR;

CREATE USER CISREAD IDENTIFIED BY CISREAD DEFAULT TABLESPACE
CISTS_01 TEMPORARY TABLESPACE TEMP;
GRANT SELECT ANY TABLE TO CISREAD;
GRANT CIS_READ TO CISREAD;
GRANT CONNECT TO CISREAD;
```

7. Review the Storage.xml file under the FW\Install-Upgrade folder prior to an initial install or upgrade install.

   This file allocates all base tables and indexes to the default tablespace CISTS_01 and the required users and roles. Information in this file is used by ORADBI while installing the Oracle Utilities Customer Care and Billing database objects. Refer to Updating Storage.xml for more details on updating this file.

   > **Note:** You will need to review the Storage.xml file, prior to an initial install, to update the default values to custom values (for example: TableSpace Name). OraDBI can be executed by a non-schema owner in order to upgrade the database. The Initial Install still needs to be done by the schema owner.

If you decide to allocate some tables or indexes outside of the default tablespace, change the tablespace name from the default value to a custom value in the Storage.xml file.

For instance, if you decide to allocate table CI_ACCT in a tablespace MyTablespace, change Storage.xml as shown:

```
<CI_ACCT>
<TABLESPACE>MyTablespace</TABLESPACE>
</CI_ACCT>
```

For optimum storage allocation, database administrators should create multiple tablespaces with extents sized to store different types of tables/indexes. They can then edit the storage.xml file before install process, to spread tables and indexes across these tablespaces. Tables and indexes can be created in parallel by editing degree of parallelism. Tablespace, storage options, secure file options, Advanced Compression, and parallel information are used only for new objects. Therefore, for initial installs, information for

each object should be reviewed. Be careful while editing this file. Make sure that tablespace names being used exist in the database. Do not change the basic format of this file.

Note: Prior to the installation of the database schema for the product, please ensure that the Database Management System software is installed according to your site standards and the installation guide provided by the database vendor. Also please make sure that you have necessary licenses to use some of the advanced database features such as Advanced Compression.

Installing the Oracle Database

This section describes how to install the Oracle database for Oracle Utilities Customer Care and Billing 2.7.0.1.0, including:

- Database Scripts and Utilities
- Initial Install (Installing V2.7.0.1.0 for the First Time)
- Upgrade Install
- Demo Install

Note: The installation tools outlined in this guide run on Windows and UNIX/Linux only. Refer to Supported Database Platforms for more information on supported platforms.

Database Scripts and Utilities

Follow these steps before you begin installing the database:

Starting V2.6.0.0.0, installation scripts can be executed in a Linux or Windows machine. Upgrades from version beyond 2.6.0.0.0 need to set up a Microsoft Windows desktop with the Oracle Client installed.

Initial Install (Installing V2.7.0.1.0 for the First Time)

This section describes an initial installation of the v2.7.0.1.0 database. It focuses on the following:

- Copying and Decompressing Install Media
- Database Creation
- Installing the CISADM Schema

Note: You must have a supported version of the Java Development Kit installed on the Windows desktop where you stage and run the database installation package. Refer to the *Oracle Utilities Customer Care and Billing Installation Guide* for more information.

Copying and Decompressing Install Media

To copy and decompress the Oracle Utilities Customer Care and Billing database:

1. Download Oracle Utilities Application Framework V4.4.0.0.0 Oracle Database, Oracle Utilities Application Framework V4.4.0.0.0 Single Fix Prerequisite Database Rollup for CCB V2.7.0.1.0 (if there is any), Oracle Utilities Customer Care and Billing V2.7.0.1.0 Oracle Database and Oracle Utilities Customer Care and Billing V2.7.0.1.0 Single Fix Database Rollup MultiPlatform (if there is any) from the Oracle Software Delivery Cloud.

2. Copy FW-V4.4.0.0.0-Oracle-Database-Multiplatform, CCB-V2.7.0.1.0-FW-Database-PREREQ-MultiPlatform (if there is any), CCB-V2.7.0.1.0-Oracle-Database-Multiplatform and CCB-V2.7.0.1.0-Database-Rollup-MultiPlatform (if there is any) directories to your local machine.

   These files contain all the database components required to install the Oracle Utilities Application Framework and Customer Care and Billing database.

Database Creation

> **Note:** You must have Oracle Database Server installed on your machine in order to create the database. This step is not required if you are performing a database upgrade from a previous version of Oracle Utilities Customer Care and Billing.

Creating the Database on UNIX

Create the database using the Database Configuration Assistant (DBCA).

Refer to the article *Master Note: Overview of Database Configuration Assistant (DBCA) (Doc ID 1488770.1)* on My Oracle Support for more information. Make sure to set character set for database as AL32UTF8. Refer to Creating the Database for steps to create the database.

Creating the Database on Windows

You should be logged in as a user who is a member of the local ORA_DBA group on that server. The ORA_DBA group should have "administrator" privileges assigned to it.

Refer to the article *Master Note: Overview of Database Configuration Assistant (DBCA) (Doc ID 1488770.1)* on My Oracle Support for more information. Make sure to set character set for database as AL32UTF8.

Refer to Creating the Database for steps to create the database.

Database Globalization Support Consideration

Oracle Utilities Application Framework is a multilingual capable application that supports the storage, processing, and retrieval of data in multiple languages by leveraging the Oracle Database globalization support architecture. Use of the AL32UTF8 Unicode character encoding system allows the database to support multiple languages. If your application supports multiple languages with any one of which being multibyte, then consider the use of Character Length Semantics to store data in database columns in terms of CHARACTERS rather than in terms of BYTEs.

At this time, Oracle Utilities Application Framework only supports CHAR NLS_LENGTH_SEMANTICS setting at the instance level. Since this is an instance wide setting, great care should be taken and a thorough evaluation should be performed

if custom or third party components utilize the same database instance as the Framework application.

Limitations

The application will only allow for half of the number of characters if the characters are four bytes. In Java, four-byte characters consume two characters in memory. Due to Legacy Program support and performance considerations, the Framework will allocate storage based on the size defined by the Field.

For example, a Field defined as 12 characters will only be able to store 6 four-byte characters. This does not apply to two- or three-byte characters: in those cases, all 12 of the two- or three-byte characters would fit into the allocated memory. This is not a database limitation - but an application limitation.

MAX_STRING_SIZE of EXTENDED is not supported at this time.

By default, the database is created with BYTE length semantics. To store data using CHARACTER length semantics, follow the procedure below.

There are multiple ways to migrate a database from BYTE to CHAR length semantics:

- **By script**: For details, refer to the *Doc ID 313175.1* on My Oracle Support.
- **Alternative procedure**: Below is an alternate way to create a schema with character-length semantics, and then importing the date from a byte-based export.

Initial Install

1. Execute the following statement to set nls_length_semantics=CHAR.

   Note: For Container DB, it should be done on the container DB.

   ```
   SQL> ALTER SYSTEM SET nls_length_semantics=CHAR SCOPE=BOTH;
   ```

2. Restart the database.
3. Verify that the nls_length_semantics is CHAR using the following command:

   ```
   SQL> SHOW PARAMETER nls_length_semantics
   ```

   Note: For pluggable databases ensure to set nls_length_semantics=CHAR.

Migrating from BYTE Based Storage to CHARACTER Based Storage

1. Create a database using DBCA.
2. Set nls_length_semantics=CHAR.

   ```
   SQL> ALTER SYSTEM SET nls_length_semantics=CHAR SCOPE=BOTH;
   ```

3. Restart the database.
4. Ensure nls_length_semantics is CHAR.

   ```
   SQL> SHOW PARAMETER nls_length_semantics
   ```

   Note: For pluggable database ensure to set nls_length_semantics=CHAR.

5. Export schema from the database that has nls_length_semantics=BYTE.

```
expdp userid=system/<code>@<SID> directory=<DIR_NAME>
schemas=<schema_name> dumpfile=<schema_name>.dmp
logfile=<schema_name>.log
```

6. Generate DDL from dump file using Oracle impdp utility.

```
impdp userid=system/<code>@<SID> directory=<DIR_NAME>
DUMPFILE=<schema_name>.dmp SCHEMAS=<schema_name>
SQLFILE=<schema_name>_DDL.sql
```

7. Replace "Byte" with "Char" in <schema_name>DDL.sql.

   For vi editor (in Linux), use the following command to replace Byte to Char.

```
:%s/BYTE/CHAR/g
```

8. Replace the schema name also if it is required for environment.
9. Execute <schema_name>DDL.sql (generated in step 6) that creates objects in the schema.

   To ensure the number of objects at source and target are equal:

```
SQL>select OWNER || '  ' || OBJECT_TYPE || '  ' ||  COUNT(*) || '  '
||  STATUS FROM DBA_OBJECTS WHERE OWNER in ('<SCHEMA_NAMe>') GROUP
BY OWNER, OBJECT_TYPE , STATUS ORDER BY OBJECT_TYPE;
```

10. If an object is missing for any reason, create it by fixing DDL manually (DDL for each object is available in the file which was created in step 6).

    Execute DDL for the objects that are not created.

11. Generate DDL to disable triggers.

```
SQL> SELECT 'ALTER TABLE' || ' ' ||TABLE_NAME || ' ' || 'DISABLE ALL
TRIGGERS;' FROM USER_TABLES;
```

12. Execute the script generated from step 11 to disable all triggers.
13. Import the data only.

    To import data only into the schema created to support CHAR based database storage:

```
impdp userid=system/<code>@<SID> dumpfile=<schema_name>.dmp
CONTENT=DATA_ONLY SCHEMAS=<schema_name>
LOGFILE=<schema_name>_import.log
```

14. Enable the triggers.

    To generate DDL for triggers:

```
SQL>SELECT 'ALTER TABLE' || ' ' ||TABLE_NAME || ' ' || 'ENABLE ALL
TRIGGERS;' FROM USER_TABLES;
```

15. Execute the script generated from step No.14 to enable all triggers.

Installing the CISADM Schema

You should install Oracle Utilities Application Framework V4.4.0.0.0 and Oracle Utilities Application Framework V4.4.0.0.0 Single Fix Prerequisite Database Rollup for CCB V2.7.0.1.0 (if there is any) prior to installing Oracle Utilities Customer Care and Billing 2.7.0.1.0.

The files for Oracle Utilities Application Framework installers are located in the ..\FW-V4.4.0.0.0-Oracle-Database-Multiplatform\FW\Install-Upgrade folder.

The installation process prompts you for the following information:

- The target database name in which the product is to be installed.
- A database user that will own the application schema (Example: CISADM).
- A database user that has read-write (select/update/insert/delete) privileges to the objects in the application schema. (Example: CISUSER).

  The application will access the database as this user.
- A database user with read-only privileges to the objects in the application schema. (Example: CISREAD).
- A database role that has read-write (select/update/insert/delete) privileges to the objects in the application schema. The application will access the database as this user. (Example: CIS_USER).
- A database role with read-only privileges to the objects in the application schema. (Example: CIS_READ).
- Location for jar files. (The Jar files are bundled with the database package.)
- Java Home (Example: C:\Java\jdk1.8)

This section focuses on the following:

- Installing the Oracle Utilities Application Framework Database Component
- Installing Framework Prerequisite Database Single Fixes
- Installing Oracle Utilities Customer Care and Billing Database Component
- Installing Oracle Utilities Customer Care and Billing Database Rollup
- Tasks Performed by ORADBI
- Post-installation Tasks

Installing the Oracle Utilities Application Framework Database Component

Note: Oracle Utilities Application Framework Database Component can be installed using OraDBI.java. While prior versions of the product have included OraDBI.exe, this is no longer supported going forward as this does not support latest functionality/features introduced in OraDBI.java.OraDBI.jar is delivered in directory jarfiles.

This section includes the instructions to install the database component.

Installing the Oracle Utilities Application Framework Database Component Using OraDBI.java

OraDBI.java is a new tool to install and upgrade database components. It can be run from UNIX or Windows machines that has Oracle 12.1.0.2 (+) or Oracle Client 12.1.0.2 (+) installed.

Before installing the database component, ensure the following prerequisites are met.

- JDK 1.8
- Oracle Database
- Schema (such as CISADM) should exist in the database

To install the Oracle Utilities Application Framework V4.4.0.0.0:

1. Unzip FW-V4.4.0.0.0-Oracle-Database-Multiplatform.zip package.
2. Set JAVA_HOME, PATH, and CLASSPATH.

   UNIX:

   ```
   export JAVA_HOME=/scratch/software/jdk1.8.0_102/
   export PATH=$JAVA_HOME/bin:$PATH
   export CLASSPATH=/../FW-V4.4.0.0.0-Oracle- Database-
   Multiplatform/FW/jarfiles/*
   ```

   WINDOWS:

   ```
   SET JAVA_HOME=C:\Program Files\Java\jdk1.8.0_101
   SET PATH=%JAVA_HOME%\bin:%PATH%
   SET CLASSPATH= C:\ FW-V4.4.0.0.0-Oracle-Database-
   Multiplatform\FW\jarfiles\*
   ```

3. Execute the following command from the command line or command prompt from ..\FW\Install-Upgrade folder.

   There are two options available to execute OraDBI.java:

   - Using interactive mode
   - Using command on command line

   Using Interactive Mode:

   UNIX:

   ```
   java -Xmx1500M com.oracle.ouaf.oem.install.OraDBI -p
   <RW_USERPASS>,<R_USERPASS>
   ```

   WINDOWS:

   ```
   "C:\Program Files\Java\jdk1.8.0_101"\bin\java -Xmx1500M -cp C:\ FW-
   V4.4.0.0.0-Oracle-Database-Multiplatform\FW\jarfiles\*
   com.oracle.ouaf.oem.install.OraDBI -p <RW_USERPASS>,<R_USERPASS>
   ```

   The utility prompts you to enter values for the parameters listed below:

   - Enter the database server hostname:<SERVER NAME>
   - Enter the database port number:<PORT>
   - Enter the database name/SID:<DB NAME>
   - Enter your database username:<CISADM>

- Enter your password for username CISADM:
- Enter the Oracle user with read-write privileges to Database Schema:<CISUSER>
- Enter your password for username CISUSER:
- Enter the Oracle user with read-only privileges to Database Schema:<CISREAD>
- Enter your password for username CISREAD:
- Enter the database role with read-write privileges to Database Schema:<CIS_USER>
- Enter the database role with read-only privileges to Database Schema:<CIS_READ>
- Enter the name of the target Schema where you want to install or upgrade:<CISADM>

Using the Command Line:

UNIX:

```
java -Xmx1500M com.oracle.ouaf.oem.install.OraDBI -d
jdbc:oracle:thin:@<DB_SERVER>:<PORT>/
<SID>,<DBUSER>,<DBPASS>,<RW_USER>,<R_USER>,<RW_USER_ROLE>,<R_USER_
ROLE>,<DBUSER> -p <RW_USERPASS>,<R_USERPASS> -l 1,2 -q true
```

WINDOWS:

```
"C:\Program Files\Java\jdk1.8.0_101"\bin\java -Xmx1500M -cp C:\ FW-
V4.4.0.0.0-Oracle-Database-Multiplatform\FW\jarfiles\*
com.oracle.ouaf.oem.install.OraDBI -d
jdbc:oracle:thin:@<DB_SERVER>:<PORT>/
<SID>,<DBUSER>,<DBPASS>,<RW_USER>,<R_USER>,<RW_USER_ROLE>,<R_USER
ROLE>,<DBUSER> -p <RW_USERPASS>,<R_USERPASS> -l 1,2 -q true
```

This process generates log files in the directory ..\FW\Install-Upgrade\logs.

4. Ensure to check the log files for any errors.

   Note: For OraDBI java, you may receive the following message in the display output or logs. These errors can be safely ignored and the process should proceed to completion.

   - 2016-05-23 16:31:38,315 [main] ERROR (common.cryptography.KeyStoreWrapperFactory) The keystore file '<filename>' does not exist.…

   …

   This file is either provided by the property com.oracle.ouaf.system.keystore.file or expected to exist at the default file location null Attempting to use the legacy cryptography.

   - 2016-05-23 16:31:38,566 [main] INFO (oem.install.OraDBI)

Installing Framework Prerequisite Database Single Fixes

While prior versions of the product have included the cdxpatch.exe programs for applying DB Hot Fixes, this is no longer supported going forward; the ouafDatabasePatch.cmd or ouafDatabasePatch.sh should be used instead. The new tool can be run from UNIX or Windows machines.

> **Important**: Confirm if there are Prerequisite Database Single Fixes. Check if there is a CCB-V2.7.0.1.0-FW-Database-PREREQ-MultiPlatform folder. If none, skip this step and proceed to apply Oracle Utilities Customer Care and Billing, else continue with the steps below.

Applying Hot Fixes

> **Note**: Java 8 JDK should be installed on the machine to use the command. Ensure to install the JDK that is supported for your platform.

1. Extract ..\CCB-V2.7.0.1.0-FW-Database-PREREQ-MultiPlatform\FW44000-HFix\db_patch_standalone.jar to any directory on your local machine under dbpatch_tools folder.

   UNIX:

   ```
   cd /../dbpatch_tools
   jar xvf db_patch_standalone.jar
   ```

   WINDOWS:

   ```
   cd c:\..\dbpatch_tools
   jar xvf db_patch_standalone.jar
   ```

2. SET TOOLSBIN.

   UNIX:

   ```
   export TOOLSBIN=/../dbpatch_tools/bin
   ```

   WINDOWS:

   ```
   SET TOOLSBIN=c:\..\dbpatch_tools\bin
   ```

3. Apply prerequisite Oracle Utilities Application Framework database single fixes from ..\CCB-V2.7.0.1.0-FW-Database-PREREQ-MultiPlatform\FW44000-HFix folder.

   UNIX:

   a. Change the permission of ouafDatabasePatch.sh tool.

      ```
      chmod 755 ouafDatabasePatch.sh
      ```

   b. Run the ouafDatabasePatch.sh tool.

      ```
      sh ouafDatabasePatch.sh (or) ./ ouafDatabasePatch.sh
      ```

   WINDOWS:

   Run the ouafDatabasePatch.cmd tool.

   ```
   ouafDatabasePatch.cmd
   ```

The utility prompts you to enter values for the parameters listed below:

- Enter the target database type (O/M/D) [O]: <O>
- Enter the username that owns the schema: <CISADM>
- Enter the password for the cisadm user: <CISADM Password>
- Enter the name of the Oracle Database Connection String: <DB_Server:DBPORT:ORACLE_SID>

Installing Oracle Utilities Customer Care and Billing Database Component

Oracle Utilities Customer Care and Billing Database Component can be installed using OraDBI.java.While prior versions of the product have included OraDBI.exe, this is no longer supported going forward as this does not support latest functionality/features introduced in OraDBI.java. OraDBI.jar is delivered in directory jarfiles.

Installing the Oracle Utilities Customer Care and Billing Database Component Using OraDBI.java

OraDBI.java is a new tool to install and upgrade database components. It can be run from UNIX or Windows server that has Oracle 12.1.0.2 (+) or Oracle Client 12.1.0.2 (+) installed.

Before installing the database component, ensure the below prerequisites are met.

- JDK 1.8
- Oracle Database
- Schema (such as CISADM) should exist in the database

To install the Oracle Utilities Customer Care and Billing v2.7.0.1:

1. Unzip CCB-V2.7.0.1.0-Oracle-Database-Multiplatform.zip package.
2. Set JAVA_HOME, PATH, and CLASSPATH.

   UNIX:

   ```
   export JAVA_HOME=/scratch/software/jdk1.8.0_102/
   export PATH=$JAVA_HOME/bin:$PATH
   export CLASSPATH=/../FW-V4.4.0.0.0-Oracle- Database-Multiplatform/
   FW/jarfiles/*
   ```

   WINDOWS:

   ```
   SET JAVA_HOME=C:\Program Files\Java\jdk1.8.0_101
   SET PATH=%JAVA_HOME%\bin:%PATH%
   SET CLASSPATH= C:\ FW-V4.4.0.0.0-Oracle-Database-
   Multiplatform\FW\jarfiles\*
   ```

3. Execute the following command from the command line or command prompt from ..\CCB\Upgrade\Install-Upgrade folder.

   There are two options available to execute OraDBI.java

   - Using interactive mode
   - Using command on command line

Using Interactive Mode:

UNIX:

```
java -Xmx1500M com.oracle.ouaf.oem.install.OraDBI -p
<RW_USERPASS>,<R_USERPASS>
```

Windows:

```
"C:\Program Files\Java\jdk1.8.0_101"\bin\java -Xmx1500M -cp C:\
FW-V4.4.0.0.0-Oracle-Database-Multiplatform\FW\jarfiles\*
com.oracle.ouaf.oem.install.OraDBI -p
<RW_USERPASS>,<R_USERPASS>
```

The utility prompts you to enter values for the parameters listed below:

- Enter the database server hostname:<SERVER NAME>
- Enter the database port number:<PORT>
- Enter the database name/SID:<DB NAME>
- Enter your database username:<CISADM>
- Enter your password for username CISADM:
- Enter the Oracle user with read-write privileges to Database Schema:<CISUSER>
- Enter your password for username CISUSER:
- Enter the Oracle user with read-only privileges to Database Schema:<CISREAD>
- Enter your password for username CISREAD:
- Enter the database role with read-write privileges to Database Schema:<CIS_USER>
- Enter the database role with read-only privileges to Database Schema:<CIS_READ>
- Enter the name of the target Schema where you want to install or upgrade:<CISADM>

Using the Command Line:

UNIX:

```
java -Xmx1500M com.oracle.ouaf.oem.install.OraDBI -d
jdbc:oracle:thin:@<DB_SERVER>:<PORT>/
<SID>,<DBUSER>,<DBPASS>,<RW_USER>,<R_USER>,<RW_USER_ROLE>,<R_US
ER_ROLE>,<DBUSER> -p <RW_USERPASS>,<R_USERPASS> -l 1,2 -q true
```

Windows:

```
"C:\Program Files\Java\jdk1.8.0_101"\bin\java -Xmx1500M -cp C:\
FW-V4.4.0.0.0-Oracle-Database-Multiplatform\FW\jarfiles\*
com.oracle.ouaf.oem.install.OraDBI -d
jdbc:oracle:thin:@<DB_SERVER>:<PORT>/
<SID>,<DBUSER>,<DBPASS>,<RW_USER>,<R_USER>,<RW_USER_ROLE>,<R_US
ER ROLE>,<DBUSER> -p <RW_USERPASS>,<R_USERPASS> -l 1,2 -q true
```

This process generates log files in the directory ..\CCB\Upgrade\Install-Upgrade\logs.

4. Ensure to check the log files for any errors.

   Note: For OraDBI java, you may receive the following message in the display output or logs. These errors can be safely ignored and the process should proceed to completion.

   -2016-05-23 16:31:38,315 [main] ERROR (common.cryptography.KeyStoreWrapperFactory) The keystore file '<filename>' does not exist.…

   …

   This file is either provided by the property com.oracle.ouaf.system.keystore.file or expected to exist at the default file location null Attempting to use the legacy cryptography.

   - 2016-05-23 16:31:38,566 [main] INFO (oem.install.OraDBI)

Installing Oracle Utilities Customer Care and Billing Database Rollup

The previous versions of Oracle Utilities Customer Care and Billing included cdxpatch.exe programs for applying the database hot fixes. This is not supported from the current version. Starting Oracle Utilities Customer Care and Billing V2.7.0.1.0, use the ouafDatabasePatch.cmd or ouafDatabasePatch.sh. The new tool can be run from UNIX or Windows machines.

> **Important!** Proceed with the steps in this section only if the installation package contains the CCB-V2.7.0.1.0-Database-Rollup-MultiPlatform folder.

Make sure Java 8 JDK is installed on the machine to use the commands. The JDK version that is supported for your platform should be installed.

To apply Oracle Utilities Customer Care and Billing V27010 Rollup:

1. Perform the steps 1 and 2 listed in the Installing Framework Prerequisite Database Single Fixes section.
2. Apply CCB 27010 Rollup from the ..\ CCB-V2.7.0.1.0-Database-Rollup-MultiPlatform\CCB27010-HFix folder.

   UNIX:

   a. Change the permission of ouafDatabasePatch.sh tool.

      ```
      chmod 755 ouafDatabasePatch.sh
      ```

   b. Run the ouafDatabasePatch.sh tool.

      ```
      sh ouafDatabasePatch.sh or ./ ouafDatabasePatch.sh
      ```

   Windows:

   a. Run the ouafDatabasePatch.cmd tool.

      ```
      ouafDatabasePatch.cmd
      ```

The utility prompts you to enter values for the parameters listed below:

- Enter the target database type (O/M/D) [O]: <O>
- Enter the username that owns the schema: <CISADM>
- Enter the password for the cisadm user: <CISADM Password>
- Enter the name of the Oracle Database Connection String: <DB_Server:DBPORT:ORACLE_SID>

Tasks Performed by ORADBI

The tasks performed by ORADBI are as below:

- Interacts with the user to collect information about the name of Oracle account that will own the application schema (for example: CISADM), password of this account, and the name of the Oracle account that the application user will use (for example: CISUSER), and the name of the Oracle account that will be assigned read-only privileges to the application schema (for example: CISREAD).
- Verifies whether tablespace names already exist in the Storage.xml file (if not, the process will abort).
- Installs the schema, installs the system data, and configures security.
- Maintains upgrade log tables in the database.
- Updates release ID when the upgrade is completed successfully.
- If an error occurs while executing a SQL script or another utility, it logs and displays the error message and allows you to re-execute the current step.

  Log files OraDBI###.log are created in the same folder as OraDBI and contains all the SQL commands executed against the database along with the results. The log files are incremental so that the results are never overwritten. If warning messages are generated during the upgrade, OraDBI prompts the user at the end of the process. Users should check the log files to verify the warning messages.
- Warning messages are only alerts and do not necessary mean a problem exists.
- Stores the Schema owner and password in the feature configuration table. The password is stored in encrypted format.
- OraDBI can be executed by a non-schema owner.

Post-installation Tasks

- Enabling USER_LOCK Package
- Generating Database Statistics

Enabling USER_LOCK Package

For the inbound web services to work the USER_LOCK must be enabled at the database level. This is a one-time step. If this is not already enabled, please do so using the following steps.

1. Login as SYS user.

2. At the SQL prompt run:

```
@?/rdbms/admin/userlock.sql
```

3. Grant permission by running the following SQL:

```
grant execute on USER_LOCK to public;
```

Note that grant can also be made to the database user which the application connects to only instead of public. For example: cisuser

Generating Database Statistics

During an install process new database objects may be added to the target database. Before starting to use the database, generate the complete statistics for these new objects by using the DBMS_STATS package.

Upgrade Install

This section describes how to upgrade the database components for Oracle Utilities Customer Care and Billing, including:

- Copying and Decompressing Install Media
- Exclude Table/Index
- Grant Privilege to Database Roles
- Upgrading the CISADM Schema to v2.7.0.1.0

Copying and Decompressing Install Media

To copy and decompress the Oracle Utilities Customer Care and Billing database:

1. Download Oracle Utilities Application Framework V4.4.0.0.0 Oracle Database, Oracle Utilities Application Framework V4.4.0.0.0 Single Fix Prerequisite Database Rollup for CCB V2.7.0.1.0 (if there is any), Oracle Utilities Customer Care and Billing V2.7.0.1.0 Oracle Database and Oracle Utilities Customer Care and Billing V2.7.0.1.0 Single Fix Database Rollup MultiPlatform (if there is any) from the Oracle Software Delivery Cloud.
2. Copy FW-V4.4.0.0.0-Oracle-Database-Multiplatform, CCB-V2.7.0.1.0-FW-Database-PREREQ-MultiPlatform (if there is any), CCB-V2.7.0.1.0-Oracle-Database-Multiplatform and CCB-V2.7.0.1.0-Database-Rollup-MultiPlatform (if there is any) directories to your local machine. These files contain all the database components required to install the Oracle Utilities Application Framework and Customer Care and Billing database.

Exclude Table/Index

To exclude an index or table during the upgrade process:

1. Edit the OraSchUpg.inp file in the Install-Upgrade directory.
2. Add the tables and indexes in the following format:

```
-INDEX: 'INDEX_NAME','INDEX_NAME'
-TABLE: 'TBALE1_NAME','TABLE2_NAME'
```

For example: To exclude the F1_WEB_SVC table, use the following:

```
-TABLE:'F1_WEB_SVC'
```

If there are multiple tables, include them with separated commas.

```
-TABLES: 'TABLE-1','TABLE_2','TABLE_3'
```

Similarly for indexes:

```
-INDEX:' F1C064S1'
```

Grant Privilege to Database Roles

Before running the upgrade, make sure to grant the Create Synonym to the database read write (CIS_USER) and read only (CIS_READ) roles.

```
grant CREATE SYNONYM to CIS_USER;
grant CREATE SYNONYM to CIS_READ;
```

Upgrading the CISADM Schema to v2.7.0.1.0

To upgrade to 2.7.0.1.0 from Oracle Utilities Customer Care and Billing 2.5.0.2 or below, run the following steps before running the blueprint. This is intended for huge table (s) due to changes in column(s) structure from CHAR to VARCHAR2 as part of Oracle Utilities Customer To Meter enhancement.

Below is the list of the affected columns.

UOM_CD
FINAL_UOM_CD
GRAPH_UOM_CD
SQI_CD
FINAL_SQI_CD
TOU_CD
FINAL_TOU_CD
SVC_TYPE_CD
FINAL_SQI
TOU_GRP_CD
EXT_TRANSMIT_ID

Below is the list of tables that might have a high number of records.

CI_ADJ_CALC_LN
CI_BCHG_READ
CI_BCHG_SQ
CI_BCHG_UP_READ
CI_BCHG_UP_SQ
CI_BSEG_CALC_LN

CI_BSEG_ITEM
CI_BSEG_READ
CI_REG
CI_RR_STAGE_UP
CI_TREND
CI_BSEG_SQ
CI_BFCH_TOUGRP
CI_CVAL_TMPL
CI_SA_CONTERM
CI_TMAP_TYPE
CI_TOU_GRP
CI_TOU_GRP_L
CI_DEP_CTL_ST
CI_PAY_ST
CI_PAY_TNDR_ST
CI_PAY_TNDR_ST_CHAR
CI_PEVT_DTL_ST
CI_TNDR_CTL
CI_TNDR_CTL_ST
CI_TNDR_ST_EXCP

To upgrade the CISADM schema to V2.7.0.1.0:

1. Rename the table(s) and corresponding indexes before running the upgrade.
2. Run the blueprint upgrade. This creates the replacement for the tables that were renamed which contains the new column structure that is in VARCHAR2(30) except for EXT_TRANSMIT_ID which should be VARCHAR(50).
3. Copy the data from the old table to the new table using INSERT /*+APPEND*/ but already trim the data.

This section assumes an existing Oracle Utilities Customer Care and Billing on top of Oracle Utilities Application Framework. The following upgrade paths are described:

- Upgrading from V2.7.0.0.0 to V2.7.0.1.0
- Upgrading from Version 2.6.0.0.0 or 2.6.0.1.0 to 2.7.0.1.0
- Upgrading from V2.5.0, V2.5.0.1, or V2.5.0.2 to V2.7.0.1.0
- Upgrading from V2.4.0.2 or V2.4.0.3 to V2.7.0.1.0
- Upgrading from V2.4.0.0 or V2.4.0.1 to V2.7.0.1.0
- Upgrading from V2.3.1.10 to V2.7.0.1.0

- Upgrading from V2.2.0.10 to V2.7.0.1.0
- Upgrading from V2.2.0 to V2.7.0.1.0
- Upgrading from Version 2.1.0 to 2.7.0.1.0
- Upgrading from Version 2.0.5 to 2.7.0.1.0
- Upgrading from V1.5.20 to V2.7.0.1.0
- Upgrading from V1.5.10 or V1.5.15 to V2.7.0.1.0

Upgrading from V2.7.0.0.0 to V2.7.0.1.0

You must install the Oracle Utilities Application Framework V4.4.0.0.0 and Oracle Utilities Application Framework V4.4.0.0.0 Database Single Fix Prerequisite Rollup for Oracle Utilities Customer Care and Billing V2.7.0.1.0 (if there's any) prior to Oracle Utilities Customer Care and Billing V2.7.0.1.0. The files for Oracle Utilities Application Framework installer is located in ..\FW-V4.4.0.0.0-Oracle-Database-Multiplatform\FW\Install-Upgrade folder.

Upgrading the Database as Non-Schema Owner

The product allows Non-Schema owners to run the database upgrade.

To perform upgrade, the non-schema owner must have the following database grants:

- grant connect, CREATE SESSION to <Non-Schema owner>;
- grant select on <Schema owner>.CI_WFM to <Non-Schema owner>;
- grant select on <Schema owner>.CI_WFM_OPT to <Non-Schema owner>;

Installing the Oracle Utilities Application Framework Database Component Using OraDBI.java

For instructions, refer to Installing the Oracle Utilities Application Framework Database Component Using OraDBI.java.

Installing Prerequisite Database Single Fixes

For instructions, refer to Installing Framework Prerequisite Database Single Fixes.

Installing the Oracle Utilities Customer Care and Billing Database Component

For instructions, refer to Installing Oracle Utilities Customer Care and Billing Database Component.

Installing the Oracle Utilities Customer Care and Billing Database Rollup

For instructions, refer to Installing Oracle Utilities Customer Care and Billing Database Rollup.

Generating Database Statistics

During an install process new database objects may be added to the target database. Before starting to use the database, generate the complete statistics for these new objects by using the DBMS_STATS package.

Tasks Performed by ORADBI

For more information, refer to Tasks Performed by ORADBI.

Upgrading from Version 2.6.0.0.0 or 2.6.0.1.0 to 2.7.0.1.0

You must install the Oracle Utilities Application Framework V4.4.0.0.0 and Oracle Utilities Application Framework V4.4.0.0.0 Single Fix Prerequisite Database Rollup for CCB V2.7.0.1.0 (if there's any) prior to Oracle Utilities Customer Care and Billing V2.7.0.1.0. The files for Oracle Utilities Application Framework installer is located in ..\FW-V4.4.0.0.0-Oracle-Database-Multiplatform\FW\Install-Upgrade folder.

Upgrading the Database as Non-Schema Owner

The product allows Non-Schema owners to run the database upgrade.

To perform upgrade, the non-schema owner must have the following database grants:

- grant connect, CREATE SESSION to <Non-Schema owner>;
- grant select on <Schema owner>.CI_WFM to <Non-Schema owner>;
- grant select on <Schema owner>.CI_WFM_OPT to <Non-Schema owner>;

Installing the Oracle Utilities Application Framework Database Component Using OraDBI.java

For instructions, refer to Installing the Oracle Utilities Application Framework Database Component Using OraDBI.java.

Installing Prerequisite Database Single Fixes

For instructions, refer to Installing Framework Prerequisite Database Single Fixes.

Installing the Oracle Utilities Customer Care and Billing Database Component

For instructions, refer to Installing Oracle Utilities Customer Care and Billing Database Component.

Installing the Oracle Utilities Customer Care and Billing Database Rollup

For instructions, refer to Installing Oracle Utilities Customer Care and Billing Database Rollup.

Generating Database Statistics

During an install process new database objects may be added to the target database. Before starting to use the database, generate the complete statistics for these new objects by using the DBMS_STATS package.

Tasks Performed by ORADBI

For more information, refer to Tasks Performed by ORADBI.

Upgrading from V2.5.0, V2.5.0.1, or V2.5.0.2 to V2.7.0.1.0

You must install the Oracle Utilities Application Framework V4.4.0.0.0 and Oracle Utilities Application Framework V4.4.0.0.0 Single Fix Prerequisite Database Rollup for CCB V2.7.0.1.0 (if there's any) prior to Oracle Utilities Customer Care and Billing 2.7.0.1.0. The files for Oracle Utilities Application Framework installer is located in ..\FW-V4.4.0.0.0-Oracle-Database-Multiplatform\FW\Install-Upgrade folder.

Upgrading the Database as Non-Schema Owner

The product allows non-schema owners to run the database upgrade.

To perform upgrade, the non-schema owner must have the following database grants:

- grant connect, CREATE SESSION to <Non-Schema owner>;
- grant select on <Schema owner>.CI_WFM to <Non-Schema owner>;
- grant select on <Schema owner>.CI_WFM_OPT to <Non-Schema owner>;

Installing the Oracle Utilities Application Framework Database Component Using OraDBI.java

For instructions, refer to Installing the Oracle Utilities Application Framework Database Component Using OraDBI.java.

Installing Prerequisite Database Single Fixes

For instructions, refer to Installing Framework Prerequisite Database Single Fixes.

Installing the Oracle Utilities Customer Care and Billing Database Component

For instructions, refer to Installing Oracle Utilities Customer Care and Billing Database Component.

Installing the Oracle Utilities Customer Care and Billing Database Rollup

For instructions, refer to Installing Oracle Utilities Customer Care and Billing Database Rollup.

Generating Database Statistics

During an install process new database objects may be added to the target database. Before starting to use the database, generate the complete statistics for these new objects by using the DBMS_STATS package.

ORADBI Performs the Following Tasks

For more information, refer to Tasks Performed by ORADBI.

Upgrading from V2.4.0.2 or V2.4.0.3 to V2.7.0.1.0

You must install the Oracle Utilities Application Framework V4.4.0.0.0 and Oracle Utilities Application Framework V4.4.0.0.0 Single Fix Prerequisite Database Rollup for CCB V2.7.0.1.0 (if there's any) prior to Oracle Utilities Customer Care and Billing 2.7.0.1.0. The files for Oracle Utilities Application Framework installer is located in ..\FW-V4.4.0.0.0-Oracle-Database-Multiplatform\FW\Install-Upgrade folder.

Upgrading the Database as Non-Schema Owner

The product allows Non-Schema owners to run the database upgrade.

To perform upgrade, the non-schema owner must have the following database grants:

- grant connect, CREATE SESSION to <Non-Schema owner>;
- grant select on <Schema owner>.CI_WFM to <Non-Schema owner>;
- grant select on <Schema owner>.CI_WFM_OPT to <Non-Schema owner>;

Installing the Oracle Utilities Application Framework Database Component Using OraDBI.java

For instructions, refer to Installing the Oracle Utilities Application Framework Database Component Using OraDBI.java.

Installing Prerequisite Database Single Fixes

For instructions, refer to Installing Framework Prerequisite Database Single Fixes.

Installing the Oracle Utilities Customer Care and Billing Database Component

For instructions, refer to Installing Oracle Utilities Customer Care and Billing Database Component.

Installing the Oracle Utilities Customer Care and Billing Database Rollup

For instructions, refer to Installing Oracle Utilities Customer Care and Billing Database Rollup.

Generating Database Statistics

During an install process new database objects may be added to the target database. Before starting to use the database, generate the complete statistics for these new objects by using the DBMS_STATS package.

ORADBI Performs the Following Tasks

For more information, refer to Tasks Performed by ORADBI.

Upgrading from V2.4.0.0 or V2.4.0.1 to V2.7.0.1.0

This section describes the steps for upgrading Oracle Utilities Customer Care and Billing V2.4.0.0 to V2.7.0.1.0 or from V2.4.0.1 to V2.7.0.1.0.

The files for this upgrade are located in the following directory: ..\CCB-V2.7.0.1.0-Oracle-Database-Multiplatform\CCB\Upgrade\Upgrade-From-v2400-v2401\.

1. Apply Framework version 4.2.0.2 and Oracle Utilities Customer Care and Billing 2.4.0.2 from Step_1_Upgrade_to_v2402 folder:

   Note: Ensure to run OraDBI.exe from a Window 32-bit or 64-bit desktop that has the Oracle 11.1.0.1 32-bit client. The database should already be listed in the local file tnsnames.ora.

   a. Apply Framework version 4.2.0.2.0 by running ORADBI.exe from the \01_FW420_SP2 folder.

   b. Apply Framework version 4.2.0.2.0 Rollup by running CDXPATCH.exe from the \02_FW420_SP2_Rollup folder.

   c. Execute the CCB2402_Trim_SRCH_CHAR_VAL.sql script from \03_CCB_TRIM_SRCH_VAL folder.

      a. Login as CISADM user.

      b. On SQL prompt, run CCB2402_Trim_SRCH_CHAR_VAL.sql.

      ```
      @CCB2402_Trim_SRCH_CHAR_VAL.sql
      ```

      This generates a file called CCB_TRIM_SRCH_CHAR_VAL.sql.

c. Run the generated CCB_TRIM_SRCH_CHAR_VAL.sql script.

```
@CCB_TRIM_SRCH_CHAR_VAL.sql
```

d. Apply Customer Care and Billing 2.4.0 Service Pack 2 by running the ORADBI.exe from the \04_CCB240_SP2 folder.

e. Execute the FW4202_Trim_SRCH_CHAR_VAL.sql script from \05_FW_TRIM_SRCH_VAL folder.

a. Login as CISADM user.

b. On SQL prompt, run FW4202_Trim_SRCH_CHAR_VAL.sql.

```
@FW4202_Trim_SRCH_CHAR_VAL.sql
```

This will generate a file called TRIM_SRCH_CHAR_VAL.sql.

c. Run the generated TRIM_SRCH_CHAR_VAL.sql script.

```
@TRIM_SRCH_CHAR_VAL.sql
```

f. Enable USER_LOCK Package.

For in-bound Web services to work the USER_LOCK must be enabled at the database level. This is a one-time step. If this is not already enabled, please do so using the following steps:

a. Login as SYS user. On SQL prompt run:

```
@?/rdbms/admin/userlock.sql
```

b. Grant permission by running the following SQL:

```
grant execute on USER_LOCK to public;
```

Note that grant can also be made to the database user which the application connects to only instead of to public. For example: cisuser.

2. Upgrade to Oracle Utilities Customer Care and Billing 2.7.0.1.0 by following the steps in the Upgrading from V2.4.0.2 or V2.4.0.3 to V2.7.0.1.0 section.

Upgrading from V2.3.1.10 to V2.7.0.1.0

You must install the Oracle Utilities Application Framework V4.4.0.0.0 and Oracle Utilities Application Framework V4.4.0.0.0 Single Fix Prerequisite Database Rollup for CCB V2.7.0.1.0 (if there's any) prior to Oracle Utilities Customer Care and Billing 2.7.0.1.0. The files for Oracle Utilities Application Framework installer is located in ..\FW-V4.4.0.0.0-Oracle-Database-Multiplatform\FW\Install-Upgrade folder.

Note: There is a known issue with CI_BILL and CI_PAY_EVENT tables with high volume of data; hence please follow steps documented in Doc ID 2153482.1 (My Oracle Support), prior to running the upgrade.

Upgrading the Database as Non-Schema Owner

The product allows Non-Schema owners to run the database upgrade.

To perform upgrade, the non-schema owner must have the following database grants:

- grant connect, CREATE SESSION to <Non-Schema owner>;

- grant select on <Schema owner>.CI_WFM to <Non-Schema owner>;
- grant select on <Schema owner>.CI_WFM_OPT to <Non-Schema owner>;

Installing the Oracle Utilities Application Framework Database Component Using OraDBI.java

For instructions, refer to Installing the Oracle Utilities Application Framework Database Component Using OraDBI.java.

Optional: Execute CCB2401_BpSchema2.SQL

This step is recommended to improve the performance of the upgrade process.

Before executing this script, verify the script and note that these operations are long-running and the script specifies a default level of parallelism that can be tailored to the implementation's hardware. Also, note that CCB2401_BpSchema2.SQL can be executed well in advance of the upgrade to CCB 2.4.0.2 as these changes are compatible with Oracle Utilities Customer Care and Billing 2.2.0 and 2.3.1:

1. Open a command prompt.
2. Change directory to ..\CCB-V2.7.0.1.0-Oracle-Database-Multiplatform\CCB\Upgrade\Install-Upgrade.
3. Connect to SQLPLUS.
4. Execute the file as follows:

```
@CCB2401_BpSchema2.SQL
```

Optional: Execute CCB2401_BpSchema3.SQL

This step is recommended to improve the performance of the upgrade process.

Before executing this script, please verify the script and make a note that these operations are long-running and the script specifies a default level of parallelism that can be tailored to the implementation's hardware.

1. Open a command prompt.
2. Change directory to ..\CCB-V2.7.0.1.0-Oracle-Database-Multiplatform\CCB\Upgrade\Install-Upgrade.
3. Connect to SQLPLUS.
4. Execute the file as follows:

```
@CCB2401_BpSchema3.SQL
```

Installing Prerequisite Database Single Fixes

For instructions, refer to Installing Framework Prerequisite Database Single Fixes.

Installing the Oracle Utilities Customer Care and Billing Database Component

For instructions, refer to Installing Oracle Utilities Customer Care and Billing Database Component.

Installing the Oracle Utilities Customer Care and Billing Database Rollup

For instructions, refer to Installing Oracle Utilities Customer Care and Billing Database Rollup.

ORADBI Performs the Following Tasks

For more information, refer to Tasks Performed by ORADBI.

Execute CCB2401_APDATA1.sql

Before executing this script, please verify the script and make a note that these SQLs can be run in chunks across multiple sqlplus sessions in parallel. The execution process below explains how to run the script at once.

1. Open a command prompt.
2. Change directory to ..\CCB-V2.7.0.1.0-Oracle-Database-Multiplatform\CCB\Upgrade\Install-Upgrade.
3. Connect to SQLPLUS.
4. Execute the file as follows:

   ```
   @CCB2401_APDATA1.sql
   ```

Installing the upgrade script to trim the SRCH_CHAR_VAL column on the char tables

1. Login as CISADM user.
2. On SQL prompt, run FW4202_Trim_SRCH_CHAR_VAL.sql from the ..\FW-V4.4.0.0.0-Oracle-Database-Multiplatform\FW\Install-Upgrade directory.

   ```
   @FW4202_Trim_SRCH_CHAR_VAL.sql
   ```

   This generates a file called TRIM_SRCH_CHAR_VAL.sql.
3. Run the generated TRIM_SRCH_CHAR_VAL.sql script.

   ```
   @TRIM_SRCH_CHAR_VAL.sql
   ```

Enabling USER_LOCK Package

For In-bound web services to work the USER_LOCK must be enabled at the database level. This is a one-time step. If this is not already enabled please do so using the following steps.

1. Login as SYS user
2. On SQL prompt run:

   ```
   @?/rdbms/admin/userlock.sql
   ```
3. Grant permission by running the below SQL:

   ```
   grant execute on USER_LOCK to public;
   ```

Please note that grant can also be made to the database user which the application connects to only instead of to public. For example: cisuser

Generating Database Statistics

During an install process new database objects may be added to the target database. Before starting to use the database, generate the complete statistics for these new objects by using the DBMS_STATS package.

Upgrading from V2.2.0.10 to V2.7.0.1.0

This section describes the steps for upgrading Oracle Utilities Customer Care and Billing V2.2.0.10 to V2.7.0.1.0. The files for this upgrade are located in the following directory:

..\CCB-V2.7.0.1.0-Oracle-Database-Multiplatform\CCB\Upgrade\Upgrade-From-v210-v220\From-v220-Upgrade-to-v27010.

> **Note:** Ensure to run OraDBI.exe from a Window 32-bit or 64-bit desktop that has the Oracle 11.1.0.1 32-bit client. The database should already be listed in the local file tnsnames.ora.

1. Apply Framework V4.2.0.2 from the \Step_2_Upgrade_to_v2402 folder:
   a. Apply Framework version 4.2.0 Service Pack 2 by running OraDBI.exe from the \01_FW420_SP2 folder.
   b. Apply Framework version 4.2.0 Service Pack 2 rollup by running CDXPATCH.exe from the \02_FW420_SP2_Rollup folder.
2. Optional: Execute CCB2401_BpSchema2.SQL

   This step is recommended to improve the performance of the upgrade process. Refer to the Upgrading from V2.3.1.10 to V2.7.0.1.0 section on how to execute this step.
3. Optional: Execute CCB2401_BpSchema3.SQL

   This step is recommended to improve the performance of the upgrade process. Refer to the Upgrading from V2.3.1.10 to V2.7.0.1.0 section on how to execute this step.
4. Installing the upgrade script to trim the SRCH_CHAR_VAL column on the char tables. Execute the CCB2402_Trim_SRCH_CHAR_VAL.sql

   Refer to the Upgrading from V2.4.0.0 or V2.4.0.1 to V2.7.0.1.0 section on how to execute this step.
5. Apply Customer Care and Billing 2.4.0 Service Pack 2 by running the OraDBI.exe from the \04_CCB240_SP2 folder.
6. Execute CCB2401_APDATA1.sql.

   Before executing this script, please verify the script and make a note that these Refer to the Upgrading from V2.3.1.10 to V2.7.0.1.0 section on how to execute this step.
7. Installing the upgrade script to trim the SRCH_CHAR_VAL column on the char tables. Execute the FW4202_Trim_SRCH_CHAR_VAL.sql.

   Refer to the Upgrading from V2.4.0.0 or V2.4.0.1 to V2.7.0.1.0 section on how to execute this step.
8. Enable USER_LOCK Package

   Refer to the Upgrading from V2.3.1.10 to V2.7.0.1.0 section on how to execute this step.
9. Upgrade to Oracle Utilities Customer Care and Billing 2.7.0.1.0 by following the steps in the Upgrading from V2.4.0.2 or V2.4.0.3 to V2.7.0.1.0 section.

Upgrading from V2.2.0 to V2.7.0.1.0

This section describes the steps for upgrading Oracle Utilities Customer Care and Billing V2.2.0 to V2.7.0.1.0. The files for this upgrade are located in the following directory:

..\CCB-V2.7.0.1.0-Oracle-Database-Multiplatform\CCB\Upgrade\ Upgrade-From-v210-v220\ From-v220-Upgrade-to-v27010.

Note: For steps 1 and 2, be sure to run OraDBI.exe from a Window 32-bit or 64-bit desktop that has the Oracle 11.1.0.1 32-bit client. The database should already be listed in the local file tnsnames.ora

1. Apply Framework version 2.2.0 and Customer Care and Billing 2.2.0 Service Packs from the \Step_1_Apply_v220_SP10 folder:
   a. Apply Framework version 2.2.0 Service Pack 1 by running CDXDBI.exe from the \01_FW22_SP1 folder.
   b. Apply Framework version 2.2.0 Service Pack 18 by running CDXPATCH.exe from the \02_FW_220_SP18 folder.

      Note: Use the -q parameter to prevent CDXPATCH tool from prompting to apply the patch for every patch included in the service pack.
   c. Apply Framework version 220 Service Pack 18 Rollup by running the CDXPATCH.exe from the \03_FW_220_SP18_Rollup folder.

      Note: Use the -q parameter to prevent CDXPATCH tool from prompting to apply the patch for every patch included in the service pack.
   d. Apply Customer Care and Billing 2.2.0 Service Pack 10 by running the CDXPATCH.exe from the \04_CCB_220_SP10 folder.
2. Apply Framework version 4.2.0 Service Pack 2 and Customer Care and Billing 2.4.0 Service Packs 2 from the \Step_2_Upgrade_to_v2402 folder:
   a. Apply Framework version 4.2.0 Service Pack 2 by running OraDBI.exe from the \01_FW420_SP2 folder.
   b. Apply Framework version 4.2.0 Service Pack 2 rollup by running CDXPATCH.exe from the \02_FW420_SP2_Rollup folder.
   c. Optional: Execute CCB2401_BpSchema2.SQL

      This step is recommended to improve the performance of the upgrade process. Refer to the Upgrading from V2.3.1.10 to V2.7.0.1.0 section about how to execute this step.
   d. Optional: Execute CCB2401_BpSchema3.SQL

      This step is recommended to improve the performance of the upgrade process. Refer to the Upgrading from V2.3.1.10 to V2.7.0.1.0 section on how to execute this step.
   e. Installing the upgrade script to trim the SRCH_CHAR_VAL column on the char tables.
   f. Execute CCB2402_Trim_SRCH_CHAR_VAL.sql.

      Refer to the Upgrading from V2.4.0.0 or V2.4.0.1 to V2.7.0.1.0 section on how to execute this step.
   g. Apply Customer Care and Billing 2.4.0 Service Pack 2 by running the OraDBI.exe from the \04_CCB240_SP2 folder.
   h. Execute CCB2401_APDATA1.sql.

Before executing this script, please verify the script and make a note that these Refer to the Upgrading from V2.3.1.10 to V2.7.0.1.0 section on how to execute this step.

i. Installing the upgrade script to trim the SRCH_CHAR_VAL column on the char tables.

Execute the FW4202_'Trim_SRCH_CHAR_VAL.sql.

Refer to the Upgrading from V2.4.0.0 or V2.4.0.1 to V2.7.0.1.0 section on how to execute this step.

j. Enable USER_LOCK Package.

Refer to the Upgrading from V2.3.1.10 to V2.7.0.1.0 section on how to execute this step.

3. Upgrade to Oracle Utilities Customer Care and Billing 2.7.0.1.0 by following the steps in the Upgrading from V2.4.0.2 or V2.4.0.3 to V2.7.0.1.0 section.

Upgrading from Version 2.1.0 to 2.7.0.1.0

This section describes the steps for upgrading Oracle Utilities Customer Care and Billing V2.1.0 to V2.7.0.1.0. The files for this upgrade are located in the following directory: ..\CCB-V2.7.0.1.0-Oracle-Database-Multiplatform\CCB\Upgrade\Upgrade-From-v210-v220\From-v210-Upgrade-to-v27010.

Note: For steps 1-4, be sure to run OraDBI.exe from a Window 32-bit or 64-bit desktop that has the Oracle 11.1.0.1 32-bit client. The database should already be listed in the local file tnsnames.ora

1. Apply the Framework 2.1.0 and Customer Care and Billing 2.1.0 current rollups from the \ Step_1_Apply_210_Current_Rollup folder:

   a. Apply the Framework version 2.1.0 current rollup by running CDXPATCH.exe from the \01_FW210SP7_plus_Rollup folder.

   b. Apply the Customer Care and Billing version 2.1.0 current rollup by running CDXPATCH.exe from the \02_CCB210SP7_plus_Rollup folder.

2. Upgrade to Framework version 2.2.0 and Customer Care and Billing version 2.2.0 by running CDXDBI.exe from the \Step_2_Upgrade_to_v220\Upgrade-Install folder.

3. Apply Framework version 2.2.0 and Customer Care and Billing 2.2.0 Service Packs from the \Step_3_Apply_v220_SP10 folder.

   a. Apply Framework version 2.2.0 Service Pack 1 by running CDXDBI.exe from the \01_FW22_SP1 folder.

   b. Apply Framework version 2.2.0 Service Pack 18 by running CDXPATCH.exe from the \02_FW_220_SP18 folder.

   Note: Use the -q parameter to prevent CDXPATCH tool from prompting to apply the patch for every patch included in the service pack.

   c. Apply Framework version 220 Service Pack 18 Rollup by running the CDXPATCH.exe from the \03_FW_220_SP18_Rollup folder.

   Note: Use the -q parameter to prevent CDXPATCH tool from prompting to apply the patch for every patch included in the service pack.

d. Apply Customer Care and Billing 2.2.0 Service Pack 10 by running the CDXPATCH.exe from the \04_CCB_220_SP10 folder.

4. Apply Framework version 4.2.0 Service Pack 2 and Customer Care and Billing 2.4.0 Service Packs 2 from the \Step_4_Upgrade_to_v2402 folder:

   a. Apply Framework version 4.2.0 Service Pack 2 by running OraDBI.exe from the \01_FW420_SP2 folder.

   b. Apply Framework version 4.2.0 Service Pack 2 rollup by running CDXPATCH.exe from the \02_FW420_SP2_Rollup folder.

   c. Optional: Execute CCB2401_BpSchema2.SQL.

      This step is recommended to improve the performance of the upgrade process. Refer to the Upgrading from V2.3.1.10 to V2.7.0.1.0 section on how to execute this step.

   d. Optional: Execute CCB2401_BpSchema3.SQL.

      This step is recommended to improve the performance of the upgrade process. Refer to the Upgrading from V2.3.1.10 to V2.7.0.1.0 section on how to execute this step.

   e. Installing the upgrade script to trim the SRCH_CHAR_VAL column on the char tables.

      Execute CCB2402_Trim_SRCH_CHAR_VAL.sql.

      Refer to the Upgrading from V2.4.0.0 or V2.4.0.1 to V2.7.0.1.0 section on how to execute this step.

   f. Apply Customer Care and Billing 2.4.0 Service Pack 2 by running the OraDBI.exe from the \04_CCB240_SP2 folder.

   g. Execute CCB2401_APDATA1.sql.

      Before executing this script, please verify the script and make a note that these Refer to the Upgrading from V2.3.1.10 to V2.7.0.1.0 section on how to execute this step.

   h. Installing the upgrade script to trim the SRCH_CHAR_VAL column on the char tables.

      Execute the FW4202_Trim_SRCH_CHAR_VAL.sql.

      Refer to the Upgrading from V2.4.0.0 or V2.4.0.1 to V2.7.0.1.0 section on how to execute this step.

   i. Enable USER_LOCK Package

      Refer to the Upgrading from V2.3.1.10 to V2.7.0.1.0 section on how to execute this step.

5. Upgrade to Oracle Utilities Customer Care and Billing 2.7.0.1.0 by following the steps in the Upgrading from V2.4.0.2 or V2.4.0.3 to V2.7.0.1.0 section.

Upgrading from Version 2.0.5 to 2.7.0.1.0

This section describes the steps for upgrading Oracle Utilities Customer Care and Billing V2.0.5 to V2.7.0.1.0. The files for this upgrade are located in the following directory:

..\CCB-V2.7.0.1.0-Oracle-Database-Multiplatform\CCB\Upgrade\Upgrade-From-v205\

Note: For steps 1-5, be sure to run OraDBI.exe from a Window 32-bit or 64-bit desktop that has the Oracle 11.1.0.1 32-bit client. The database should already be listed in the local file tnsnames.ora.

1. Upgrade to Customer Care and Billing 2.1.0 by running CDXDBI.exe from \Step_1_Upgrade_to_v210\Upgrade-Install folder.
2. Apply the Framework 2.1.0 and Customer Care and Billing 2.1.0 current rollups from the \ Step_2_Apply_210_Current_Rollup folder:
   a. Apply the Framework V2.1.0 current rollup by running CDXPATCH.exe from the \01_FW210SP7_plus_Rollup folder.
   b. Apply the Customer Care and Billing V2.1.0 current rollup by running CDXPATCH.exe from the \02_CCB210SP7_plus_Rollup folder.
3. Upgrade to Framework V2.2.0 and Customer Care and Billing V2.2.0 by running CDXDBI.exe from the \Step_3_Upgrade_to_v220\Upgrade-Install folder.
4. Apply Framework V2.2.0 and Customer Care and Billing 2.2.0 Service Packs from the \Step_4_Apply_v220_SP10 folder:
   a. Apply Framework V2.2.0 Service Pack 1 by running CDXDBI.exe from the \01_FW22_SP1 folder.
   b. Apply Framework V2.2.0 Service Pack 18 by running CDXPATCH.exe from the \02_FW_220_SP18 folder.

      Note: Use the -q parameter to prevent CDXPATCH tool from prompting to apply the patch for every patch included in the service pack.
   c. Apply Framework version 220 Service Pack 18 Rollup by running the CDXPATCH.exe from the \03_FW_220_SP18_Rollup folder.

      Note: Use the -q parameter to prevent CDXPATCH tool from prompting if you would like to apply the patch for every patch included in the service pack.
   d. Apply Customer Care and Billing 2.2.0 Service Pack 10 by running the CDXPATCH.exe from the \04_CCB_220_SP10 folder.
5. Apply Framework version 4.2.0 Service Pack 2 and Customer Care and Billing 2.4.0 Service Pack 2 from the \Step_5_Upgrade_to_v2402 folder:
   a. Apply Framework version 4.2.0 Service Pack 2 by running OraDBI.exe from the \01_FW420_SP2 folder.
   b. Apply Framework version 4.2.0 Service Pack 2 rollup by running CDXPATCH.exe from the \02_FW420_SP2_Rollup folder.
   c. Optional: Execute CCB2401_BpSchema2.SQL

      This step is recommended to improve the performance of the upgrade process. Refer to the Upgrading from V2.3.1.10 to V2.7.0.1.0 section on how to execute this step.
   d. Optional: Execute CCB2401_BpSchema3.SQL

      This step is recommended to improve the performance of the upgrade process. Refer to the Upgrading from V2.3.1.10 to V2.7.0.1.0 section on how to execute this step.

e. Installing the upgrade script to trim the SRCH_CHAR_VAL column on the char tables.

Execute the CCB2402_Trim_SRCH_CHAR_VAL.sql

Refer to the Upgrading from V2.4.0.0 or V2.4.0.1 to V2.7.0.1.0 section on how to execute this step.

f. Apply Customer Care and Billing 2.4.0 Service Pack 2 by running the OraDBI.exe from the \04_CCB240_SP2 folder.

g. Execute CCB2401_APDATA1.sql

Before executing this script, please verify the script and make a note that these Refer to the Upgrading from V2.4.0.0 or V2.4.0.1 to V2.7.0.1.0 section on how to execute this step.

h. Installing the upgrade script to trim the SRCH_CHAR_VAL column on the char tables.

Execute the FW4202_Trim_SRCH_CHAR_VAL.sql

Refer to the Upgrading from V2.4.0.0 or V2.4.0.1 to V2.7.0.1.0 section on how to execute this step.

i. Enable USER_LOCK Package

Refer to the Upgrading from V2.3.1.10 to V2.7.0.1.0 section on how to execute this step.

6. Upgrade to Oracle Utilities Customer Care and Billing 2.7.0.1.0 by following the steps in the Upgrading from V2.4.0.2 or V2.4.0.3 to V2.7.0.1.0 section.

Upgrading from V1.5.20 to V2.7.0.1.0

This section describes the steps for upgrading Oracle Utilities Customer Care and Billing V1.5.20 to V2.7.0.1.0. The files for this upgrade are located in the following directory:

..\CCB-V2.7.0.1.0-Oracle-Database-Multiplatform\CCB\Upgrade\Upgrade-From-v1.5.10-v1.5.15\

Note: For steps 1-7, be sure to run OraDBI.exe from a Window 32-bit or 64-bit desktop that has the Oracle 11.1.0.1 32-bit client. The database should already be listed in the local file tnsnames.ora.

1. Apply Customer Care and Billing 1.5.20 Service Pack 1 by running CDXPATCH.exe from \Step_2_Apply_ServicePack_v15201 folder.
2. Upgrade to Customer Care and Billing 2.0.5 by executing the following steps from \Step_3_Upgrade_to_v205\Upgrade-Install folder.
   a. **Pre-Install**: Run CDXDBI.exe from \01_Pre-Install Folder. During this process, Owner Flag information will be upgraded from CI to C1 or F1 on system data.
   b. **Install**: Run CDXDBI.exe from \02_Install Folder. This process will complete upgrade of rest of the system data to CCB V2.0.5.
3. Upgrade to Customer Care and Billing 2.1.0 by running CDXDBI.exe from \Step_4_Upgrade_to_v210\Upgrade-Install folder.
4. Apply the Framework 2.1.0 and Customer Care and Billing 2.1.0 current rollups from the \ Step_5_Apply_210_Current_Rollup folder:

a. Apply the Framework version 2.1.0 current rollup by running CDXPATCH.exe from the \01_FW210SP7_plus_Rollup folder.

b. Apply the Customer Care and Billing version 2.1.0 current rollup by running CDXPATCH.exe from the \02_CCB210SP7_plus_Rollup folder.

5. Upgrade to Framework version 2.2.0 and Customer Care and Billing version 2.2.0 by running CDXDBI.exe from the \Step_6_Upgrade_to_v220\Upgrade-Install folder.

6. Apply Framework version 2.2.0 and Customer Care and Billing 2.2.0 Service Packs from the \Step_7_Apply_v220_SP10 folder:

   a. Apply Framework version 2.2.0 Service Pack 1 by running CDXDBI.exe from the \01_FW22_SP1 folder.

   b. Apply Framework version 2.2.0 Service Pack 18 by running CDXPATCH.exe from the \02_FW_220_SP18 folder.

      Note: Use the -q parameter to prevent CDXPATCH tool from prompting to apply the patch for every patch included in the service pack.

   c. Apply Framework version 220 Service Pack 18 Rollup by running the CDXPATCH.exe from the \03_FW_220_SP18_Rollup folder.

      Note: Use the -q parameter to prevent CDXPATCH tool from prompting to apply the patch for every patch included in the service pack.

   d. Apply Customer Care and Billing 2.2.0 Service Pack 10 by running the CDXPATCH.exe from the \04_CCB_220_SP10 folder.

7. Apply Framework version 4.2.0 Service Pack 2 and Customer Care and Billing 2.4.0 Service Packs 2 from the \Step_8_Upgrade_to_v2402 folder:

   a. Apply Framework version 4.2.0 Service Pack 2 by running OraDBI.exe from the \01_FW420_SP2 folder.

   b. Apply Framework version 4.2.0 Service Pack 2 rollup by running CDXPATCH.exe from the \02_FW420_SP2_Rollup folder.

   c. Optional: Execute CCB2401_BpSchema2.SQL

      This step is recommended to improve the performance of the upgrade process. Refer to the Upgrading from V2.3.1.10 to V2.7.0.1.0 section on how to execute this step.

   d. Optional: Execute CCB2401_BpSchema3.SQL

      This step is recommended to improve the performance of the upgrade process. Refer to the Upgrading from V2.3.1.10 to V2.7.0.1.0 section on how to execute this step.

   e. Installing the upgrade script to trim the SRCH_CHAR_VAL column on the char tables.

      Execute the CCB2402_Trim_SRCH_CHAR_VAL.sql

      Refer to the Upgrading from V2.4.0.0 or V2.4.0.1 to V2.7.0.1.0 section on how to execute this step.

   f. Apply Customer Care and Billing 2.4.0 Service Pack 2 by running the OraDBI.exe from the \04_CCB240_SP2 folder.

g. Execute CCB2401_APDATA1.sql

Before executing this script, please verify the script and make a note that these Refer to the Upgrading from V2.3.1.10 to V2.7.0.1.0 section on how to execute this step.

h. Installing the upgrade script to trim the SRCH_CHAR_VAL column on the char tables.

Execute the FW4202_Trim_SRCH_CHAR_VAL.sql

Refer to the Upgrading from V2.4.0.0 or V2.4.0.1 to V2.7.0.1.0 section on how to execute this step.

i. Enable USER_LOCK Package

Refer to the Upgrading from V2.3.1.10 to V2.7.0.1.0 section on how to execute this step.

8. Upgrade to Oracle Utilities Customer Care and Billing 2.7.0.1.0 by following the steps in the Upgrading from V2.4.0.2 or V2.4.0.3 to V2.7.0.1.0 section.

Upgrading from V1.5.10 or V1.5.15 to V2.7.0.1.0

This section describes the steps for upgrading Oracle Utilities Customer Care and Billing V1.5.10 or V1.5.15 to V2.7.0.1.0. The files for this upgrade are located in the following directory:

..\CCB-V2.7.0.1.0-Oracle-Database-Multiplatform\CCB\Upgrade\Upgrade-From-v1.5.10-v1.5.15\

Note: For steps 1-8, be sure to run OraDBI.exe from a Window 32-bit or 64-bit desktop that has the Oracle 11.1.0.1 32-bit client. The database should already be listed in the local file tnsnames.ora.

1. Upgrade to Customer Care and Billing 1.5.20 by executing the steps below from \Step_1_Upgrade_to_v1520\Upgrade-Install folder.

a. Pre-Install Steps

The clean-up scripts for each task consist of a "select" SQL and a "delete" SQL script. The "select" SQL script when executed will display the data that will be deleted by the "delete" SQL script. All the clean-up scripts spool their results in output files that have same names as the scripts but with an ".out" extension.

To execute these scripts, users must log in as a database user with delete privileges on the CC&B schema using SQLPLUS.

1. Open a command prompt.
2. Change directory to ..\01_Pre-Install Folder.
3. Connect to SQLPLUS.
4. Execute the file as follows:

```
@ delete_mdfl.sql.sql
@ delete_tddrl.sql
@ delete_tdsrt.sql
@ delete_tde.sql
```

Users must “commit” the data cleanup transaction explicitly and roll it back if the script fails for some reason.

The following scripts are included in the \01_Pre-Install folder:

- select_mdfl.sql and delete_mdfl.sql
- select_tddrl.sql and delete_tddrl.sql
- select_tdsrt.sql and delete_tdsrt.sql
- select_tde.sql and delete_tde.sql

After completion of this follow the Install steps below.

b. Install Steps

Upgrade to Customer Care and Billing version 1.5.20 by running CDXDBI.exe from \02_Install Folder.

c. Post-Install Steps

The following steps are included in the post-install process:

Sequence synchronization

In release 1.4.5, two new sequences CI_MRSTGUPID_SEQ, CI_NTUPID_SEQ were added to ID the primary key columns of CI_MR_STAGE_UP and CI_NT_UP tables. Because these tables existed in previous versions of CC&B, we must adjust the "last number" value of the new sequences to either an existing sequence that you have already been using for the same purpose or the maximum value of the primary key column of the table(s). If you have already set these sequences in Release 1.4.5 or later, skip this step and continue from the next step (if any).

Following are the steps involved in the adjust sequences process:

1. Execute the AdjustSequences.bat utility file under \03_Post-Install folder, by double-clicking it or running it from command line. The utility prompts you to enter values for following parameters:

   ```
   Enter the username that owns the CC&B schema (e.g. CISADM):
   Enter the password for the CC&B schema owner:
   Enter the name of the Oracle Database:
   ```

2. The utility connects to the database and prompts you to continue the processing.
3. The utility checks for the two new sequences in the database and for each sequence, prompts you to enter the name of an existing sequence that you have already been using to ID the primary key column of its corresponding table. You can press Enter if you are not using any existing sequence (the utility sets its value to the maximum value of the primary key in that case).

   If you choose to adjust a new sequence to an existing sequence, the utility sets the new sequence to the current "last number" value of the existing sequence, drops the existing sequence, and creates a synonym for the dropped sequence. This way the existing sequence is replaced with a CC&B sequence without breaking any existing code that may be referring to the existing sequence.
4. After sequences are adjusted, the utility reconfigures the security in the database and prompts you to enter the values for following parameters:

   ```
   Enter the Oracle user that owns the schema (e.g. CISADM):
   Enter the password for the CC&B schema:
   ```

```
Enter a comma-separated list of Oracle users in which
synonyms need to be created (e.g. cisuser,cisread):
Enter the name of the Oracle Database:
```

This completes the adjust sequences process. The process generates a log file (AdjustSequences.log). You can review the log for the actions performed and any errors that may have occurred during the process.

2. Apply Customer Care and Billing 1.5.20 Service Pack 1 by running CDXPATCH.exe from \Step_2_Apply_ServicePack_v15201 folder.
3. Upgrade to Customer Care and Billing 2.0.5 by executing the following steps from \Step_3_Upgrade_to_v205\Upgrade-Install folder.
    a. **Pre-Install:** Run CDXDBI.exe from \01_Pre-Install Folder. During this process, Owner Flag information will be upgraded from CI to C1 or F1 on system data.
    b. **Install:** Run CDXDBI.exe from \02_Install Folder. This process will complete upgrade of rest of the system data to CC&B V2.0.5.
4. Upgrade to Customer Care and Billing 2.1.0 by running CDXDBI.exe from \Step_4_Upgrade_to_v210\Upgrade-Install folder.
5. Apply the Framework 2.1.0 and Customer Care and Billing 2.1.0 current rollups from the \ Step_5_Apply_210_Current_Rollup folder:
    a. Apply the Framework version 2.1.0 current rollup by running CDXPATCH.exe from the \01_FW210SP7_plus_Rollup folder.
    b. Apply the Customer Care and Billing version 2.1.0 current rollup by running CDXPATCH.exe from the \02_CCB210SP7_plus_Rollup folder.
6. Upgrade to Framework version 2.2.0 and Customer Care and Billing version 2.2.0 by running CDXDBI.exe from the \Step_6_Upgrade_to_v220\Upgrade-Install folder.
7. Apply Framework version 2.2.0 and Customer Care and Billing 2.2.0 Service Packs from the \Step_7_Apply_v220_SP10 folder:
    a. Apply Framework version 2.2.0 Service Pack 1 by running CDXDBI.exe from the \01_FW22_SP1 folder.
    b. Apply Framework version 2.2.0 Service Pack 18 by running CDXPATCH.exe from the \02_FW_220_SP18 folder.

       Note: Use the -q parameter to prevent CDXPATCH tool from prompting to apply the patch for every patch included in the service pack.
    c. Apply Framework version 220 Service Pack 18 Rollup by running the CDXPATCH.exe from the \03_FW_220_SP18_Rollup folder.

       Note: Use the -q parameter to prevent CDXPATCH tool from prompting to apply the patch for every patch included in the service pack.
    d. Apply Customer Care and Billing 2.2.0 Service Pack 10 by running the CDXPATCH.exe from the \04_CCB_220_SP10 folder.

8. Apply Framework version 4.2.0 Service Pack 2 and Customer Care and Billing 2.4.0 Service Packs 2 from the \Step_8_Upgrade_to_v2402 folder:

    a. Apply Framework version 4.2.0 Service Pack 2 by running OraDBI.exe from the \01_FW420_SP2 folder.

    b. Apply Framework version 4.2.0 Service Pack 2 rollup by running CDXPATCH.exe from the \02_FW420_SP2_Rollup folder.

    c. Optional: Execute CCB2401_BpSchema2.SQL

    This step is recommended to improve the performance of the upgrade process. Refer to the Upgrading from V2.3.1.10 to V2.7.0.1.0 section on how to execute this step.

    d. Optional: Execute CCB2401_BpSchema3.SQL

    This step is recommended to improve the performance of the upgrade process. Refer to the Upgrading from V2.3.1.10 to V2.7.0.1.0 section on how to execute this step.

    e. Installing the upgrade script to trim the SRCH_CHAR_VAL column on the char tables.

    Execute the CCB2402_Trim_SRCH_CHAR_VAL.sql.

    Refer to the Upgrading from V2.4.0.0 or V2.4.0.1 to V2.7.0.1.0 section on how to execute this step.

    f. Apply Customer Care and Billing 2.4.0 Service Pack 2 by running the OraDBI.exe from the \04_CCB240_SP2 folder.

    g. Execute CCB2401_APDATA1.sql.

    Before executing this script, please verify the script and make a note that these Refer to the Upgrading from V2.3.1.10 to V2.7.0.1.0 section on how to execute this step.

    h. Installing the upgrade script to trim the SRCH_CHAR_VAL column on the char tables.

    Execute the FW4202_Trim_SRCH_CHAR_VAL.sql.

    Refer to the Upgrading from V2.4.0.0 or V2.4.0.1 to V2.7.0.1.0 section on how to execute this step.

    i. Enable USER_LOCK Package.

    Refer to the Upgrading from V2.3.1.10 to V2.7.0.1.0 section on how to execute this step.

9. Upgrade to Oracle Utilities Customer Care and Billing 2.7.0.1.0 by following the steps in the Upgrading from V2.4.0.2 or V2.4.0.3 to V2.7.0.1.0 section.

Demo Install

This section describes how to install the demo database components for Oracle Utilities Customer Care and Billing, including:

- Copying and Decompressing Install Media
- Creating the Database

- Importing the Demo Dump File
- Configuring Security

Copying and Decompressing Install Media

To copy and decompress the Oracle Utilities Customer Care and Billing database:

1. Download Oracle Utilities Application Framework V4.4.0.0.0 Oracle Database, Oracle Utilities Application Framework V4.4.0.0.0 Single Fix Prerequisite Database Rollup for CCB V2.7.0.1.0 (if there's any), Oracle Utilities Customer Care and Billing V2.7.0.1.0 Oracle Database and Oracle Utilities Customer Care and Billing V2.7.0.1.0 Single Fix Database Rollup MultiPlatform (if there is any) from the Oracle Software Delivery Cloud.
2. Copy FW-V4.4.0.0.0-Oracle-Database-Multiplatform, CCB-V2.7.0.1.0-FW-Database-PREREQ-MultiPlatform (if there's any), CCB-V2.7.0.1.0-Oracle-Database-Multiplatform and CCB-V2.7.0.1.0-Database-Rollup-MultiPlatform (if there is any) directories to your local machine.

   These files contain all the database components required to install the Oracle Utilities Application Framework and Customer Care and Billing database.

Creating the Database

> **Note:** You must have Oracle Database Server 12.1.0.2+ installed on your machine to create the database.

It is strongly recommended to use DBCA to create the database.

Creating the Database on UNIX

Create the database using the Database Configuration Assistant (DBCA).

Refer to the article *Master Note: Overview of Database Configuration Assistant (DBCA) (Doc ID 1488770.1)* on My Oracle Support for more information. Make sure to set character set for database as AL32UTF8.

Refer to the Creating the Database section for steps to create the database.

Creating the Database on Windows

You should be logged in as a user who is a member of the local ORA_DBA group on that server. The ORA_DBA group should have "administrator" privileges assigned to it.

Refer to the article *Master Note: Overview of Database Configuration Assistant (DBCA) (Doc ID 1488770.1)* on My Oracle Support for more information. Make sure to set character set for database as AL32UTF8.

Refer to the Creating the Database section for steps to create the database.

Database Storage BYTES / CHARACTER

Database created by default will store data in BYTES. To store data in CHARACTER follow the procedure below:

Initial Install

1. Execute the following statement to set nls_length_semantics=CHAR.

   ```
   SQL> ALTER SYSTEM SET nls_length_semantics=CHAR SCOPE=BOTH;
   ```

2. Restart the database.
3. Ensure nls_length_semantics is CHAR. Perform the following command:

   ```
   SQL> SHOW PARAMETER nls_length_semantics
   ```

   Note: For pluggable database ensure to set nls_length_semantics=CHAR for both container and pluggable database.

Upgrade and Migration from BYTE Based Storage to CHARACTER Based Storage

1. Create a database using DBCA.
2. Execute following statement to set nls_length_semantics=CHAR.

   ```
   SQL> ALTER SYSTEM SET nls_length_semantics=CHAR SCOPE=BOTH;
   ```

3. Restart the database.
4. Ensure nls_length_semantics is CHAR using the following command:

   ```
   SQL> SHOW PARAMETER nls_length_semantics
   ```

   Note: For pluggable database ensure to set nls_length_semantics=CHAR for container and pluggable database both.

5. Export schema from the database that has nls_length_semantics=BYTE.

   ```
   expdp userid=system/<code>@<SID> directory=<DIR_NAME>
   schemas=<schema_name> dumpfile=<schema_name>.dmp
   logfile=<schema_name>.log
   ```

6. Generate DDL from dump file using Oracle impdp utility.

   ```
   impdp userid=system/<code>@<SID> directory=<DIR_NAME>
   DUMPFILE=<schema_name>.dmp SCHEMAS=<schema_name>
   SQLFILE=<schema_name>_DDL.sql
   ```

7. Replace “Byte” with “Char” in <schema_name>DDL.sql.

   For vi editor (in Linux), use the following command to replace Byte to Char.

   ```
   :%s/BYTE/CHAR/g
   ```

8. Replace the schema name also if it is required for environment.
9. Execute <schema_name>DDL.sql (generated in step 6) that creates objects in the schema.

   Execute the following command to ensure the number of objects at source and target are equal.

```
SQL>select OWNER || '  ' || OBJECT_TYPE || '  ' ||  COUNT(*) || '  '
||  STATUS FROM DBA_OBJECTS WHERE OWNER in ('<SCHEMA_NAMe>') GROUP
BY OWNER, OBJECT_TYPE , STATUS ORDER BY OBJECT_TYPE;
```

10. If an object is missing for any reason, create it by fixing DDL manually (DDL for each object is available in the file which was created in step 6).

    Execute DDL for the objects that are not created.

11. Generate DDL to disable triggers using following command:

```
SQL> SELECT 'ALTER TABLE' || ' ' ||TABLE_NAME || ' ' || 'DISABLE ALL
TRIGGERS;' FROM USER_TABLES;
```

12. Execute the script generated from step 11 to disable all triggers.
13. Import the data only.

    Use the following command to import data only into the schema created to support CHAR based database storage.

```
impdp userid=system/<code>@<SID> dumpfile=<schema_name>.dmp
CONTENT=DATA_ONLY SCHEMAS=<schema_name>
LOGFILE=<schema_name>_import.log
```

14. Enable the triggers.

    Use the following command to generate DDL for triggers.

```
SQL>SELECT 'ALTER TABLE' || ' ' ||TABLE_NAME || ' ' || 'ENABLE ALL
TRIGGERS;' FROM USER_TABLES;
```

15. Execute the script generated from step No.14 to enable all triggers.

Importing the Demo Dump File

After a successful database creation, demo data can also be imported by using by following these steps:

1. Set the correct ORACLE_SID and ORACLE_HOME.
2. Run following command to import demo dump:

   NOTE: Ensure the ..\CCB-V2.7.0.1.0-Oracle-Database-Multiplatform/CCB/Demo/exp_demo.dmp.gz file is extracted and available in data_pump_dir's location before running the below import command.

```
impdp directory= data_pump_dir dumpfile= exp_demo.dmp
logfile=exp_demo.log schemas=CISADM
```

Configuring Security

The configuration security utility and scripts are already part of the delivered jarfiles in ..\FW-V4.4.0.0.0-Oracle-Database-Multiplatform\FW\jarfiles directory. It can be executed from a Linux or a Windows machine.

Please note the following:

- Database vault must be disabled before running.
- Interactive mode for this utility is currently not working.

- `-f` parameter is not working and will be deprecated soon.

The utility configures security for the application owner schema objects.

Oragensec, by default, grantS permissions to CIS_USER and CIS_READ roles. To use site-specific roles, execute Oragensec after providing the command line options and specifying the specific roles.

OraGenSec Java Usage

usage: java OraGenSec [-a <arg>] [-d <arg>] [-f <arg>] [-h] [-l <arg>] [-o <arg>] [-p <arg>] [-q] [-r <arg>] [-u <arg>]

```
OraGenSec Help
-a <arg>     generate security for All objects in the database
-d <arg>     db connection as: db_host:db_port/db_service
-f <arg>     generate security for specific objects from an input
File
-h           help
-l <arg>     log file
-o <arg>     generate security for comma separated list of objects
corresponding passwords of users to create synonyms
-p <arg>
forsilent mode
-q
-r <arg>roles as: CIS_READ,CIS_USER
-u <arg>comma-separated list of users to create synonyms for
End of OraGenSec Help
```

To run the utility:

1. Set JAVA_HOME, PATH, and CLASSPATH.

   UNIX:

   ```
   export JAVA_HOME=/scratch/software/jdk1.8.0_102/
   export PATH=$JAVA_HOME/bin:$PATH
   export CLASSPATH=/../FW-V4.4.0.0.0-Oracle-
   Database-Multiplatform/FW/jarfiles/*
   ```

   WINDOWS:

   ```
   SET JAVA_HOME=C:\Program Files\Java\jdk1.8.0_101
   SET PATH=%JAVA_HOME%\bin:%PATH%
   SET CLASSPATH= C:\ FW-V4.4.0.0.0-Oracle-Database-
   Multiplatform\FW\jarfiles\*
   ```

2. From any directory you can execute Oragensec as long as Step 1 is done. Execute the following command:

   UNIX:

   ```
   java -Xmx1500M com.oracle.ouaf.oem.install.OraGenSec-d
   <DBUSER>,<DBPASS>, jdbc:oracle:thin:@<DB_SERVER>:<PORT>/
   <SID> -u <RW_USER>,<R_USER> -r <RW_USER_ROLE>,<R_USER_ROLE> -a A -p
   <RW_USERPASS>,<R_USERPASS> -l <Logfile Name>
   ```

   WINDOWS:

   ```
   "%JAVA_HOME%"\bin\java -Xmx1500M
   com.oracle.ouaf.oem.install.OraGenSec -d <DBUSER>,<DBPASS>,
   jdbc:oracle:thin:@<DB_SERVER>:<PORT>/
   <SID> -u <RW_USER>,<R_USER> -r <RW_USER_ROLE>,<R_USER_ROLE> -a A -p
   <RW_USERPASS>,<R_USERPASS> -l <Logfile Name>
   ```

Chapter 3

Database Design

The standard for database objects such as tables, columns, and indexes, for products using the Oracle Utilities Application Framework helps in smooth integration and upgrade processes by ensuring clean database design, promoting communications, and reducing errors.

Just as Oracle Utilities Application Framework goes through innovation in every release of the software, it is also inevitable that the product will take advantage of various database vendors' new features in each release. The recommendations in the database installation section include only the ones that have been proved by vigorous QA processes, field tests and benchmarks.

This chapter describes the following:

Database Object Standard

This section discusses the rules applied to naming database objects and the attributes that are associated with these objects.

Categories of Data

A table can belong to one of the three categories:

- Control (admin)
- Master
- Transaction

For purposes of physical table space design, metadata and control tables can belong to the same category.

Example of tables in each category:

- **Control:** SC_USER, CI_ADJ_TYPE, F1_BUS_OBJ
- **Master:** CI_PER, CI_PREM,
- **Transaction:** F1_FACT, CI_FT

All tables have the category information in their index name. The second letter of the index carries this information. Refer to Indexes for more information.

Naming Standards

The following naming standards must be applied to database objects.

Table

Table names are prefixed with the owner flag value of the product. For customer modification **CM** must prefix the table name. The length of the table names must be less than or equal to 30 characters. A language table should be named by suffixing **_L** to the main table. The key table name should be named by suffixing **_K** to the main table.

It is recommended to start a table name with the 2-3 letter acronym of the subsystem name that the table belongs to. For example, **MD** stands for metadata subsystem and all metadata table names start with **CI_MD**.

Some examples are:

- CI_ADJ_TYPE
- CI_ADJ_TYPE_L

A language table stores language sensitive columns such as a description of a code. The primary key of a language table consists of the primary key of the code table plus language code (LANGAGUE_CD).

A key table accompanies a table with a surrogate key column. A key value is stored with the environment id that the key value resides in the key table.

The tables prior to V2.0.0 are prefixed with CI_ or SC_.

Columns

The length of a column name must be less than or equal to 30 characters. For customer modification, CM must prefix the column name. The following conventions apply when you define special types of columns in the database.

- Use the suffix **FLG** to define a lookup table field. Flag columns must be CHAR(4). Choose lookup field names carefully as these column names are defined in the lookup table (CI_LOOKUP_FLD) and must be prefixed by the product owner flag value.
- Use the suffix **CD** to define user-defined codes. User-defined codes are primarily found as the key column of the admin tables.
- Use the suffix **ID** to define system assigned key columns.
- Use the suffix **SW** to define Boolean columns. The valid values of the switches are 'Y' or 'N'. The switch columns must be CHAR(1)
- Use the suffix **DT** to define Date columns.
- Use the suffix **DTTM** to define Date Time columns.
- Use the suffix **TM** to define Time columns.

Some examples are:

- ADJ_STATUS_FLG
- CAN_RSN_CD

Indexes

Index names are composed of the following parts:

[OF][*application specific prefix*][C/M/T]NNN[P/S]n

- **OF**- Owner Flag. The standard is to use the two characters of the product's owner flag. Note that there may be some older indexes that use only the first character of the owner flag. For client specific implementation of index, use CM for Owner Flag. If implementation creates a CM Index on table-columns for which the base product already provides an index, then the CM Index will be overridden by the based index.
- Application specific prefix could be C, F, T or another letter.
- **C/M/T** - The second character can be either C or M or T. C is used for control tables (Admin tables). M is for the master tables. T is reserved for the transaction tables.
- **NNN** - A three-digit number that uniquely identifies the table on which the index is defined.
- **P/S** - P indicates that this index is the primary key index. S is used for indexes other than primary keys.
- **n** is the index number, unique across all indexes on a given table (0 for primary and 1, 2, etc., for the secondary indexes).

Some examples are:

- F1C066P0
- F1C066S1

- XT206C2
- CMT206S2

  Warning! Do not use index names in the application as the names can change due to unforeseeable reasons.

Updating Storage.xml

The storage.xml file that comes with the product allocates all base tables and indexes to the default tablespace CISTS_01. If you decide to allocate some tables or indexes outside of the default tablespace, then this has to be reflected in the storage.xml file by changing the tablespace name from the default value to a custom value, according to the format shown below:

Format:

```
<Table_Name>
  <TABLESPACE>CISTS_01</TABLESPACE>
  <PARALLEL>1</PARALLEL>
- <LOB>
- <Column Name>
  <TABLESPACE>CISTS_01</TABLESPACE>
  <SECUREFILE>Y</SECUREFILE>
  <CHUNK>8192</CHUNK>
  <CACHE>N</CACHE>
  <LOGGING>Y</LOGGING>
  <INROW>Y</INROW>
  <COMPRESS>N</COMPRESS>
  </Column Name>
  </LOB>
  </Table_Name>
```

Where Parallel defines the number of threads, that Oracle DB Server will use to access a table or create an index.

We recommend creating CLOBs stored as SECUREFILE with Medium compression and Cache enabled. Please note that by default, medium compression is turned-off and must only be enabled if you have the Advanced compression license.

For instance, if a DBA decided to allocate table CI_ACCT in a tablespace MyTablespace, then they would have to change the storage.xml as follows:

```
<CI_ACCT>
<TABLESPACE>MyTablespace</TABLESPACE>
</CI_ACCT>
```

The oradbi process uses the storage.xml file to place the new database objects into defined tablespaces. A tablespace referenced in the storage.xml file must exist in the database.

The storage.xml file has to be adjusted before each upgrade and/or new installation as required to allocate the tables and indexes across those tablespaces.

Table name is included as a comment for each of the indexes for clarity.

For initial installs, information for each object should be reviewed by a DBA. For upgrades, only tablespace information for the objects added in the new release needs to be reviewed by a DBA.

Be careful while editing this file. Make sure that the tablespace names being used exist in the database. Do not change the basic format of this file.

Sequence

The base sequence name must be prefixed with the owner flag value of the product. For customer modification **CM** must prefix the sequence name. The sequence numbers should be named as below:

1. If the Sequence is used for a specific table, then use the following sequence name:

   [OF][C/M/T]NNN_SEQ

   - OF stands for Owner Flag. For example, for Framework its F1 and for CCB it is C1.
   - C/M/T stands for Control (Admin)/Master/Transaction Tables.
   - NNN is a three digit unique Identifier for a table on which the sequence is defined.

   For e.g: F1T220_SEQ

2. If more than one sequence is used for a specific table, then use the following Sequence Name:

   [OF][C/M/T]NNN_Column_Name_SEQ

   - C/M/T stands for Control (Admin)/Master/Transaction tables.
   - NNN is a three digit unique identifier for a table on which the sequence is defined.

   For Example: F1T220_BO_STATUS_CD_SEQ and F1T220_BUS_OBJ_CD_SEQ

3. If sequence is used for a generic requirement and not specific to a table, then use the following sequence name.

   [OF]Column_Name_SEQ

   For Example: F1FKVALID_SEQ

   - For a customer modification, CM must prefix the sequence name.

Trigger

The base trigger name must be prefixed with the owner flag value of the product.

When implementers add database objects, such as tables, triggers and sequences, the name of the objects should be prefixed by CM.

Column Data Type and Constraints

This section discusses the rules applied to column data type and constraints, and the attributes that are associated with these objects.

User Defined Code

User Defined Codes are defined as CHAR type. The length can vary by the business requirements but a minimum of eight characters is recommended. You will find columns defined in less than eight characters but with internationalization in mind, new columns should be defined as CHAR(10) or CHAR(12). Also note that when the code is referenced in the application the descriptions are shown to users in most cases.

System Assigned Identifier

System assigned random numbers are defined as CHAR type. The length of the column varies to meet the business requirements. Number type key columns are used when a sequential key assignment is allowed or number type is required to interface with external software. For example, Notification Upload Staging ID is a Number type because most EDI software uses a sequential key assignment mechanism. For sequential key assignment implementation, the DBMS sequence generator is used in conjunction with Number Type ID columns.

Date/Time/Timestamp

Date, Time and Timestamp columns are defined physically as DATE in Oracle. Non-null constraints are implemented only for the required columns.

Number

Numeric columns are implemented as NUMBER type in Oracle. The precision of the number should always be defined. The scale of the number might be defined. Non-null constraints are implemented for all number columns.

Fixed Length/Variable Length Character Columns

When a character column is a part of the primary key of a table define the column in CHAR type. For the non-key character columns, the length should be the defining factor. If the column length should be greater than 10, use VARCHAR2 type in Oracle.

Null Column Support

The product supports Nullable columns. This means that the application can write NULLs instead of a blank space or zero (for numeric columns) by using NULLABLE_SW on CI_MD_TBL_FLD. If REQUIRED_SW is set to 'N' and the NULLABLE_SW is set to 'Y', the application will write a NULL in that column. The artifact generator will create hibernate mapping files with appropriate parameters so that the framework hibernate mapping types will know if a given property supports a null value.

NULLABLE_SW is not new, but has previously been used for certain fields such as dates, and some string and number foreign-key columns. Because of this, there is the possibility that there is incorrect metadata for some columns, and that turning on this new feature could result in incorrect behavior when using that metadata. The upgrade script fixes the metadata to make sure that the existing tables will not be affected.

This new feature only supports tables maintained by Java but NOT a Java program converted from COBOL. Thus, enhancing any existing tables to use null columns must be done only after making sure that the tables are maintained by Java, and not Java converted COBOL programs.

XML Type Support

The product supports XML Type. XML Type provides following advantages

1. The ability to use XQuery for querying nodes in the XML document stored within a column defined as XMLType.
2. The option to use the XML engine, which is built into the Oracle Database, to create indexes using nodes within the XML document stored in the XMLType column.

Cache and Key Validation Flags

By default, the Cache Flag is set to NONE. For most of the admin tables the CACHE Flag should be 'Cached for Batch'. This specifies that the table is cached as L2 cache to reduce database trips.

By default the Key Validation Flag is set to ALL. For tables which have the user defined keys, the KEY_VALIDATION_FLG should be set as 'ALL'. This checks the existence of the key before inserting a new one.

Table Classification and Table Volume Flags

There are multiple types of tables in the application, namely Admin system tables, Admin non-system tables, master tables and transaction tables. The Table Classification flag (TBL_CLASSIFICATION_FLG) sets the appropriate value for this lookup field to give a better view of the table classification.

Table Volume flag (TBL_VOLUME_FLG) is a customer modifiable field which is initially populated by product, but can be overridden by implementation. The field gives an idea of the relative data volume (categorized as highVolume, lowVolume and mediumVolume) of the table to make informed decisions.

Default Value Setting

The rules for setting the database default values are as follows:

- When a predefined default value is not available, set the default value of Non-null CHAR or VARCHAR columns to blank except the primary key columns.
- When a predefined default value is not available, set the default value Non-null Number columns to 0 (zero) except the primary key columns.

- No database default values should be assigned to the Non Null Date, Time, and Timestamp columns.

Foreign Key Constraints

Referential integrity is enforced by the application. In the database do not define FK constraints. Indexes are created on most of Foreign Key columns to increase performance.

Standard Columns

This section discusses the rules applied to standard columns and the attributes that are associated with these objects.

Owner Flag

Owner Flag (OWNER_FLG) columns exist on the system tables that are shared by multiple products. Oracle Utilities Application Framework limits the data modification of the tables that have owner flag to the data owned by the product.

Version

The Version column is used to for optimistic concurrency control in the application code. Add the Version column to all tables that are maintained by a Row Maintenance program.

Chapter 4

Database Implementation Guidelines

This section outlines the general guidelines for database components, namely:

- Configuration Guidelines
- Oracle Database Implementation Guidelines

Configuration Guidelines

This section describes the general recommendations for configuring various database objects and includes a brief syntax overview. It covers the general aspects of the database objects and does not cover any specific implementation requirements.

Index

Index recommendations specify points that need to be considered when creating indexes on a table.

1. Indexes on a table should be created according to the functional requirements of the table and not in order to perform SQL tuning.
2. The foreign keys on a table should be indexes.

   Note: If the implementation creates a CM index on table-columns where the product already provides an index, then the CM index will be overridden by the base index.

Table Partitioning Recommendations

Oracle Utilities recommends using a minimum of 'n' partitions for selective database objects, where 'n' is number of RAC nodes.

Transparent Data Encryption Recommendations

Oracle Utilities supports Oracle Transparent Data Encryption (TDE). Oracle 11gR1 supports tablespace level encryption. The application supports tablespace level encryption for all application data. Make sure that the hardware resources are sufficiently sized for this as TDE uses additional hardware resources. The Oracle Advanced Security license is a prerequisite for using TDE.

Please consider the following when implementing TDE:

- Create a wallet folder to store the master key. By default, the wallet folder should be created under $ORACLE_BASE/admin/<sid>.

- The wallet containing the master key can be created using the following command:

```
alter system set encryption key authenticated by "keypasswd"
```

- The wallet can be closed or opened using the following commands:

```
alter system set wallet open identified by "keypasswd";
alter system set wallet close;
```

- Column level encryption can be achieved using the following commands:

```
create table <table_name>
(name varchar2(200) default ' ' not null,
bo_data_area CLOB encrypt using 'AES128',
bo_status_cd char(12)    encrypt using 'AES128')
lob (bo_data_area) store as securefile (cache compress)
tablespace <tablespace_name>;
```

- AES128 is the default encryption algorithm.
- Tablespace level encryption is also supported using the following command:

```
Create tablespace <tablespace_name> logging datafile '<datafile
location>' size <initial size> reuse autoextend on next <next
size>
maxsize unlimited extent management local uniform size
<uniform size> encryption using 'AES128' default
storage(encrypt);
```

- Indexed columns can only be encrypted using the NO SALT Option. Salt is a way to strengthen the security of encrypted data. It is a random string added to the data before it is encrypted, causing repetition of text in the clear to appear different when encrypted.

Data Compression Recommendations

Oracle Utilities supports Advanced Data Compression, available with Oracle 11gR1 onwards, to reduce the database storage footprint. Make sure that your resources are sufficiently sized for this as it uses additional system resources. Compression can be enabled at the Tablespace level or at the Table level.

Exadata Hardware

For Exadata hardware the compression recommendations are:

- For high volume tables, keep the current table partition uncompressed. All of the older partitions will be compressed based on QUERY HIGH compression.
- For high volume tables with CLOBs, always keep the CLOBs in securefiles with MEDIUM compression. Also keep the current table partition uncompressed. All of the older partitions will be compressed based on QUERY HIGH compression.
- Load data into the uncompressed table partitions using a conventional load and then, once data is loaded using a CTAS operation, load into a temporary heap table. Then truncate the original partition. Alter the original partition into HCC compressed and then partition exchange this with the temporary heap table.

- All multi column Indexes (primary as well as secondary) will be compressed using the default compression. HCC or OLTP compression is not applicable on the top of compressed Indexes.

Non- Exadata Hardware

For non-Exadata hardware the recommendations are the same as above, except that you cannot use HCC compression (it is only available in Exadata database machine). Instead of HCC you can use any other compression tool available to you for non-Exadata hardware.

CLOB Fields

All CLOB fields should be stored as SecureFiles and Medium compressed. This requires a separate license for Advanced Data Compression. As a part of the schema, we create the product-owned tables with compression turned OFF at the LOB level. If you have the license for Advanced Data Compression, you can enable compression by updating the storage.xml.

Compression Guidelines

- Admin and Metadata tables and their indexes will NOT be compressed.
- All Transactional Tables will be compressed.
  This includes ILM enabled MOs where applicable.
- Compression will be done at the tablespace level.
  - Different MOs will have different tablespaces.
  - Partitioned MOs will have one tablespace per partition.
  - Child tables will use reference partitioning with parent + children sharing the same tablespace. (parent and child will always be managed/archived together).
- All multicolumn indexes on transactional/ILM tables will be compressed.
  - Use 'compress advanced low'.
  - Local partitioned indexes will reside in the same tablespace as the table.
  - Each MO will have an index tablespace. All MO (Parent-Child Table) indexes will share this tablespace.
  - Do NOT specify standard index compression.
- Securefile medium compression in row for LOBs and CLOBs.

Examples:

Create a Tablespace with Advanced Rowstore Compress

```
CREATE BIGFILE TABLESPACE CM_XT012_P2017JANDATAFILE '+DATA' SIZE
50M AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS
ADVANCED;
```

Create Table with Subpartitions using Compressed Tablespaces & Securefiles Compression

```
CREATE TABLE CI_ADJ (
```

```
ADJ_ID        CHAR(12) NOT NULL ENABLE,

SA_ID         CHAR(10) DEFAULT ' ' NOT NULL ENABLE, ADJ_TYPE_CD
CHAR(8) DEFAULT ' ' NOT NULL ENABLE, ADJ_STATUS_FLG CHAR(2) DEFAULT
' ' NOT NULL ENABLE, CRE_DT DATE,

CAN_RSN_CD   CHAR(4) DEFAULT ' ' NOT NULL ENABLE,

ADJ_AMT     NUMBER(15,2) DEFAULT 0 NOT NULL ENABLE, XFER_ADJ_ID
CHAR(12) DEFAULT ' ' NOT NULL ENABLE, CURRENCY_CD  CHAR(3) DEFAULT
' ' NOT NULL ENABLE, COMMENTS    VARCHAR2(254) DEFAULT ' ' NOT NULL
ENABLE, VERSION     NUMBER(5,0) DEFAULT 1 NOT NULL ENABLE,
BEHALF_SA_ID CHAR(10) DEFAULT ' ' NOT NULL ENABLE, BASE_AMT
NUMBER(15,2) DEFAULT 0 NOT NULL ENABLE, GEN_REF_DT DATE,

APPR_REQ_ID CHAR(12) DEFAULT ' ' NOT NULL ENABLE,

ADJ_DATA_AREA CLOB, ILM_DT DATE,

ILM_ARCH_SW CHAR(1),)

ENABLE ROW MOVEMENT

PARTITION BY RANGE (ILM_DT)

SUBPARTITION BY RANGE (ADJ_ID) SUBPARTITION TEMPLATE (

SUBPARTITION S01 VALUES LESS THAN ( '124999999999' ), SUBPARTITION
S02 VALUES LESS THAN ( '249999999999' ), SUBPARTITION S03 VALUES
LESS THAN ( '374999999999' ), SUBPARTITION S04 VALUES LESS THAN (
'499999999999' ), SUBPARTITION S05 VALUES LESS THAN (
'624999999999' ), SUBPARTITION S06 VALUES LESS THAN (
'749999999999' ), SUBPARTITION S07 VALUES LESS THAN (
'874999999999' ), SUBPARTITION S08 VALUES LESS THAN ( MAXVALUE )

) (

PARTITION "P2017JAN" VALUES LESS THAN (TO_DATE('2017-02-01
00:00:01',

'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))

tablespace CM_XT012_P2017JAN,

PARTITION "P2017FEB" VALUES LESS THAN (TO_DATE('2017-03-01
00:00:01',

'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))

tablespace CM_XT012_P2017FEB,

PARTITION "P2017MAR" VALUES LESS THAN (TO_DATE('2017-04-01
00:00:01',

'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))

tablespace CM_XT012_P2017MAR,

PARTITION "P2017APR" VALUES LESS THAN (TO_DATE('2017-05-01
00:00:01',

'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))

tablespace CM_XT012_P2017APR,

PARTITION "P2017MAY" VALUES LESS THAN (TO_DATE('2017-06-01
00:00:01',

'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
```

```
tablespace CM_XT012_P2017MAY,

PARTITION "P2017JUN" VALUES LESS THAN (TO_DATE('2017-07-01
00:00:01',

'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))

tablespace CM_XT012_P2017JUN,

PARTITION "P2017JUL" VALUES LESS THAN (TO_DATE('2017-08-01
00:00:01',

'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))

tablespace CM_XT012_P2017JUL,

PARTITION "P2017AUG" VALUES LESS THAN (TO_DATE('2017-09-01
00:00:01',

'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))

tablespace CM_XT012_P2017AUG,

PARTITION "P2017SEP" VALUES LESS THAN (TO_DATE('2017-10-01
00:00:01',

'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))

tablespace CM_XT012_P2017SEP,

PARTITION "P2017OCT" VALUES LESS THAN (TO_DATE('2017-11-01
00:00:01',

'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))

tablespace CM_XT012_P2017OCT,

PARTITION "P2017NOV" VALUES LESS THAN (TO_DATE('2017-12-01
00:00:01',

'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))

tablespace CM_XT012_P2017NOV,

PARTITION "P2017DEC" VALUES LESS THAN (TO_DATE('2017-01-01
00:00:01',

'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))

tablespace CM_XT012_P2017DEC,

PARTITION "PMAX" VALUES LESS THAN (MAXVALUE)

tablespace CM_XT012_PMAX

);
```

Create a Compressed Local Index

```
CREATE UNIQUE INDEX XT012S3 ON CI_ADJ ( ILM_DT, ILM_ARCH_SW, ADJ_ID

) TABLESPACE CM_XT012_IND COMPRESS ADVANCED LOW;
```

Create a Compressed Global Partitioned Index

```
CREATE UNIQUE INDEX XT012S2 ON CI_ADJ ( XFER_ADJ_ID, ADJ_ID )
TABLESPACE CM_XT012_IND

GLOBAL PARTITION BY HASH  (XFER_ADJ_ID, ADJ_ID ) (
```

```
PARTITION PART1 TABLESPACE CM_XT012_IND, PARTITION PART2 TABLESPACE
CM_XT012_IND, PARTITION PART3 TABLESPACE CM_XT012_IND, PARTITION
PART4 TABLESPACE CM_XT012_IND, PARTITION PART5 TABLESPACE
CM_XT012_IND, PARTITION PART6 TABLESPACE CM_XT012_IND, PARTITION
PART7 TABLESPACE CM_XT012_IND, PARTITION PART8 TABLESPACE
CM_XT012_IND

)

COMPRESS ADVANCED LOW;

Do NOT specify standard index compression.

CREATE INDEX XT012S1 ON CI_ADJ ( SA_ID, ADJ_TYPE_CD ) TABLESPACE
CM_XT012_IND LOCAL COMPRESS 1 COMPRESS ADVANCED LOW;
```

Database Vault Recommendations

The product supports Database Vault. All non-application User IDs can be prevented from using DDL or DML statements against the application schema. So SYS and SYSTEM cannot issue DDL or DML statements against CISADM schema.

The application-specific administration account can issue DDL statements but should not be able to perform any DML or DCL statements.

Application user must be given DML only permissions.

Database Vault can be used to control access during patch process and Install/Upgrade process.

Oracle Fuzzy Search Support

The product supports Oracle Fuzzy searches. To use this feature, Oracle Text must be installed. After Oracle Text is installed, an index must be created on the table where the fuzzy search needs to be performed from the application. This is only an Oracle database option and is not supported by other databases. Additionally, not all languages are supported. Refer to the Oracle database documentation for more information about fuzzy searching.

A typical syntax for implementation of fuzzy searching is as below. For the most updated syntax, please refer to Oracle Fuzzy documentation.

```
GRANT CTXAPP TO <Application schema owner e.g CISADM>;

GRANT EXECUTE ON CTX_DDL TO <Application schema owner e.g CISADM>;

create index <Application schema owner e.g CISADM>.<Index_Name> on
Application schema owner e.g CISADM>.<Table_Name> (<column_name>)
indextype is ctxsys.context parameters ('sync (on commit)');
begin
ctx_ddl.sync_index('Application schema owner e.g
CISADM>.<Index_Name>');
end
/
```

Information Lifecycle Management (ILM) and Data Archiving Support

The product supports Data Archiving based on Information Lifecycle Management (ILM). If Information Lifecycle Management is part of your implementation, refer to the Information Lifecycle Management and Data Archiving in CCB chapter for instructions on partitioning objects when using ILM.

Storage Recommendations

This section specifies recommended options for storing the database objects.

SecureFile for Storing LOBs

Beginning with Oracle 11g, tables having fields with data type of CLOB or BLOBS should have the LOB Columns stored as SecureFiles.

- The storage options with SecureFiles for Heap Tables should be ENABLE STORAGE IN ROW, CACHE and COMPRESS.
- For the IOT Table the PCTTHRESHOLD 50 OVERFLOW clause should be specified and the storage options with SecureFiles should be ENABLE STORAGE IN ROW, CACHE and COMPRESS.
- The PCTTHRESHOLD should be specified as a percentage of the block size. This value defines the maximum size of the portion of the row that is stored in the Index block when an overflow segment is used.
- The CHUNK option for storage, which is the data size used when accessing or modifying LOB values, can be set to higher than one database block size if big LOBs are used in the IO Operation.
- For SecureFiles, make sure that the initialization parameter db_securefile is set to ALWAYS.
- The Tablespace where you are creating the SecureFiles should be enabled with Automatic Segment Space Management (ASSM). In Oracle Database 11g, the default mode of Tablespace creation is ASSM so it may already be set for the Tablespace. If it's not, then you have to create the SecureFiles on a new ASSM Tablespace.

  Note: To enable compression on SecureFiles, you must have an Oracle Advanced Compression license in addition to Oracle Database Enterprise Edition. This feature is not available for the standard edition of the Oracle database.

  If you are using Oracle Database Enterprise Edition, please verify that the "COMPRESS" flag is turned on by setting it to "Y" in Storage.xml.

Refer to the Database Syntax section for more information on SecureFiles.

Database Configuration Recommendations

This section specifies the recommended methods for configuring the database with a focus on specific functional area.

Large Redo Log File Sizes

The Redo Log files are written by the Log Writer Background process. These log files are written in a serial manner. Once a log File is full, a log switch occurs and the next log file starts getting populated.

It is recommended that the size of the Redo log files should be sufficiently high so that you do not see frequent Log Switches in the alert logs of the database. Frequent Log Switches impact the IO performance and can be avoided by having a larger Redo log file size.

Frequent Log Switches impacts the IO performance and can be avoided by having a bigger Redo log File Size.

Database Syntax

SecureFile

```
CREATE TABLE <Table_Name>
     ( COLUMN1 ….,
       COLUMN2 (CLOB)
     )
LOB(COLUMN2) STORE AS SECUREFILE (CACHE COMPRESS);

CREATE TABLE <Table_Name>
     ( COLUMN1 ….,
       COLUMN2 (CLOB)
       CONTRAINT <> PRIMARY KEY(…)
     )
ORGANIZATION INDEX PCTTHRESHOLD 50 OVERFLOW
LOB(COLUMN2) STORE AS SECUREFILE (ENABLE STORAGE IN ROW CHUNK CACHE
COMPRESS);
```

Database Initialization Parameters

The recommended initialization parameters are given below. These parameters are a starting point for database tuning. An optimal value for a production environment may differ from one customer deployment to another.

db_block_size=8192

log_checkpoint_interval=0

db_file_multiblock_read_count=8

transactions=3000

open_cursors=800

db_writer_processes=10

db_files=1024

dbwr_io_slaves=10 (Only if Asynchronous IO is not Supported)

sessions=4500

memory_target=0

memory_max_target=0

processes=3000

dml_locks=48600

_b_tree_bitmap_plans=FALSE

Oracle Database Implementation Guidelines

This section provides specific guidelines for implementing the Oracle database.

Oracle Partitioning

If you use a base index for the partitioning key, rename the index to CM**.

If you use the primary key index of the table as the partitioning key:

- Make the index non-unique.
- Primary constraints should still exist.

The upgrade on the partitioned table works best if the partitioning key is not unique. This allows the upgrade tool to drop the PK constraints if the primary key columns are modified and recreate the PK constraints without dropping the index.

Database Statistic

During an install process, new database objects may be added to the target database. Before starting to use the database, generate the complete statistics for these new objects by using the DBMS_STATS package. You should gather statistics periodically for objects where the statistics become stale over time because of changing data volumes or changes in column values. New statistics should be gathered after a schema object's data or structure are modified in ways that make the previous statistics inaccurate. For example, after loading a significant number of rows into a table, collect new statistics on the number of rows. After updating data in a table, you do not need to collect new statistics on the number of rows, but you might need new statistics on the average row length.

A sample syntax that can be used is as following:

```
BEGIN
SYS.DBMS_STATS.GATHER_SCHEMA_STATS (
OwnName => 'CISADM'
,Degree => 16
,Cascade => TRUE
,Method_opt => 'FOR ALL COLUMNS SIZE AUTO'
, Granularity => 'ALL' );
END;
/
```

Materialized View

Oracle Enterprise Edition supports query rewrite Materialized view. If you use Oracle Enterprise Edition, you can create following Materialized Views to improve performance of the Monitor batch jobs.

Prerequisites

Make sure to set up the following:

1. Set parameter QUERY_REWRITE_ENABLED=TRUE at database level.

   ```
   ALTER SYSTEM SET QUERY_REWRITE_ENABLED=TRUE; OR
   ALTER SYSTEM SET QUERY_REWRITE_ENABLED=TRUE SCOPE=BOTH;
   ```

2. To create a materialized view in another user's schema you must have the **CREATE ANY MATERIALIZED VIEW** system privilege. The owner of the materialized view must have the CREATE TABLE system privilege. The owner must also have access to any master tables of the materialized view that the schema owner does not own (for example: if the master tables are on a remote database) and to any materialized view logs defined on those master tables, either through a **SELECT** object privilege on each of the tables or through the **SELECT ANY TABLE** system privilege.
3. To create a refresh-on-commit materialized view (**ON COMMIT REFRESH** clause), in addition to the preceding privileges, you must have the **ON COMMIT REFRESH** object privilege on any master tables that you do not own or you must have the **ON COMMIT REFRESH** system privilege.

   To create the materialized view with query rewrite enabled, in addition to the preceding privileges: If the schema owner does not own the master tables, then the schema owner must have the **GLOBAL QUERY REWRITE** privilege or the **QUERY REWRITE** object privilege on each table outside the schema.

To debug materialized views, refer the below URLs:

- Oracle 11g - https://docs.oracle.com/cd/B28359_01/server.111/b28313/qrbasic.htm
- Oracle 12c - https://docs.oracle.com/database/121/DWHSG/qrbasic.htm#DWHSG01813
- Troubleshoot Materialized View - http://docs.oracle.com/database/121/ARPLS/d_mview.htm#ARPLS67193

```
CREATE MATERIALIZED VIEW F1_BO_LIFECYCLE_STATUS_MVW
(
BUS_OBJ_CD,
LIFE_CYCLE_BO_CD,
BO_STATUS_CD,
BATCH_CD
)
BUILD IMMEDIATE REFRESH ON COMMIT ENABLE QUERY REWRITE AS
SELECT
BO2.BUS_OBJ_CD,BO.LIFE_CYCLE_BO_CD,BOSA.BO_STATUS_CD,LCBOS.BATCH_CD as
LC_BATCH_CD
FROM
F1_BUS_OBJ BO2,
```

```
F1_BUS_OBJ BO,
F1_BUS_OBJ_STATUS LCBOS,
F1_BUS_OBJ_STATUS_ALG BOSA
WHERE
BO2.LIFE_CYCLE_BO_CD =BO.LIFE_CYCLE_BO_CD AND
BO.BUS_OBJ_CD = BOSA.BUS_OBJ_CD AND
BOSA.BO_STATUS_SEVT_FLG = 'F1AT' AND
LCBOS.BUS_OBJ_CD = BO.LIFE_CYCLE_BO_CD AND
LCBOS.BO_STATUS_CD = BOSA.BO_STATUS_CD
/
```

create synonym SPLUSR.F1_BO_LIFECYCLE_STATUS_MVW for SPLADM.F1_BO_LIFECYCLE_STATUS_MVW;

grant select on F1_BO_LIFECYCLE_STATUS_MVW to FW_DEV;

grant select on F1_BO_LIFECYCLE_STATUS_MVW to SPL_USER;

grant select on F1_BO_LIFECYCLE_STATUS_MVW to SPL_READ;

For more information, refer to the following documents:

- Basic Query Rewrite (Oracle 11g) - https://docs.oracle.com/cd/B28359_01/server.111/b28313/qrbasic.htm
- Basic Query Rewrite for Materialized Views (Oracle 12c) - https://docs.oracle.com/database/121/DWHSG/qrbasic.htm#DWHSG01813
- Troubleshooting Materialized Views - http://docs.oracle.com/database/121/ARPLS/d_mview.htm#ARPLS67193
- Debugging materialized Views - http://docs.oracle.com/cd/B28359_01/server.111/b28313/qradv.htm

Known Issues

The following are some of the known issues at the time of release. For more information, refer to these articles on My Oracle Support:

- Query Did Not Rewrite For A User Other Than The Owner Of the Materialized View (Doc ID 1594725.1) - A patch is available for bug report 14772096 for some platforms.
- Query rewrite not working as expected with SELECT DISTINCT (Doc ID 7661113.8) for Oracle version – 11.2.0.1 and 11.1.0.7 Fixed in version - 12.1.0.1 (Base Release), 11.2.0.2 (Server Patch Set)

Chapter 5

Conversion Tools

This section describes the following database conversion tools:

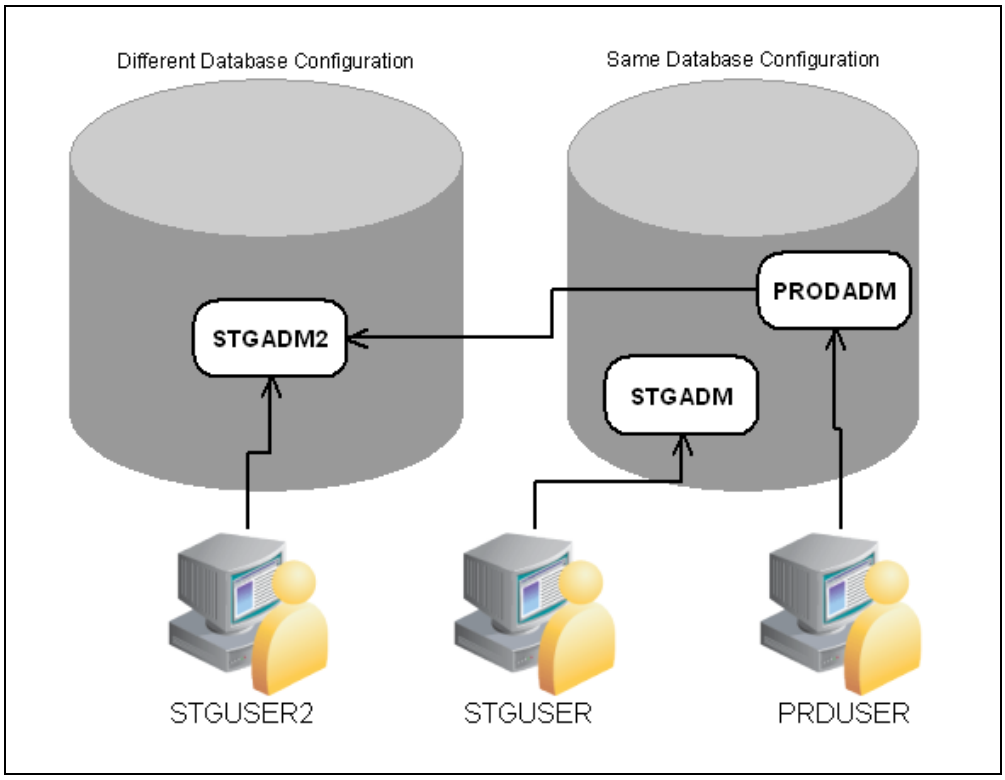
- Database Configuration
- Script Installation
- Preparing the Production Database
- Preparing the Staging Database

Note that all database related single fixes and service packs need to be applied against the production schema. Staging schema should not be updated with database single fixes or service packs. Staging schema need to be rebuilt for any fixes that contain DDL to create new database objects in production schema.

Database Configuration

The Conversion Tool Kit requires at least two sets of schema. One is to hold the staging data that the conversion tool gets the data from and performs validations. We call this schema the staging database. The target schema, which is referred to as the production database, is where the conversion tool inserts the validated data. Both the production database and the staging databases can reside in a single Oracle database or in different databases that are connected via a database link. Only the single database configuration is supported.

The following schematic diagram shows a sample configuration of both the production and staging environments in which the Conversion Tool Kit operates. The production and staging databases must be the same release level.



All the tables and views for the application are defined in the production database. The staging database has the same set of tables and views as the production database, except the tables that are grouped as part of the business configuration (control tables). Details on the differences of the tables of the two databases and of the conversion tool functionality are found in the Conversion Tool document.

Script Installation

The Conversion Setup Utility, ConvSetup.exe, is provided in this release of Oracle Utilities Customer Care and Billing to set up conversion schemas.

Install the Oracle Client 12.1.0.2(+) on Windows desktop and configure SQLNet to connect to the target database.

The Conversion folder contains the conversion setup utility: ConvSetup.exe and Conversion.bat.

This section of the document describes how to create the databases for the conversion tool kit.

Preparing the Production Database

If the production database does not exist create the database under the production schema owner (CISADM).

> If the production database is upgraded from the previous version of the application make sure all public synonyms that are created on the application tables are deleted. Instead, each application user should have private synonyms created on the application tables in order for the conversion tool configuration to work.

Preparing the Staging Database

Once you have created a staging owner (STGADM), application user (STGUSER) and read access user (STGREAD), install the initial database option in the staging schema. The rest of the steps are listed below.

Run ConvSetup.exe from under the Conversion folder. The script prompts you for the following values:

- Database Platform: Oracle (O)
- Database connection information
- Database Name
- System Password
- Production Schema Name
- Staging Schema Name
- Read-Write user for Staging Schema.

ConvSetup.exe performs following tasks:

- Creates cx* views on the master/transaction tables in the production database.
- Grants the privileges on the master/transaction tables in the production database to the staging owner.
- Drops control tables and creates views on production control tables in the staging database.
- Grants privileges on the control tables to the staging owner.
- Grants privileges on the cx* views to the staging application user.
- Creates generated key tables.
- Creates generated table primary key and secondary indexes.

In addition to above tasks ConvSetup.exe also generates the following SQL scripts:

- create_cxviews.sql
- create_ctlviews.sql

- createck_tbls.sql
- create_grants.sql
- createck_pkix.sql
- createck_secix.sql

By default the conversion.bat updates all changes to the staging schema. If you want to generate only the above sql scripts and not apply changes to staging schema then update conversion.bat by removing "-u". The sql scripts can be applied to the staging schema later. The sqls scripts need to be executed in the same order as described above using SQL*PLus.

Once the staging schema has been set up, generate the security for the staging user using:

oragensec.exe -d stgadm,schemapassword,database_name -r
stg_read,stg_user -u stguser

Chapter 6

Information Lifecycle Management and Data Archiving in CCB

Oracle Utilities Customer Care and Billing provides support for Information Lifecycle Management (ILM) and Data Archiving.

ILM is process to address data management issues, with a combination of processes, policies, software and hardware so that the appropriate technology can be used for each phase of the lifecycle of the data. The lifecycle of data typically refers to the fact that the most recent data is active in the system and as time passes the data is accessed less frequently or not at all. The costs of storing data that are accessed infrequently can be reduced by moving the data to lower cost mass storage media. Typically this involves a trade-off between cost and increased access times. Based on business needs, data may eventually be archived and purged from the database and kept offline ready to be restored if required.

This chapter includes:

ILM Implementation Overview

The implementation of ILM for products based on Oracle Utilities Application Framework includes a combination of application and database configuration and requires Oracle Partitioning.

An underlying design principle of the Oracle Utilities Application Framework ILM implementation is the concept that the age of the data may not the only criterion used to determine when a record is able to be archived. There may be business rules that dictate that some records are still current and must not be archived yet.

ILM enabled objects have a combination of an ILM date and an ILM Archive Switch. The ILM date is used in conjunction with partitioning to group data by age. The ILM Archive Switch is set by a background process when the record meets the business rules specific to that Maintenance Object if the record is eligible to be archived. The ILM Archive Switch gives Database Administrators an easy method to check when all records in a partition meet the business criteria that make the partition eligible to be archived. If the ILM Archive Switch is set for all records, then the DBA can take the steps required to archive the partition.

Moving data between storage tiers takes advantage of the partitioning by ILM Date but does not require that the ILM Archive Switch is set. Oracle recommends using the Oracle Database ILM Assistant to assist with this process.

ILM Implementation Components

The ILM based solution contains a number of components.

- ILM Specific Table Columns - For any Maintenance Object (MO) that has been configured to support ILM, the primary table of the MO includes two columns: ILM Date and ILM Archive Switch.
    - ILM_DT - This date column is defaulted to an appropriate date (typically the system date) when a new record is inserted, the MO is partitioned on the ILM_DT, so it should only be updated in exceptional circumstances as this would cause the record to be deleted from its current partition and inserted into a different partition, which is a relatively expensive operation.
    - ILM_ARCHIVE_SW - This field is set to N (Not yet eligible for archiving) when a new record is inserted. Subsequent reviews of "old" records may assess the data and change the value to "Y" based on business rules indicating that the record is eligible to be archived.
- Database Referential Integrity Constraints - These are required for reference partitioning of Child tables of ILM enabled MOs
- Partitioning - Partitioning is mandatory for ILM implementation. It is used to separate the data by ILM date so that data of a similar age is kept together.
- One Tablespace per Partition - The ILM implementation requires that each MO partition resides in a dedicated tablespace so that they can be easily managed.
- Naming Convention - This section covers the recommended naming convention to be used for partitions/subpartitions and tablespaces.

ILM Database Administrator's Tasks

For a database administrator, there are two key phases involved with managing your data using ILM.

- Preparation Phase - This phase covers the database level configuration that needs to be done before the ILM solution runs in a production environment.
- Business FlagOn-going Maintenance Phase - This phase covers the ongoing maintenance tasks such as add partition, archive and restore partitions.

Preparation Phase

Note: In order to successfully implement ILM as described here, the following database version and patch are pre-requisites: version 12.1.0.2.0 and Patch 15996848.

The steps needed to enable ILM functionality differ depending on whether ILM is enabled as part of the initial implementation of the product or enabled ILM on an existing implementation where data already exists in the respective tables.

- **Initial Install** – For an initial installation, the section Module Specific ILM Implementation Details outlines the additional steps to be performed on base delivered ILM Enabled Tables to conform to ILM requirements. In addition, Appendix A: Sample SQL for Enabling ILM in CCB (Initial Install) provides sample reference DDLs using two maintenance objects as examples.
- **Transform NON-ILM implementation to ILM Enabled Implementation**: The following steps provide a high level overview of steps that must be performed to implement ILM on enabled MOs for an existing implementation. Please refer to Appendix B: Sample SQL For Enabling ILM in CCB (Existing Installation) for detailed information using To Do Entry as an example.

1. Rename the existing tables (Parent table followed by child table(s)), and primary key index associated with ILM enabled MOs by renaming the tables.
2. Save the DDLs for the secondary indexes as you will need to recreate them later.
3. Drop secondary indexes on the renamed tables.
4. Create Partitioned table with no secondary indexes for ILM enabled MOs using a CTAS operation (Create Table as Select), which will also load the data into the partitioned table structure.

   Functional Note: ILM enabled MOs should have the ILM date (ILM_DT) populated when data is moved into the new partitioned table. Please refer to the Module Specific ILM Implementation Details section below for initial load details on which date column to use as the basis for populating the ILM date. Often it is based on Create Date (CRE_DTTM). ILM_ARCH_SW should initially be set to 'N'.
5. Enable logging option.
6. Create Primary Key index.
7. Create Primary Key Constraint of parent table.
8. Create secondary indexes for the newly-created partitioned tables. This includes creating an index used specifically to benefit the ILM Crawler batch. The recommendation for this index name is to prefix it with "ILM".

Note: This can be created specifying parallel index create; remember to turn off parallelism after the index is created.

9. Follow similar operation for all child tables for this MO, such as rename child table, and primary key index, generate DDL for secondary index, drop secondary index etc. Sample DDL for child tables their partitioning and indexes can be found in Appendix B: Sample SQL For Enabling ILM in CCB (Existing Installation). Please note that child table should be partitioned using reference partitioning of the parent table's partitioning key.
10. Drop the original, renamed tables after verifying the newly created partitioned tables.

On-going Maintenance Phase

The following steps provide a high level overview of what needs to be done for on-going maintenance for ILM on enabled MOs.

Please refer to the Appendix C: Sample SQL for Periodic Maintenance for detailed information using two maintenance objects as examples.

1. Add the partition:
    a. Create Tablespace to be used for the new parent table partition.
    b. Since, we define MAXVALUE Partition; new partition can only be created using "SPLIT" operation. Identify and use next HIGH_VALUE Partition for the split operation.
    c. All the child table(s) partition(s)\LOB(s) must be altered to use the same tablespace as that of the parent table's partition.
    d. Enable advanced compression on all child table(s).
    e. Copy partition level statistics from the previous partition
2. Archive the partition:
    a. Make the tablespace that will be archived READ ONLY.
    b. Check that no records have ILM_ARCH_SW = 'N'.
        - If record count is zero, proceed with further steps.
        - If record count is not zero, then change the tablespace back to READ WRITE MODE as Archive is not Feasible at the time.
    c. Create an archive tablespace for the partition that needs to be archived.
    d. Create staging tables using the new archive tablespace. Load data for all child tables first.
    e. Create staging table using the new archive tablespace and load data for the parent table.
    f. Export tablespace using TRANSPORT_TABLESPACES method.

    Make Sure Tablespace datafile required for further import is preserved.
    g. Drop the partition, partition the tablespace and archive the tablespace (as it is already exported).

3. Restore the partition:
    a. Create a new tablespace to restore the partition.
    b. Add partition using split operation on next greater high value partition.

       If the table contains LOBS, there will an additional statement in split partition DDL indicating tablespace where the LOBs will be stored.
    c. Enable advanced compression on all child table(s).
    d. Import Tablespace using TRANSPORT_TABLESPACES method.
    e. Load data into the parent table first from the staging table
    f. Load data into the child table from the staging table
    g. Drop the archive tablespace after import and data loading is successful.
4. Move Data between different storage tiers:

The ILM facilities can be used within the database to implement storage savings, as follows:

- Use ILM Assistant to define the data groups to be used for the individual objects. Assign those data groups to partitions and storage devices to implement the storage savings.
- Use ILM assistant to generate the necessary commands to implement the data changes manually or use Automatic Storage Management (ASM) to automate the data storage policies.
- Optionally, use Automatic Data Optimization to provide further optimizations.

For more information about ILM Assistant and ILM refer to the following:

- ILM Assistant Users Guide available at:

  http://download.oracle.com/otn/other/ilm/ilma-users-guide.html
- Oracle Database VLDB and Partitioning Guide (11.2) available at:

  http://docs.oracle.com/cd/E11882_01/server.112/e25523/part_lifecycle.htm#CACECAFB
- Oracle Database VLDB and Partitioning Guide (12.1) available at:

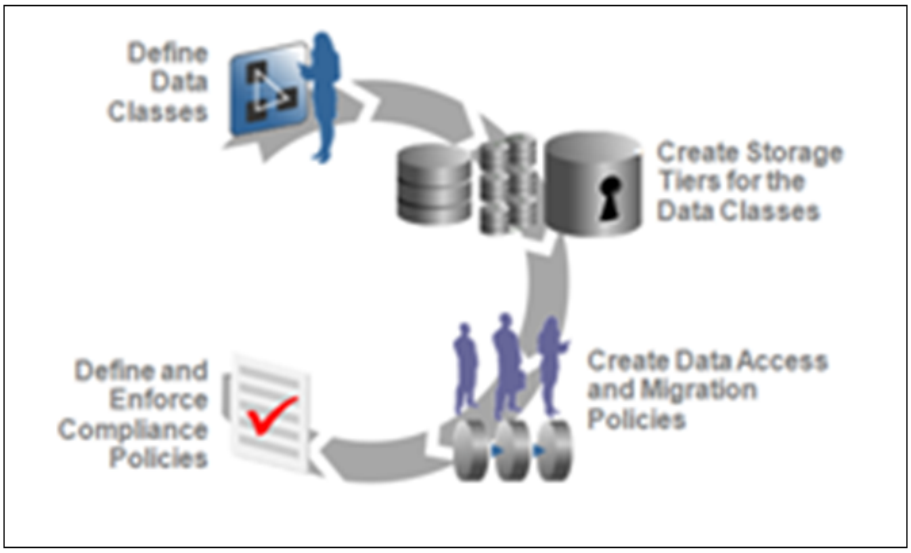
  https://docs.oracle.com/database/121/VLDBG/title.htm

ILM Assistant

The ILM Assistant in the current 11g database implementation can provide the following

- Setup ILM Lifecycle definition - Define different lifecycle definitions for different MOs and say that after what period of time the data is ready to be moved to a slower disk.
- Setup ILM Lifecycle tables - Define the tables you want to manage and assign it to a Lifecycle definition defined above. You can setup policies for when data is moved it can be automatically compressed to desired degree.
- Lifecycle Management - The Lifecycle Management tab alerts the system admin when partitions are eligible for archiving.

ILM Assistant can then be used with the ILM to make sure the records that have ILM_ARCH_SW = 'Y' can be moved to slower and slower disks and possibly get purged.



Note: For further guidelines on ILM Assistant refer to Implementing Information Lifecycle Management Using the ILM Assistant available at: http://www.oracle.com/webfolder/technetwork/tutorials/obe/db/11g/r2/prod/storage/ilm/ilm.htm?cid=4196&ssid=115606280996764

Naming Convention

The naming convention for tablespace, partitions & subpartition is standardized as follows

- Each name consists of some or all of the following parts
- The parts of the name are organized hierarchically
- Each part of the Name is separated with an underscore.
- The maximum name length must not exceed 30 Characters.
- For an MO, the parent table and child table share the same tablespace for the corresponding partition (or sub partition as appropriate).
- Square brackets [ ] indicate that this part of the name should be omitted if not required.

OWNERFLAG_TABLEIDENTIFIER_PARTITIONNAME[_SUBPARTITIONNAME][_ARCHIVEFLAG][_COMPRESSFLAG]

For details on the convention, please refer to the table below:

Convention	Description
OWNERFLAG	Owner flag for the relevant application for example “C1” for CCB

Convention	Description
TABLE IDENTIFIER	The Index Name of the Primary Key index without the "P0" suffix. For example, if the PK index name is XT039P0, the table identifier would be "XT039".
PARTITION NAME	The Partition name should be prefixed with a P followed by a name which conforms to one of the following standards: • 4 digit year and 3 letter month abbreviation PYYYYMON corresponding to the ILM date e.g. P2017JAN • PMAX if it is the Max Value partition
SUBPARTITION NAME	If subpartitions are used, name should be prefixed with S followed by a name of not more than 5 characters which conforms to the following requirements: • SMAX if this is the Max Value sub partition • If the sub partition holds data for a sub retention period use a number equal to that period e.g S91 if the sub retention period < 91 days. • For a range based SubPartition on Primary Key, use an integral number increasing by +1. For example, if there are 8 sub partitions use S01 through S08
ARCHIVEFLAG	This flag is used as a suffix to the table and tablespace name for the staging tables created for the archiving operation. • ARC

Convention	Description
COMPRESS FLAG	This flag is used as a suffix to the tablespace name for the staging tables created when compressing a partition. • C For compression related tasks, this is used as suffix to the tablespace name. • Partition Tablespace Name: It is formed by OWNERFLAG_TABLEIDENTIFIER_PARTITIONNAME. For example: CM_D1T304_PMAX CM_D1T304_P2017JAN • SubPartition Tablespace Name: It is formed by OWNERFLAG_TABLEIDENTIFIER_PARTITIONNAME_SUBPARTITIONNAME. For example: CM_D1T304_PMAX_SMAX CM_D1T304_P2017JAN_SMAX CM_D1T304_PMAX_S001 CM_D1T304_P2017JAN_S181 • Archive Staging Table And Its Tablespace Name (When archiving partition): It is formed by OWNERFLAG_TABLEIDENTIFIER_PARTITIONNAME_ ARCHIVEFLAG. For example: CM_D1T304_P2017JAN_ARC • Archive Staging Table And Its Tablespace Name (When archiving subpartition): It is formed by OWNERFLAG_TABLEIDENTIFIER_PARTITIONNAME_SUBPARTITIONNAME_ ARCHIVEFLAG. For example: CM_D1T304_P2017JAN_S181_ARC • Compressed Tablespace name (When compressing partition): For example: CM_D1T304_P2017JAN_ C

Module Specific ILM Implementation Details

This section outlines each maintenance object that has been configured to support ILM. The parent table is noted. Other tables are child tables of the parent unless otherwise noted. In each case, the partitioning strategy is indicated.

All indexes are listed with a recommendation whether the index should be global or local and whether the index should be partitioned. In addition to the base delivered indexes, each parent table includes a recommended ILM specific local index to build with the ILM_DT, ILM_ARCH_SW and the primary key of the table. The recommended column that should be used to populate the ILM_DT is also shown. Refer to Appendix B: Sample SQL For Enabling ILM in CCB (Existing Installation) for sample DDL(s).

To Do Entry

This table describes the To Do Entry maintenance object.

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
CI_TD_ENTRY (Parent)	RANGE (ILM_DT, TD_ENTRY_ID)				RANGE (TD_ENTRY_ID)	CI_TD_ENTRY. CRE_DTTM
		XT039P0	TD_ENTRY_ID	Global Partitioned		
		XT039S2	ASSIGNED_TO, TD_ENTRY_ID	Global		
		XT039S3	ENTRY_STATUS_FLG, ASSIGNED_TO	Global		
		XT039S4	ROLE_ID, TD_TYPE_CD, ENTRY_STATUS_FLG, TD_PRIORITY_FLG, ASSIGNED_TO, CRE_DTTM	Global		
		XT039S5	BATCH_CD, BATCH_NBR, ENTRY_STATUS_FLG	Global		
		XT039S6	TD_ENTRY_ID, ASSIGNED_TO, ENTRY_STATUS_FLG	Global		
		XT039S7	COMPLETE_USER_ID, COMPLETE_DTTM, TD_ENTRY_ID	Global		
		CM_ILM_XT039S8	ILM_DT, ILM_ARCH_SW, TD_ENTRY_ID	Local Partitioned		
CI_TD_ENTRY_CHA	Reference Partitioning	XT701P0	TD_ENTRY_ID, CHAR_TYPE_CD, SEQ_NUM	Global Partitioned		
		XT701S1	SRCH_CHAR_VAL, CHAR_TYPE_CD, TD_ENTRY_ID	Global		

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
		XT701S2	CHAR_VAL_FK1	Global		
CI_TD_DRLKEY	Reference Partitioning	XT037P0	TD_ENTRY_ID, SEQ_NUM	Global Partitioned		
		XT037S1	KEY_VALUE, TD_ENTRY_ID	Global		
CI_TD_LOG	Reference Partitioning	XT721P0	TD_ENTRY_ID, SEQ_NUM	Global Partitioned		
		XT721S1	LOG_DTTM,USER_ID, LOG_TYPE_FLG, TD_ENTRY_ID	Global		
CI_TD_MSG_ PARM(Child table of CI_TD_LOG)	Reference Partitioning	XT040P0	TD_ENTRY_ID, SEQ_NUM	Global Partitioned		
CI_TD_SRTKEY	Reference Partitioning	XT041P0	TD_ENTRY_ID, SEQ_NUM	Global Partitioned		
		XT041S1	KEY_VALUE, TD_ENTRY_ID	Global		

Sync Request (Outbound)

This table describes the Sync Request (Outbound) maintenance object.

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
F1_SYNC_REQ (Parent)	RANGE (ILM_DT, F1_SYNC_REQ_ ID)				RANGE (F1_SYNC_REQ_ ID)	F1_SYNC_REQ.C RE_DTTM
		F1T014P0	F1_SYNC_REQ_ID	Global Partitioned		
		F1T014S1	BO_STATUS_CD, BUS_OBJ_CD, F1_SYNC_REQ_ID	Global		
		F1T014S2	BO_STATUS_ REASON_CD	Global		

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
		F1T014S3	MAINT_OBJ_CD, PK_VALUE1, PK_VALUE2, F1_SYNC_REQ_ID	Global		
		CM_ILM_F1T014S4	ILM_DT, ILM_ARC_SW, F1_SYNC_REQ_ID	Local Partitioned		
F1_SYNC_REQ_CHAR	Reference Partitioning	F1T017P0	F1_SYNC_REQ_ID, CHAR_TYPE_CD, SEQ_NUM	Global Partitioned		
		F1T017S1	SRCH_CHAR_VAL	Global		
F1_SYNC_REQ_EXTRACT	Reference Partitioning	F1T019P0	F1_SYNC_REQ_ID, SEQ_NUM	Global Partitioned		
F1_SYNC_REQ_LOG	Reference Partitioning	F1T015P0	F1_SYNC_REQ_ID, SEQNO	Global Partitioned		
		F1T015S1	CHAR_TYPE_CD, CHAR_VAL_FK1	Global		
		F1T015S2	CHAR_TYPE_CD, CHAR_VAL	Global		
		F1T015S3	BO_STATUS_REASON_CD	Global		
F1_SYNC_REQ_LOG_PARM (Child Table of F1_SYNC_REQ_LOG_PARM)	Reference Partitioning	F1T016P0	F1_SYNC_REQ_ID, SEQNO, PARM_SEQ	Global Partitioned		

Note: It is recommended that data retention policies and rules for this object match the policies and rules implemented for the Inbound Sync Request on the target system to avoid data inconsistencies when auditing.

Inbound Sync Request

This table describes the Inbound Sync Request maintenance object.

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
F1_SYNC_REQ_IN (Parent)	RANGE(ILM_DT, F1_SYNC_REQ_IN_ID)				RANGE (F1_SYNC_REQ_IN_ID)	F1_SYNC_REQ_IN.CRE_DTTM
		F1T191P0	F1_SYNC_REQ_IN_ID	Global Partitioned		
		F1T191S1	BO_STATUS_CD, BUS_OBJ_CD, F1_SYNC_REQ_IN_ID	Global		
		F1T191S2	MAINT_OBJ_CD, EXT_PK_VALUE 1, NT_XID_CD, PK_VALUE1	Global		
		F1T191S3	EXT_REFERENCE_ID	Global		
		CM_ILM_F1T191 S3	ILM_DT, ILM_ARCH_SW, F1_SYNC_REQ_IN_ID	Local Partitioned		
F1_SYNC_REQ_IN_CHAR	Reference Partitioning	F1T193P0	F1_SYNC_REQ_IN_ID, CHAR_TYPE_CD, SEQ_NUM	Global Partitioned		
		F1T193S1	SRCH_CHAR_VAL	Global		
F1_SYNC_REQ_IN_EXCP	Reference Partitioning	F1T197P0	F1_SYNC_REQ_IN_ID, SEQNO	Global Partitioned		
F1_SYNC_REQ_IN_EXCP_PARM (Child Table of F1_SYNC_REQ_IN_EXCP)	Reference Partitioning	F1T198P0	F1_SYNC_REQ_IN_ID, SEQNO, PARM_SEQ	Global Partitioned		
F1_SYNC_REQ_IN_LOG	Reference Partitioning	F1T194P0	F1_SYNC_REQ_IN_ID, SEQNO	Global Partitioned		
		F1T194S1	CHAR_TYPE_CD, CHAR_VAL_FK1	Global		
		F1T194S2	CHAR_TYPE_CD, CHAR_VAL	Global		

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
F1_SYNC_REQ_IN_LOG_PARM (Child Table of F1_SYNC_REQ_IN_LOG)	Reference Partitioning	F1T195P0	F1_SYNC_REQ_IN_ID, SEQNO, PARM_SEQ	Global Partitioned		
F1_SYNC_REQ_IN_REL_OBJ	Reference Partitioning	F1T192P0	F1_SYNC_REQ_IN_ID, MAINT_OBJ_CD, REL_OBJ_TYPE_FLG	Global Partitioned		
		F1T192S1	PK_VALUE1	Global		

Note: It is recommended that data retention policies and rules for this object match the policies and rules implemented for the Outbound Sync Request on the source system to avoid data inconsistencies when auditing.

Outbound Message

This table describes the Outbound Message maintenance object.

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
F1_OUTMSG (Parent)	RANGE (ILM_DT, OUTMSG_ID)				RANGE (OUMSG_ID)	F1_OUTMSG.CRE_DTTM
		FT010P0	OUTMSG_ID	Global Partitioned		
		FT010S1	OUTMSG_STATUS_FLG, OUTMSG_TYPE_CD	Global		
		CM_ILM_FT010S2	ILM_DT, ILM_ARC_SW, OUTMSG_ID	Local Partitioned		
F1_OUTMSG_ERRPARM	Reference Partitioning	FT011P0	OUTMSG_ID, PARM_SEQ	Global Partitioned		

Service Task

This table describes the Service Task maintenance object.

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
F1_SVC_TASK (Parent)	RANGE (ILM_DT, F1_SVC_TASK_ID)				RANGE (F1_SVC_TASK_ ID_)	F1_SVC_TASK. CRE_DTTM
		F1C474P0	F1_SVC_TASK_ID	Global Partitioned		
		F1C474S1	F1_STASK_TYPE_ CD	Global		
		F1C474S2	BUS_OBJ_CD	Global		
		CM_ILM_F1C474 S3	ILM_DT, ILM_ARC_SW, F1_SVC_TASK_ID	Local Partitioned		
F1_SVC_TASK_ CHAR	Reference Partitioning	F1C476P0	F1_SVC_TASK_ID, CHAR_TYPE_CD, SEQ_NUM	Global Partitioned		
		F1C476S1	SRCH_CHAR_VAL	Global		
F1_SVC_TASK_ LOG	Reference Partitioning	F1C477P0	F1_SVC_TASK_ID, SEQNO	Global Partitioned		
		F1C477S1	CHAR_TYPE_CD, CHAR_VAL_FK1	Global		
		F1C477S2	CHAR_TYPE_CD, CHAR_VAL	Global		
F1_SVC_TASK_ LOG_PARM (Child Table of F1_SVC_TASK_ LOG)	Reference Partitioning	F1C478P0	F1_SVC_TASK_ID, SEQNO, PARM_SEQ	Global Partitioned		
F1_SVC_TASK_ REL_OBJ	Reference Partitioning	F1C479P0	F1_SVC_TASK_ID, MAINT_OBJ_CD, SEQ_NUM	Global Partitioned		

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
		F1C479S1	MAINT_OBJ_CD, PK_VALUE1, PK_VALUE2, PK_VALUE3 PK_VALUE4 PK_VALUE5	Global		

Object Revision

This table describes the Object Revision maintenance object.

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
F1_OBJ_REV (Parent)	RANGE (ILM_DT, REV_ID)				RANGE (REV_ID)	F1_OBJ_REV. STATUS_UPD_DTTM
		FT035P0	REV_ID	Global Partitioned		
		FT035S1	BO_STATUS_CD, BUS_OBJ_CD, REV_ID	Global		
		FT035S2	MAINT_OBJ_CD, PK_VALUE1	Global		
		FT035S3	EXT_REFERENCE_ID, MAINT_OBJ_CD	Global		
		FT035S4	USER_ID, MAINT_OBJ_CD	Global		
		FT035S5	PK_VALUE1	Global		
		CM_ILM_FT035S6	ILM_DT, ILM_ARC_SW, REV_ID	Local Partitioned		

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
F1_OBJ_REV_CHAR	Reference Partitioning	FT037P0	REV_ID, CHAR_TYPE_CD, SEQ_NUM	Global Partitioned		
		FT037S1	SRCH_CHAR_VAL	Global		
F1_OBJ_REV_LOG	Reference Partitioning	FT039P0	REV_ID, SEQNO	Global Partitioned		
F1_OBJ_REV_LOG_PARM (Child Table of F1_OBJ_REV_LOG)	Reference Partitioning	FT040P0	REV_ID, SEQNO, PARM_SEQ	Global Partitioned		

Note: This maintenance object is enabled for ILM, however it is not used in a production environment. It is typically used in a development or configuration environment. Your implementation should review its use of this functionality and consider whether or not it is a candidate for ILM and in which region.

Object Erasure

This table describes the Object Erasure maintenance object.

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
F1_ERASURE_SCHED (Parent)	RANGE (ILM_DT, ERASURE_SCHED_ID)					F1_ERASURE_SCHED. STATUS_UPD_DTTM.
		F1T756P0	ERASURE_SCHED_ID	GLOBAL Partitioned	RANGE (ERASURE_SCHED_ID)	
		CM_ILM_F1T756S1	ILM_DT, ILM_ARCH_SW, ERASURE_SCHED_ID	LOCAL		
F1_ERASURE_SCHED_LOG	Reference partitioning	F1T757P0	ERASURE_SCHED_ID, SEQNO	GLOBAL Partitioned		

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
F1_ERASURE_SCHED_LOG_PARM	Reference partitioning	F1T758P0	ERASURE_SCHED_ID, SEQNO, PARM_SEQ	GLOBAL Partitioned		

Process Flow

This table describes the Process Flow maintenance object.

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
F1_PROC_STORE (Parent)	RANGE (ILM_DT, PROC_STORE_ID)					F1_PROC_STORE. STATUS_UPD_DTTM.
		F1T747P0	PROC_STORE_ID	GLOBAL Partitioned	RANGE (PROC_STORE_ID)	
		CM_ILM_F1T747S1	ILM_DT, ILM_ARCH_SW, PROC_STORE_ID	LOCAL		
F1_PROC_STORE_DTL_ELEMENTS	Reference partitioning	F1T748P0	PROC_STORE_ID, CHAR_TYPE_CD, SEQ_NUM	GLOBAL Partitioned		
F1_PROC_STORE_LOG	Reference partitioning	F1T749P0	PROC_STORE_ID, SEQNO	GLOBAL Partitioned		
F1_PROC_STORE_LOG_PARM	Reference partitioning	F1T750P0	PROC_STORE_ID, SEQNO, PARM_SEQ	GLOBAL Partitioned		

Adjustment

This table describes the Adjustment maintenance object.

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
CI_ADJ (Parent)	RANGE (ILM_DT, ADJ_ID)				RANGE (ADJ_ID)	CI_ADJ.CRE_DT
		XT012P0	ADJ_ID	Global Partitioned		
		XT012S1	SA_ID, ADJ_TYPE_CD	Global		
		XT012S2	XFER_ADJ_ID, ADJ_ID	Global		
		XT012S4	APPR_REQ_ID, ADJ_ID	Global		
		CM_ILM_ XT012S3	ILM_DT, ILM_ARCH_SW, ADJ_ID	Local Partitioned		
CI_ADJ_APREQ	Reference Partitioning	XT160P0	AP_REQ_ID	Global		
		XT160S1	ADJ_ID	Global		
		XT160S2	BATCH_CD, BATCH_NBR	Global		
CI_ADJ_CALC_ LN	Reference Partitioning	XT310P0	ADJ_ID, SEQNO	Global Partitioned		
CI_ADJ_CL_ CHAR	Reference Partitioning	XT309P0	ADJ_ID, SEQNO, CHAR_TYPE_ CD	Global Partitioned		
CI_ADJ_CHAR	Reference Partitioning	XC781P0	ADJ_ID, CHAR_TYPE_ CD, SEQ_NUM	Global Partitioned		
		XC781S1	SRCH_CHAR_VA L	Global		

Approval Request

This table describes the Approval Request maintenance object.

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
CI_APPR_REQ (Parent)	RANGE (ILM_DT, APPR_REQ_ID)				RANGE (APPR_REQ_ID)	MIN(LOG_DTTM) on CI_APPR_REQ_LOG for given APPR_REQ_ID
		XT600P0	APPR_REQ_ID	Global Partitioned		
		CM_ILM_XT600S1	ILM_DT, ILM_ARCH_SW, APPR_REQ_ID	Local partitioned		
CI_APPR_REQ_CHAR	Reference Partitioning	XT601P0	APPR_REQ_ID, CHAR_TYPE_CD, SEQ_NUM	Global Partitioned		
		XT601S1	SRCH_CHAR_VAL	Global		
CI_APPR_REQ_LOG	Reference Partitioning	XT602P0	APPR_REQ_ID, SEQNO	Global Partitioned		
		XT602S1	CHAR_TYPE_CD, CHAR_VAL_FK1	Global		
CI_APPR_REQ_LOG_PARM	Reference Partitioning	XT603P0	APPR_REQ_ID, SEQNO, PARM_SEQ	Global Partitioned		

Bill

This table describes the Bill maintenance object.

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
CI_BILL (Parent)	RANGE (ILM_DT, BILL_ID)				RANGE (BILL_ID)	CI_BILL.CRE_DTTM
		XT033P0	BILL_ID	Global Partitioned		

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
		XT033S1	ACCT_ID, BILL_STAT_ FLG, BILL_CYC_CD, WIN_START_DT ,CR_NOTE_FR_ BILL_ID	Global		
		XT033S2	CR_NOTE_FR_B ILL_ID, BILL_ID	Global		
		XT033S3	ALT_BILL_ID, BILL_STAT_ FLG, BILL_ID	Global		
		XT033S4	OFFCYC_BGEN _ID, BILL_ID	Global		
		XT033S5	DOC_ID, DOC_TYPE_ FLG, BILL_STAT_FLG , BILL_ID	Global		
		CM_ILM_ XT033S6	ILM_DT, ILM_ARCH_SW, BILL_ID	Local Partitioned		
		XT033S7	ACCT_ID, OFFCYC_BGEN _ID, CRE_DTTM	Global		
		XT033S8	LATE_PAY_ CHARGE_SW, LATE_PAY_ CHARGE_DT, BILL_ID	Global		
CI_BILL_CHAR	Reference Partitioning	XT313P0	BILL_ID, CHAR_TYPE_ CD, SEQ_NUM	Global Partitioned		
		XT313S1	SRCH_CHAR_ VAL	Global		
CI_BILL_EXCP	Reference Partitioning	XT038P0	BILL_ID	Global Partitioned		
CI_BILL_MSGS	Reference Partitioning	XT091P0	BILL_ID, BILL_MSG_CD	Global Partitioned		

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
CI_BILL_MSG_PRM	Reference Partitioning	XT085P0	BILL_ID, BILL_MSG_CD, SEQ_NUM	Global Partitioned		
CI_BILL_SA	Reference Partitioning	XT046P0	BILL_ID, SA_ID	Global Partitioned		
		XT046S1	SA_ID	Global		
CI_BILL_ROUTING	Reference Partitioning	XT075P0	BILL_ID, SEQNO	Global Partitioned		
		XT075S1	BATCH_CD, BATCH_NBR, BILL_ID, NO_BATCH_PRT_SW	Global		

Bill Segment

This table describes the Bill Segment maintenance object.

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
CI_BSEG (Parent)	RANGE (ILM_DT, BSEG_ID)				RANGE (BSEG_ID)	CI_BSEG.CRE_DTTM
		XT048P0	BSEG_ID	Global Partitioned		
		XT048S1	BILL_ID	Global		
		XT048S2	SA_ID	Global		
		XT048S3	QUOTE_DTL_ID, BSEG_ID	Global		

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
		CM_ILM_ XT048S4	ILM_DT, ILM_ARCH_SW, BSEG_ID	Local Partitioned		
CI_BSEG_CALC	Reference partitioning	XT072P0	BSEG_ID, HEADER_SEQ	Global Partitioned		
		XT072S1	BILLABLE_CHG _ID, BSEG_ID	Global		
CI_BSEG_CALC _LN	Reference partitioning	XT050P0	BSEG_ID, HEADER_SEQ, SEQNO	Global Partitioned		
CI_BSEG_CL_ CHAR	Reference partitioning	XT056P0	BSEG_ID, HEADER_SEQ, SEQNO, CHAR_TYPE_ CD	Global Partitioned		
CI_BSEG_EXCP	Reference partitioning	XT051P0	BSEG_ID	Global Partitioned		
CI_BSEG_MSG	Reference partitioning	XT080P0	BSEG_ID, BILL_MSG_CD	Global Partitioned		
CI_BSEG_READ	Reference partitioning	XT054P0	BSEG_ID, SP_ID, SEQNO	Global Partitioned		
		XT054S1	SP_ID	Global		
		XT054S2	START_REG_RE AD_ID	Global		
		XT054S3	END_REG_ READ_ID	Global		
CI_BSEG_SQ	Reference partitioning	XT055P0	BSEG_ID, UOM_CD, TOU_CD, SQI_CD	Global Partitioned		
CI_BSEG_ITEM	Reference partitioning	XT053P0	BSEG_ID, SEQNO	Global Partitioned		

Statement

This table describes the Statement maintenance object.

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
CI_STM (Parent)	RANGE (ILM_DT, STM_ID)				RANGE (STM_ID)	CI_STM.STM_DT
		XT088P0	STM_ID	Global Partitioned		
		CM_ILM_ XT088S1	ILM_DT, ILM_ARCH_SW, STM_ID	Local Partitioned		
CI_STM_DTL		XT119P0	STM_DTL_ID	Global		
		XT119S1	STM_ID	Global		

Off Cycle Bill Generator

This table describes the Off Cycle Bill Generator maintenance object.

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
C1_OFFCYC_ BGEN (Parent)	RANGE (ILM_DT, OFFCYC_ BGEN_ID)				RANGE (OFFCYC_BGEN _ID)	C1_OFFCYC_BG EN.STATUS_UP D_DTTM
		XT197P0	OFFCYC_BGEN _ID	Global Partitioned		
		XT197S1	ACCT_ID	Global		
		CM_ILM_ XT197S2	ILM_DT, ILM_ARCH_SW, OFFCYC_BGEN _ID	Local Partitioned		

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
C1_OFFCYC_BGEN_ADJ	Reference partitioning	XT285P0	OFFCYC_BGEN_ID, ADJ_ID	Global Partitioned		
		XT285S1	ADJ_ID	Global		
C1_OFFCYC_BGEN_BCHG	Reference partitioning	XT326P0	OFFCYC_BGEN_ID, BILLABLE_CHG_ID	Global Partitioned		
		XT326S1	BILLABLE_CHG_ID	Global		
C1_OFFCYC_BGEN_CHAR	Reference partitioning	XT343P0	OFFCYC_BGEN_ID, CHAR_TYPE_CD, SEQ_NUM	Global Partitioned		
		XT343S1	SRCH_CHAR_VAL	Global		
C1_OFFCYC_BGEN_LOG	Reference partitioning	XT344P0	OFFCYC_BGEN_ID, SEQNO	Global Partitioned		
		XT344S1	CHAR_TYPE_CD, CHAR_VAL_FK1	Global		
C1_OFFCYC_BGEN_LOG_PARM	Reference partitioning	XT357P0	OFFCYC_BGEN_ID, SEQNO, PARM_SEQ	Global Partitioned		
C1_OFFCYC_BGEN_SA	Reference partitioning	XT359P0	OFFCYC_BGEN_ID, SA_ID	Global Partitioned		
		XT359S1	SA_ID	Global		

Billable Charge

This table describes the Billable Charge maintenance object.

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
CI_BILL_CHG (Parent)	RANGE (ILM_DT, BILLABLE_CHG_ID)				RANGE (BILLABLE_CHG_ID)	CI_BILL_CHG.START_DT

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
		XT035P0	BILLABLE_CHG_ID	Global Partitioned		
		XT035S1	SA_ID	Global		
		CM_ILM_XT035S2	ILM_DT, ILM_ARCH_SW, BILLABLE_CHG_ID	Local Partitioned		
CI_BCHG_READ	Reference Partitioning	XT271P0	BILLABLE_CHG_ID, SP_ID, SEQNO	Global Partitioned		
CI_BCHG_SQ	Reference Partitioning	XT081P0	BILLABLE_CHG_ID, SEQ_NUM	Global Partitioned		
CI_B_CHG_LINE	Reference Partitioning	XT001P0	BILLABLE_CHG_ID, LINE_SEQ	Global Partitioned		
CI_B_LN_CHAR	Reference Partitioning	XT083P0	BILLABLE_CHG_ID, LINE_SEQ, CHAR_TYPE_CD	Global Partitioned		

Case

This table describes the Case maintenance object.

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
CI_CASE (Parent)	RANGE (ILM_DT, CASE_ID)				RANGE (CASE_ID)	MIN(LOG_DTTM) on CI_CASE_LOG table for given CASE_ID
		XT220P0	CASE_ID	Global Partitioned		
		XT220S1	PER_ID	Global		

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
		XT220S2	ACCT_ID	Global		
		XT220S3	PREM_ID	Global		
		XT220S4	USER_ID	Global		
		CM_ILM_ XT220S5	ILM_DT, ILM_ARCH_SW, CASE_ID	Local Partitioned		
CI_CASE_CHAR	Reference Partitioning	XT222P0	CASE_ID, CHAR_TYPE_ CD, SEQ_NUM	Global Partitioned		
		XT222S1	SRCH_CHAR_ VAL	Global		
CI_CASE_LOG	Reference Partitioning	XT221P0	CASE_ID, SEQ_NUM	Global Partitioned		
		XT221S1	CHAR_TYPE_ CD, CHAR_VAL_FK1	Global		
CI_CASE_LOG_ PARM	Reference Partitioning	XT290P0	CASE_ID, SEQ_NUM, PARM_SEQ	Global Partitioned		

Field Activity

This table describes the Field Activity maintenance object.

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
CI_FA (Parent)	RANGE (ILM_DT, FA_ID)				RANGE (FA_ID)	CI_FA.CRE_ DTTM
		XT094P0	FA_ID	Global Partitioned		
		XT094S1	SP_ID, FO_ID	Global		

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
		XT094S2	FO_ID, FA_ID	Global		
		XT094S3	FA_STATUS_ FLG, ELIG_ DISPATCH_SW, FO_ID, FA_ID	Global		
		XT094S4	APP_SCHED_ID, FA_ID	Global		
		XT094S5	TEST_SEL_ID, FA_ID	Global		
		CM_ILM_ XT094S6	ILM_DT, ILM_ARCH_SW, FA_ID	Local Partitioned		
		XT094S7	FA_EXT_ID, FA_ID	Global		
CI_FA_CHAR	Reference Partitioning	XT406P0	FA_ID, CHAR_TYPE_ CD, SEQ_NUM	Global Partitioned		
		XT406S1	SRCH_CHAR_ VAL	Global		
CI_FA_LOG	Reference Partitioning	XT350P0	FA_ID, SEQ_NUM	Global Partitioned		
CI_FA_REM	Reference Partitioning	XT407P0	FA_ID, FA_REM_CD	Global Partitioned		
CI_FA_REM_ EXC	Reference Partitioning	XT312P0	FA_ID, FA_REM_CD	Global Partitioned		
CI_FA_REM_ EXP	Reference Partitioning	XT405P0	FA_ID, FA_REM_CD, PARM_SEQ	Global Partitioned		
CI_FA_STEP	Reference Partitioning	XT095P0	FA_ID, STEP_SEQ_NBR	Global Partitioned		
		XT095S1	CC_ID	Global		
		XT095S2	SPAWNED_FA_ ID, FA_ID	Global		
		XT095S3	MR_ID	Global		

Enrollment (Order)

This table describes the Enrollment (Order) maintenance object.

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
CI_ENRL (Parent)	RANGE (ILM_DT, ENRL_ID)				RANGE (ENRL_ID)	CI_ENRL. START_DT
		XT193P0	ENRL_ID	Global Partitioned		
		XT193S1	PER_ID	Global		
		XT193S2	ACCT_ID	Global		
		XT193S3	PREM_ID	Global		
		CM_ILM_XT193S4	ILM_DT, ILM_ARCH_SW, ENRL_ID	Local Partitioned		
CI_ENRL_ADDR	Reference Partitioning	XT200P0	ENRL_ID, ENRL_ADDR_ ENT_FLG	Global Partitioned		
		XT200S1	ADDRESS1_UPR , ENRL_ID	Global		
		XT200S2	CITY_UPR, ENRL_ID	Global		
CI_ENRL_FLD	Reference Partitioning	XT191P0	ENRL_ID, SEQ_NUM	Global Partitioned		
CI_ENRL_LOG	Reference Partitioning	XT198P0	ENRL_ID, SEQ_NUM	Global Partitioned		
CI_ENRL_PER_ ID	Reference Partitioning	XT199P0	ENRL_ID, ID_TYPE_CD	Global Partitioned		
		XT199S1	HASH_PER_ID_ NBR, ENRL_ID, ID_TYPE_CD	Global		
CI_ENRL_PER_ NM	Reference Partitioning	XT194P0	ENRL_ID, SEQ_NUM	Global Partitioned		
		XT194S1	ENTITY_NAME _UPR, ENRL_ID	Global		
CI_ENRL_PER_ PHN	Reference Partitioning	XT195P0	ENRL_ID, SEQ_NUM	Global Partitioned		

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
C1_ENRL_CONTDET	Reference Partitioning	C1T011S1	C1_ENRL_CONTACT_ID	Global Partitioned		

Payment Event

This table describes the Payment Event maintenance object.

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
CI_PAY_EVENT (Parent)	RANGE (ILM_DT, PAY_EVENT_ID)				RANGE (PAY_EVENT_ID)	CI_PAY_EVENT.PAY_DT
		XT159P0	PAY_EVENT_ID	Global Partitioned		
		XT159S1	PAY_DT, PAY_EVENT_ID	Global		
		XT159S2	DOC_ID, PAY_EVENT_ID	Global		
		CM_ILM_XT159S3	ILM_DT, ILM_ARCH_SW, PAY_EVENT_ID	Local Partitioned		
CI_PAY_EVT_CHAR	Reference Partitioning	XT244P0	PAY_EVENT_ID, CHAR_TYPE_CD, SEQ_NUM	Global Partitioned		
		XT244S1	SRCH_CHAR_VAL	Global		
CI_PAY_EVT_EXCP	Reference Partitioning	XT161P0	PAY_EVENT_ID	Global Partitioned		
CI_PAY_TNDR	Reference Partitioning	XT265P0	PAY_TENDER_ID	Global		
		XT265S1	MICR_ID	Global		
		XT265S2	TENDER_AMT, PAY_TENDER_ID	Global		

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
		XT265S3	PAY_EVENT_ID	Global		
		XT265S4	PAYOR_ACCT_ID, TENDER_AMT	Global		
		XT265S5	TNDR_CTL_ID, PAY_EVENT_ID	Global		
		XT265S6	ADJ_ID	Global		
		XT265S7	HASH_MICR_ID, PAY_TENDER_ID	Global		
CI_APAY_CLR_STG		XT003P0	APAY_CLR_ID	Global		
		XT003S1	BILL_ID	Global		
		XT003S2	PAY_TENDER_ID	Global		
		XT003S3	BATCH_CD, BATCH_NBR, SCHED_EXTRACT_DT	Global		
		XT003S4	ACCT_APAY_ID	Global		
CI_PAY_TNDR_CHAR		XT413P0	PAY_TENDER_ID, CHAR_TYPE_CD, SEQ_NUM	Global		
		XT413S1	SRCH_CHAR_VAL	Global		
CI_PEVT_DST_DTL	Reference Partitioning	XT730P0	PAY_EVENT_ID, SEQ_NUM	Global Partitioned		
		XT730S1	SRCH_CHAR_VAL	Global		

Payment

This table describes the Payment maintenance object.

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
CI_PAY (Parent)	RANGE (ILM_DT, PAY_ID)				RANGE (PAY_ID)	CI_PAY_EVENT. PAY_DT
		XT156P0	PAY_ID	Global Partitioned		
		XT156S1	ACCT_ID	Global		
		XT156S2	PAY_EVENT_ID	Global		
		XT156S3	PAY_AMT, PAY_ID	Global		
		CM_ILM_ XT156S4	ILM_DT, ILM_ARCH_SW, PAY_ID	Local Partitioned		
CI_PAY_CHAR	Reference Partitioning	XT412P0	PAY_ID, CHAR_TYPE_ CD, SEQ_NUM	Global Partitioned		
		XT412S1	SRCH_CHAR_ VAL	Global		
CI_PAY_EXCP	Reference Partitioning	XT163P0	PAY_ID	Global Partitioned		
CI_PAY_SEG	Reference Partitioning	XT165P0	PAY_SEG_ID	Global		
		XT165S1	PAY_ID	Global		
		XT165S2	SA_ID	Global		

Match Event

This table describes the Match Event maintenance object.

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
CI_MATCH_EVT (Parent)	RANGE (ILM_DT, MATCH_EVT_ID)				RANGE (MATCH_EVT_ID)	CI_MATCH_EVT .CREATE_DT
		XT266P0	MATCH_EVT_ID	Global Partitioned		
		XT266S1	ACCT_ID, MEVT_STATUS_FLG	Global		
		XT266S2	PAY_ID	Global		
		CM_ILM_XT266S3	ILM_DT, ILM_ARCH_SW, MATCH_EVT_ID	Local Partitioned		

Usage Request

This table describes the Usage Request maintenance object.

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
C1_USAGE (Parent)	RANGE (ILM_DT, USAGE_ID)				RANGE (USAGE_ID)	C1_USAGE.CRE_DTTM
		CT368S2	BSEG_ID	Global		
		CT368S3	USER_ID, USAGE_ID	Global		
	Reference Partitioning	XT368P0	USAGE_ID	Global Partitioned		

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
		XT368S1	BUS_OBJ_CD, BO_STATUS_CD, WIN_START_DT, BILL_CYC_CD, USAGE_ID	Global		
		CT368S2	BSEG_ID			
		CT368S3	USER_ID, USAGE_ID			
		XT368S4	SA_ID	Global		
		CT368S4	MASTER_USAGE_ID			
		CT368S5	SA_REL_ID			
		CM_ILM_XT368S5	ILM_DT, ILM_ARCH_SW, USAGE_ID	Local Partitioned		
C1_USAGE_CHAR	Reference Partitioning	XT387P0	USAGE_ID, CHAR_TYPE_CD, SEQ_NUM	Global Partitioned		
		XT387S1	SRCH_CHAR_VAL	Global		
C1_USAGE_LOG	Reference Partitioning	XT388P0	USAGE_ID, SEQNO	Global Partitioned		
		XT388S1	CHAR_TYPE_CD, CHAR_VAL_FK1	Global		
C1_USAGE_LOG_PARM	Reference Partitioning	XT389P0	USAGE_ID, SEQNO, PARM_SEQ	Global Partitioned		

Business Flag

This table describes the Business Flag maintenance object.

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
F1_BUS_FLG (Parent)	RANGE (ILM_DT,BUS_FLG_ID)				RANGE(BUS_FLG_ID)	F1_BUS_FLG. CRE_DTTM
		F1T681P0	BUS_FLG_ID	Global Partitioned		
		F1T681S1	BUS_OBJ_CD, BO_STATUS_CD, BUS_FLG_ID	Global		
		CM_ILM_ F1T681S2	ILM_DT, ILM_ARCH_SW, BUS_FLG_ID	Local Partitioned		
F1_BUS_FLG_ CHAR	Reference Partitioning	F1T684P0	BUS_FLG_ID, CHAR_TYPE_CD, SEQ_NUM	Global Partitioned		
		F1T684S0	SRCH_CHAR_VAL	Global		
F1_BUS_FLG_ LOG	Reference Partitioning	F1T685P0	BUS_FLG_ID, SEQNO	Global Partitioned		
		F1T685S1	CHAR_TYPE_CD, CHAR_VAL_FK1	Global		
		F1T685S2	CHAR_TYPE_CD, CHAR_VAL	Global		
F1_BUS_FLG_ LOG_PARM	Reference Partitioning	F1T686P0	BUS_FLG_ID, SEQNO, PARM_SEQ	Global Partitioned		
F1_BUS_FLG_ REL	Reference Partitioning	F1T682P0	BUS_FLG_ID, BUS_FLG_REL_ TYPE_FLG, SEQ_NUM	Global Partitioned		
F1_BUS_FLG_ REL_OBJ	Reference Partitioning	F1T683P0	BUS_FLG_ID, BUS_FLG_REL_ OBJ_TYPE_FLG, SEQ_NUM	Global Partitioned		

Remote Message

This table describes the Remote Message maintenance object.

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
F1_REMOTE_MSG (Parent)	RANGE (ILM_DT,F1_REMOTE_MSG_ID)				RANGE(F1_REMOTE_MSG_ID)	F1_REMOTE_MSG.CRE_DTTM
		F1T735P0	F1_REMOTE_MSG_ID	Global Partitioned		
		F1T735S1	CRE_DTTM	Global		
		F1T735S2	F1_MDT_ID	Global		
		F1T735S3	MAINT_OBJ_CD	Global		
		F1T735S4	PK_VALUE1	Global		
		F1T735S5	F1_DEVICE_MSG_ID	Global		
		F1T735S6	F1_MDT_ID, F1_MSG_CLASS_FLG, F1_DELIVERY_STATE_FLG	Global		
		CM_ILM_F1T735S7	ILM_DT, ILM_ARCH_SW, F1_REMOTE_MSG_ID	Local Partitioned		
F1_REMOTE_MSG_CHA	Reference Partitioning	F1T736P0	F1_REMOTE_MSG_ID, CHAR_TYPE_CD, SEQ_NUM	Global Partitioned		
		F1T736S1	SRCH_CHAR_VAL	Global		
F1_REMOTE_MSG_LOG	Reference Partitioning	F1T737P0	F1_REMOTE_MSG_ID, SEQNO	Global Partitioned		
		F1T737S1	CHAR_TYPE_CD, CHAR_VAL_FK1	Global		
		F1T737S2	CHAR_TYPE_CD, CHAR_VAL	Global		
F1_REMOTE_MSG_LOG_PARM	Reference Partitioning	F1T738P0	F1_REMOTE_MSG_ID, SEQNO, PARM_SEQ	Global Partitioned		

Statistics Snapshot

This table describes the Statistics Snapshot maintenance object.

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
F1_STATS_ SNPSHT(Parent)	RANGE (ILM_DT, SNAPSHOT_ID)				RANGE (SNAPSHOT_ID)	F1_STATS_ SNPSHT. CRE_DTTM
		F1C706P0	SNAPSHOT_ID	Global Partitioned		
		F1C706S1	BUS_OBJ_CD, BO_STATUS_CD, SNAPSHOT_ID	Global		
		CM_ILM_ F1C706S2	ILM_DT, ILM_ARCH_SW, SNAPSHOT_ID	Local		
F1_STATS_ SNPSHT_CHAR	Reference Partitioning	F1C707P0	SNAPSHOT_ID, CHAR_TYPE_CD, SEQ_NUM	Global Partitioned		
		F1C707S1	SRCH_CHAR_VAL	Global		
F1_STATS_ SNPSHT_LOG	Reference Partitioning	F1C708P0	SNAPSHOT_ID, SEQNO	Global Partitioned		
		F1C708S1	CHAR_TYPE_CD, CHAR_VAL_FK1	Global		
		F1C708S2	CHAR_TYPE_CD, CHAR_VAL	Global		
F1_STATS_SNPSHT _LOG_PARM	Reference Partitioning	F1C709P0	SNAPSHOT_ID, SEQNO, PARM_SEQ	Global Partitioned		
F1_STATS_ SNPSHT_REL_ OBJ	Reference Partitioning	F1C710P0	SNAPSHOT_ID, STATS_SNPSHT_ REL_OBJ_TYPE_ FLG, SEQ_NUM	Global Partitioned		

Customer Relationship Request

This table describes the Customer Relationship Request maintenance object.

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
C1_CUST_REL_REQ	RANGE(ILM_DT,CUST_REL_REQ_ID)				RANGE(CUST_REL_REQ_ID)	C1_CUST_REL_REQ.CRE_DTTM
		C1T017P0	CUST_REL_REQ_ID	Global Partitioned		
		C1T017S1	CUST_REL_REQ_ID, BUS_OBJ_CD, BO_STATUS_CD	Global		
		CM_ILM_C1T017S2	ILM_DT, ILM_ARCH_SW, CUST_REL_REQ_ID	Local		
C1_CUST_REL_REQ_CHAR	Reference Partitioning	C1T014P0	CUST_REL_REQ_ID, CHAR_TYPE_CD, SEQ_NUM	Global Partitioned		
		C1T014S1	SRCH_CHAR_VAL	Global		
C1_CUST_REL_REQ_LOG	Reference Partitioning	C1T015P0	CUST_REL_REQ_ID, SEQNO	Global Partitioned		
		C1T015S1	CHAR_TYPE_CD, CHAR_VAL_FK1	Global		
		C1T015S2	CHAR_TYPE_CD, CHAR_VAL	Global		
C1_CUST_REL_REQ_LOG_PARM	Reference Partitioning	C1T016P0	CUST_REL_REQ_ID, SEQNO, PARM_SEQ	Global Partitioned		
C1_CUST_REL_REQ_REL_OBJ	Reference Partitioning	C1T018P0	CUST_REL_REQ_ID, CRR_REL_OBJ_TY_FLG, SEQ_NUM	Global Partitioned		

Customer Contact

This table describes the Customer Contact maintenance object.

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
CI_CC	RANGE(ILM_DT ,CC_ID)				RANGE(CC_ID)	CI_CC. CC_DTTM, CI_CC. LETTER_PRINT _DTTM
		XT057P0	CC_ID	Global Partitioned		
		XT057S1	PER_ID	Global		
		XT057S2	BATCH_CD, BATCH_NBR, CC_ID, PRINT_LETTER_ SW	Global		
		XT057S3	CC_TYPE_CD, CC_CL_CD, CC_ID	Global		
		XT057S4	ACCT_ID	Global		
		XT057S6	C1_CONTACT_ID	Global		
		CM_ILM_XT057S 7	ILM_DT,ILM_ARC H_SW,CC_ID	Local		
CI_CC_CHAR	Partitioning		CHAR_TYPE_CD	Partitioned	RANGE(CC_ID)	
		C1T007S1	SRCH_CHAR_VAL	Global		
CI_CC_LOG	Reference Partitioning	XT281P0	CC_LOG_ID	Global Partitioned	RANGE(CC_LO G_ID)	
		XT281S1	CC_ID	Global		

Collection Process

This table describes the Collection Process maintenance object.

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
CI_COLL_PROC	RANGE(CRE_D TTM,COLL_PRO C_ID)				RANGE (COLL_PROC_ ID)	N/A

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
		XT073P0	COLL_PROC_ID	Global Partitioned		
		XT073S1	ACCT_ID	Global		
			COLL_STATUS_FLG			
CI_COLL_EVT	Reference Partitioning	XT069P0	COLL_PROC_ID, EVT_SEQ	Global Partitioned		
		XT069S0	TRIGGER_BP_SW, COLL_PROC_ID, EVT_SEQ, COLL_EVT_TYP_CD	Global		
CI_COLL_EVT_CC	Reference Partitioning	XT070P0	COLL_PROC_ID, EVT_SEQ, CC_ID	Global Partitioned		
		XT070S1	CC_ID	Global		
CI_COLL_PROC_SA	Reference Partitioning	XT074P0	COLL_PROC_ID, SA_ID	Global Partitioned		
		XT074S1	SA_ID	Global		

Cut Process

This table describes the Cut Process maintenance object.

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
CI_CUT_PROC	RANGE(CRE_DTTM,CUT_PROC_ID)				RANGE (CUT_PROC_ID)	N/A
		XT324P0	CUT_PROC_ID	Global Partitioned		
		XT324S1	OD_PROC_ID, OD_EVT_SEQ	Global		
CI_CUT_EVT	Reference Partitioning	XT322P0	CUT_PROC_ID, EVT_SEQ	Global Partitioned		
CI_CUT_EVT_DEP	Reference Partitioning	XT323P0	CUT_PROC_ID, EVT_SEQ, SEQ_NUM	Global Partitioned		

Overdue Process

This table describes the Overdue Process maintenance object.

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
CI_OD_PROC	RANGE(CRE_DTTM,OD_PROC_ID)				RANGE (OD_PROC_ID)	N/A
		XT315P0	OD_PROC_ID	Global Partitioned		
CI_OD_EVT	Reference Partitioning	XT318P0	OD_PROC_ID, EVT_SEQ	Global Partitioned		
CI_OD_EVT_DEP	Reference Partitioning	XT319P0	OD_PROC_ID, EVT_SEQ, SEQ_NUM	Global Partitioned		
CI_OD_PROC_LOG	Reference Partitioning	XT320P0	OD_PROC_ID, LOG_SEQ	Global Partitioned		
		XT320S1	CHAR_TYPE_CD, CHAR_VAL_FK1	Global		
CI_OD_PROC_LOGPARM	Reference Partitioning	XT321P0	OD_PROC_ID, LOG_SEQ, PARM_SEQ	Global Partitioned		
CI_OD_PROC_OBJ	Reference Partitioning	XT317P0	OD_PROC_ID, SEQ_NUM	Global Partitioned		

Severance Event

This table describes the Severance Event maintenance object.

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
CI_SEV_PROC	RANGE(CRE_DTTM,SEV_PROC_ID)				RANGE (SEV_PROC_ID)	N/A
		XT118P0	SEV_PROC_ID	Global Partitioned		
		XT118S1	SA_ID, SEV_PROC_ID	Global		
		XT118S2	COLL_PROC_ID, EVT_SEQ	Global		
CI_SEV_EVT	Reference Partitioning	XT214P0	SEV_PROC_ID, EVT_SEQ	Global Partitioned		

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
CI_SEV_EVT_DEP	Reference Partitioning	XT216P0	SEV_PROC_ID, EVT_SEQ, SEQ_NUM	Global Partitioned		
CI_SEV_EVT_FA	Reference Partitioning	XT217P0	SEV_PROC_ID, EVT_SEQ, FA_ID	Global Partitioned		
		XT217S1	FA_ID	Global		
CI_SEV_EVT_CC	Reference Partitioning	XT215P0	SEV_PROC_ID, EVT_SEQ, CC_ID	Global Partitioned		
		XT215S1	CC_ID	Global		

WO Process

This table describes the WO Process maintenance object.

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
CI_WO_PROC	RANGE(CRE_DTTM,WO_PROC_ID)				RANGE (WO_PROC_ID)	N/A
		XT061P0	WO_PROC_ID	Global Partitioned		
		XT061S1	ACCT_ID, WO_STATUS_FLG	Global		
CI_WO_EVT	Reference Partitioning	XT059P0	WO_PROC_ID, EVT_SEQ	Global Partitioned		
CI_WO_EVT_CC	Reference Partitioning	XT060P0	WO_PROC_ID, EVT_SEQ, CC_ID	Global Partitioned		
		XT060S1	CC_ID	Global		
CI_WO_PROC_SA	Reference Partitioning	XT062P0	WO_PROC_ID, SA_ID	Global Partitioned		
		XT062S1	SA_ID, WO_SA_STAT_FLG	Global		

General Audit

This table describes the General Audit maintenance object.

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
F1_GNRL_AUDIT	RANGE(ILM_DT, AUDIT_ID)				RANGE (AUDIT_ID)	F1_GNRL_AUDIT.CRE_DTTM
		F1T901P0	AUDIT_ID	Global Partitioned		
		F1T901S1	USER_ID	Global		
		F1T901S1	AUDIT_ID, USER_ID, CRE_DTTM	Global		
		CM_ILM_F1T901S3	ILM_DT, ILM_ARCH_SW, AUDIT_ID	Local		
F1_GNRL_AUDIT_CHAR	Reference Partitioning	F1C504P0	AUDIT_ID, CHAR_TYPE_CD, SEQ_NUM	Global Partitioned		
		F1C504S1	SRCH_CHAR_VAL	Global		
F1_GNRL_AUDIT_VAL	Reference Partitioning	F1T902P0	AUDIT_ID, FLD_NAME	Global Partitioned		
		F1T902S1	AUDIT_ID, FLD_NAME, FLD_VAL	Global		

Meter Read

This table describes the Meter Read maintenance object.

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
CI_MR	RANGE(ILM_DT,MR_ID)				RANGE(MR_ID)	CI_MR.READ_DTTM,
		XT132P0	MR_ID	Global Partitioned		
		XT132S1	MTR_CONFIG_ID, USE_ON_BILL_SW, READ_DTTM, MR_ID	Global		
		CM_ILM_XT132L0	ILM_DT,ILM_ARCH_SW,MR_ID	Local		

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
CI_MR_CHAR	Reference Partitioning	XT410P0	MR_ID, CHAR_TYPE_CD, SEQ_NUM	Global Partitioned	RANGE(MR_ID)	
		XTT410S1	SRCH_CHAR_VAL	Global		
CI_MR_REM	Reference Partitioning	XT135P0	MR_ID, READER_REM_CD	Global Partitioned	RANGE(MR_ID)	
CI_MR_REM_EXCP	Reference Partitioning	XT024P0	MR_ID, READER_REM_CD	Global Partitioned	RANGE(MR_ID)	
CI_REG_READ	Reference Partitioning	XT186P0	REG_READ_ID	Global Partitioned	RANGE(REG_READ_ID)	
		XT186S1	MR_ID	Global		
		XT186S2	REVIEW_HILO_SW, TRENDED_SW, REG_READ_ID	Global		
		XT186S3	REG_ID	Global		

Payment Arrangement Process

This table describes the Payment Arrangement Process maintenance object.

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
C1_PA_RQST (Parent)	RANGE (ILM_DT,PA_RQST_ID)					C1_PA_RQST.CRE_DTTM
		C1T020P0	PA_RQST_ID	Global Partitioned	RANGE (PA_RQST_ID)	
		C1T020S1	PA_RQST_ID, BUS_OBJ_CD, BO_STATUS_CD	Global		
		CM_ILM_C1T020S1	ILM_DT,ILM_ARCH_SW,PA_RQST_ID	Local		
C1_PA_RQST_CHAR	Reference Partitioning	C1T024P0	PA_RQST_ID, CHAR_TYPE_CD, SEQ_NUM	Global Partitioned		
		C1T024S1	SRCH_CHAR_VAL	Global		
C1_PA_RQST_ELIG_RSLT	Reference Partitioning	C1T022P0	PA_RQST_ID, PA_RQST_ELIG_CRIT_SEQ	Global Partitioned		

Table Name	Table Partitioning Type (Partitioning, Sub-Partitioning Key)	Index Name	Index Columns	Index Type Global or Local	Index Partitioning Sub-Partitioning Key	ILM_DT Initial Load
C1_PA_RQST_LOG	Reference Partitioning	C1T025P0	PA_RQST_ID, SEQNO	Global Partitioned		
		C1T025S1	CHAR_TYPE_CD, CHAR_VAL_FK1	Global		
		C1T025S2	CHAR_TYPE_CD, CHAR_VAL	Global		
C1_PA_RQST_LOG_PARM	Reference Partitioning	C1T026P0	PA_RQST_ID, SEQNO, PARM_SEQ	Global Partitioned		
C1_PA_RQST_REL_OBJ	Reference Partitioning	C1T023P0	PA_RQST_ID, PA_RQST_REL_OBJ_TYPE_FLG, SEQ_NUM	Global Partitioned		

Appendix A

Sample SQL for Enabling ILM in CCB (Initial Install)

This section provides more detail about steps needed to fully support ILM on tables for maintenance objects that support the functionality. Two maintenance objects are shown:

- To Do Entry, which does not include a LOB field.
- Sync Request, which does include a LOB field.

  Other maintenance object's implementations can follow the appropriate pattern based on whether there is a LOB field or not.

The following DDL(s):

- Follow Naming convention recommendations for partitions\subpartitions\tablespaces.
- Ensure all the ILM Storage requirements are incorporated, failing which, ILM functionality will not be achieved.
  - Partitions are defined with respective Tablespace.
  - Child Tables are referenced partitioned.
- Ensure all compression recommendations are incorporated.

Maintenance Object: TO DO ENTRY

Parent Table: CI_TD_ENTRY

```
CREATE BIGFILE TABLESPACE CM_XT039_P2017JAN DATAFILE '+DATADG' SIZE
50M AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS
ADVANCED;
CREATE BIGFILE TABLESPACE CM_XT039_P2017FEB DATAFILE '+DATADG' SIZE
50M AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS
ADVANCED;
CREATE BIGFILE TABLESPACE CM_XT039_P2017MAR DATAFILE '+DATADG' SIZE
50M AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS
ADVANCED;
CREATE BIGFILE TABLESPACE CM_XT039_P2017APR DATAFILE '+DATADG' SIZE
50M AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS
ADVANCED;
```

```
CREATE BIGFILE TABLESPACE CM_XT039_P2017MAY DATAFILE '+DATADG' SIZE
50M AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS
ADVANCED;
CREATE BIGFILE TABLESPACE CM_XT039_P2017JUN DATAFILE '+DATADG' SIZE
50M AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS
ADVANCED;
CREATE BIGFILE TABLESPACE CM_XT039_P2017JUL DATAFILE '+DATADG' SIZE
50M AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS
ADVANCED;
CREATE BIGFILE TABLESPACE CM_XT039_P2017AUG DATAFILE '+DATADG' SIZE
50M AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS
ADVANCED;
CREATE BIGFILE TABLESPACE CM_XT039_P2017SEP DATAFILE '+DATADG' SIZE
50M AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS
ADVANCED;
CREATE BIGFILE TABLESPACE CM_XT039_P2017OCT DATAFILE '+DATADG' SIZE
50M AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS
ADVANCED;
CREATE BIGFILE TABLESPACE CM_XT039_P2017NOV DATAFILE '+DATADG' SIZE
50M AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS
ADVANCED;
CREATE BIGFILE TABLESPACE CM_XT039_P2017DEC DATAFILE '+DATADG' SIZE
50M AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS
ADVANCED;
CREATE BIGFILE TABLESPACE CM_XT039_PMAX DATAFILE '+DATADG' SIZE 50M
AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;

CREATE TABLE CI_TD_ENTRY (
TD_ENTRY_ID     CHAR(14) NOT NULL ENABLE,
BATCH_CD        CHAR(8) DEFAULT ' ' NOT NULL ENABLE,
BATCH_NBR       NUMBER(10,0) DEFAULT 0 NOT NULL ENABLE,
MESSAGE_CAT_NBR NUMBER(5,0) DEFAULT 0 NOT NULL ENABLE,
MESSAGE_NBR     NUMBER(5,0) DEFAULT 0 NOT NULL ENABLE,
ASSIGNED_TO     CHAR(8) DEFAULT ' ' NOT NULL ENABLE,
TD_TYPE_CD      CHAR(8) DEFAULT ' ' NOT NULL ENABLE,
ROLE_ID         CHAR(10) DEFAULT ' ' NOT NULL ENABLE,
ENTRY_STATUS_FLG CHAR(2) DEFAULT ' ' NOT NULL ENABLE,
VERSION         NUMBER(5,0) DEFAULT 1 NOT NULL ENABLE,
CRE_DTTM DATE,
ASSIGNED_DTTM DATE,
COMPLETE_DTTM DATE,
COMPLETE_USER_ID CHAR(8) DEFAULT ' ' NOT NULL ENABLE,
COMMENTS        VARCHAR2(254) DEFAULT ' ' NOT NULL ENABLE,
ASSIGNED_USER_ID CHAR(8) DEFAULT ' ' NOT NULL ENABLE,
TD_PRIORITY_FLG CHAR(4) DEFAULT ' ' NOT NULL ENABLE,
ILM_DT DATE,
ILM_ARCH_SW CHAR(1)
)
ENABLE ROW MOVEMENT
PARTITION BY RANGE (ILM_DT)
SUBPARTITION BY RANGE (TD_ENTRY_ID) SUBPARTITION TEMPLATE
(
SUBPARTITION S01 VALUES LESS THAN ( '12499999999999' ),
SUBPARTITION S02 VALUES LESS THAN ( '24999999999999' ),
SUBPARTITION S03 VALUES LESS THAN ( '37499999999999' ),
SUBPARTITION S04 VALUES LESS THAN ( '49999999999999' ),
SUBPARTITION S05 VALUES LESS THAN ( '62499999999999' ),
SUBPARTITION S06 VALUES LESS THAN ( '74999999999999' ),
SUBPARTITION S07 VALUES LESS THAN ( '87499999999999' ),
SUBPARTITION SMAX VALUES LESS THAN ( MAXVALUE )
)
(
```

```
PARTITION "P2017JAN" VALUES LESS THAN (TO_DATE('2017-02-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT039_P2017JAN,
PARTITION "P2017FEB" VALUES LESS THAN (TO_DATE('2017-03-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT039_P2017FEB,
PARTITION "P2017MAR" VALUES LESS THAN (TO_DATE('2017-04-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT039_P2017MAR,
PARTITION "P2017APR" VALUES LESS THAN (TO_DATE('2017-05-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT039_P2017APR,
PARTITION "P2017MAY" VALUES LESS THAN (TO_DATE('2017-06-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT039_P2017MAY,
PARTITION "P2017JUN" VALUES LESS THAN (TO_DATE('2017-07-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT039_P2017JUN,
PARTITION "P2017JUL" VALUES LESS THAN (TO_DATE('2017-08-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT039_P2017JUL,
PARTITION "P2017AUG" VALUES LESS THAN (TO_DATE('2017-09-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT039_P2017AUG,
PARTITION "P2017SEP" VALUES LESS THAN (TO_DATE('2017-10-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT039_P2017SEP,
PARTITION "P2017OCT" VALUES LESS THAN (TO_DATE('2017-11-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT039_P2017OCT,
PARTITION "P2017NOV" VALUES LESS THAN (TO_DATE('2017-12-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT039_P2017NOV,
PARTITION "P2017DEC" VALUES LESS THAN (TO_DATE('2018-01-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT039_P2017DEC,
PARTITION "PMAX" VALUES LESS THAN (MAXVALUE)
tablespace CM_XT039_PMAX
);
```

INDEX

```
CREATE BIGFILE TABLESPACE CM_XT039_IND DATAFILE '+DATADG' SIZE 50M
AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;

CREATE UNIQUE INDEX XT039P0 ON CI_TD_ENTRY ( TD_ENTRY_ID ) TABLESPACE
CM_XT039_IND
GLOBAL PARTITION BY RANGE (TD_ENTRY_ID)
(
PARTITION P1 VALUES LESS THAN ( '12499999999999' ),
PARTITION P2 VALUES LESS THAN ( '24999999999999' ),
PARTITION P3 VALUES LESS THAN ( '37499999999999' ),
PARTITION P4 VALUES LESS THAN ( '49999999999999' ),
PARTITION P5 VALUES LESS THAN ( '62499999999999' ),
PARTITION P6 VALUES LESS THAN ( '74999999999999' ),
PARTITION P7 VALUES LESS THAN ( '87499999999999' ),
PARTITION P8 VALUES LESS THAN ( MAXVALUE )
);

ALTER TABLE CI_TD_ENTRY ADD CONSTRAINT XT039P0 PRIMARY
KEY(TD_ENTRY_ID) USING INDEX;
```

```
CREATE UNIQUE INDEX XT039S2 ON CI_TD_ENTRY ( ASSIGNED_TO, TD_ENTRY_ID
) TABLESPACE CM_XT039_IND COMPRESS ADVANCED LOW;

CREATE INDEX XT039S3 ON CI_TD_ENTRY ( ENTRY_STATUS_FLG, ASSIGNED_TO )
TABLESPACE CM_XT039_IND COMPRESS ADVANCED LOW;

CREATE INDEX XT039S4 ON CI_TD_ENTRY ( ROLE_ID, TD_TYPE_CD,
ENTRY_STATUS_FLG, TD_PRIORITY_FLG, ASSIGNED_TO, CRE_DTTM ) TABLESPACE
CM_XT039_IND COMPRESS ADVANCED LOW;

CREATE INDEX XT039S5 ON CI_TD_ENTRY ( BATCH_CD, BATCH_NBR,
ENTRY_STATUS_FLG ) TABLESPACE CM_XT039_IND COMPRESS ADVANCED LOW;

CREATE UNIQUE INDEX XT039S6 ON CI_TD_ENTRY ( TD_ENTRY_ID, ASSIGNED_TO,
ENTRY_STATUS_FLG )TABLESPACE CM_XT039_IND COMPRESS ADVANCED LOW;

CREATE UNIQUE INDEX XT039S7 ON CI_TD_ENTRY ( COMPLETE_USER_ID,
COMPLETE_DTTM, TD_ENTRY_ID ) TABLESPACE CM_XT039_IND COMPRESS ADVANCED
LOW;

CREATE UNIQUE INDEX ILM_XT039S8 ON CI_TD_ENTRY ( ILM_DT, ILM_ARCH_SW,
TD_ENTRY_ID ) LOCAL COMPRESS ADVANCED LOW;
```

Child Table: CI_TD_DRLKEY

```
CREATE TABLE CI_TD_DRLKEY
(
TD_ENTRY_ID CHAR(14) NOT NULL ENABLE,
SEQ_NUM    NUMBER(3,0) NOT NULL ENABLE,
KEY_VALUE  VARCHAR2(50) DEFAULT ' ' NOT NULL ENABLE,
VERSION    NUMBER(5,0) DEFAULT 1 NOT NULL ENABLE,
CONSTRAINT CI_TD_DRLKEY_FK FOREIGN KEY(TD_ENTRY_ID) REFERENCES
CI_TD_ENTRY ON DELETE CASCADE
)
PARTITION BY REFERENCE (CI_TD_DRLKEY_FK)
ENABLE ROW MOVEMENT;
```

INDEX

```
CREATE UNIQUE INDEX XT037P0 ON CI_TD_DRLKEY ( TD_ENTRY_ID, SEQ_NUM )
TABLESPACE CM_XT039_IND
GLOBAL PARTITION BY RANGE (TD_ENTRY_ID)
(
PARTITION P1 VALUES LESS THAN ( '12499999999999' ),
PARTITION P2 VALUES LESS THAN ( '24999999999999' ),
PARTITION P3 VALUES LESS THAN ( '37499999999999' ),
PARTITION P4 VALUES LESS THAN ( '49999999999999' ),
PARTITION P5 VALUES LESS THAN ( '62499999999999' ),
PARTITION P6 VALUES LESS THAN ( '74999999999999' ),
PARTITION P7 VALUES LESS THAN ( '87499999999999' ),
PARTITION P8 VALUES LESS THAN ( MAXVALUE )
)
COMPRESS ADVANCED LOW;

ALTER TABLE CI_TD_DRLKEY ADD CONSTRAINT XT037P0 PRIMARY
KEY(TD_ENTRY_ID, SEQ_NUM) USING INDEX;
```

```
CREATE INDEX XT037S1 ON CI_TD_DRLKEY ( KEY_VALUE, TD_ENTRY_ID )
TABLESPACE CM_XT039_IND COMPRESS ADVANCED LOW;
```

Child Table: CI_TD_ENTRY_CHA

```
CREATE TABLE CI_TD_ENTRY_CHA
(
TD_ENTRY_ID   CHAR(14) NOT NULL ENABLE,
CHAR_TYPE_CD   CHAR(8) NOT NULL ENABLE,
SEQ_NUM       NUMBER(3,0) DEFAULT 0 NOT NULL ENABLE,
CHAR_VAL       CHAR(16) DEFAULT ' ' NOT NULL ENABLE,
VERSION        NUMBER(5,0) DEFAULT 1 NOT NULL ENABLE,
ADHOC_CHAR_VAL VARCHAR2(254) DEFAULT ' ' NOT NULL ENABLE,
CHAR_VAL_FK1   VARCHAR2(50) DEFAULT ' ' NOT NULL ENABLE,
CHAR_VAL_FK2   VARCHAR2(50) DEFAULT ' ' NOT NULL ENABLE,
CHAR_VAL_FK3   VARCHAR2(50) DEFAULT ' ' NOT NULL ENABLE,
CHAR_VAL_FK4   VARCHAR2(50) DEFAULT ' ' NOT NULL ENABLE,
CHAR_VAL_FK5   VARCHAR2(50) DEFAULT ' ' NOT NULL ENABLE,
SRCH_CHAR_VAL  VARCHAR2(50) DEFAULT ' ' NOT NULL ENABLE,
CONSTRAINT CI_TD_ENTRY_CHA_FK FOREIGN KEY(TD_ENTRY_ID) REFERENCES
CI_TD_ENTRY ON DELETE CASCADE
)
PARTITION BY REFERENCE (CI_TD_ENTRY_CHA_FK)
ENABLE ROW MOVEMENT;
```

INDEX

```
CREATE UNIQUE INDEX XT701P0 ON CI_TD_ENTRY_CHA ( TD_ENTRY_ID,
CHAR_TYPE_CD, SEQ_NUM ) TABLESPACE CM_XT039_IND
GLOBAL PARTITION BY RANGE (TD_ENTRY_ID)
(
PARTITION P1 VALUES LESS THAN ( '12499999999999' ),
PARTITION P2 VALUES LESS THAN ( '24999999999999' ),
PARTITION P3 VALUES LESS THAN ( '37499999999999' ),
PARTITION P4 VALUES LESS THAN ( '49999999999999' ),
PARTITION P5 VALUES LESS THAN ( '62499999999999' ),
PARTITION P6 VALUES LESS THAN ( '74999999999999' ),
PARTITION P7 VALUES LESS THAN ( '87499999999999' ),
PARTITION P8 VALUES LESS THAN ( MAXVALUE )
)
COMPRESS ADVANCED LOW;

ALTER TABLE CI_TD_ENTRY_CHA ADD CONSTRAINT XT701P0 PRIMARY
KEY(TD_ENTRY_ID, CHAR_TYPE_CD, SEQ_NUM) USING INDEX;

CREATE INDEX XT701S1 ON CI_TD_ENTRY_CHA ( SRCH_CHAR_VAL, CHAR_TYPE_CD,
TD_ENTRY_ID ) TABLESPACE CM_XT039_IND COMPRESS ADVANCED LOW;
```

Child Table: CI_TD_LOG

```
CREATE TABLE CI_TD_LOG
(
TD_ENTRY_ID CHAR(14) NOT NULL ENABLE,
SEQ_NUM    NUMBER(3,0) NOT NULL ENABLE,
LOG_DTTM DATE NOT NULL ENABLE,
LOG_TYPE_FLG CHAR(4) DEFAULT ' ' NOT NULL ENABLE,
```

```
USER_ID     CHAR(8) DEFAULT ' ' NOT NULL ENABLE,
ASSIGNED_TO CHAR(8) DEFAULT ' ' NOT NULL ENABLE,
VERSION     NUMBER(5,0) DEFAULT 1 NOT NULL ENABLE,
DESCRLONG   VARCHAR2(4000) DEFAULT ' ' NOT NULL ENABLE,
CONSTRAINT CI_TD_LOG_FK FOREIGN KEY(TD_ENTRY_ID) REFERENCES
CI_TD_ENTRY ON DELETE CASCADE
)
PARTITION BY REFERENCE (CI_TD_LOG_FK)
ENABLE ROW MOVEMENT;
```

INDEX

```
CREATE UNIQUE INDEX XT721P0 ON CI_TD_LOG ( TD_ENTRY_ID, SEQ_NUM )
TABLESPACE CM_XT039_IND
GLOBAL PARTITION BY RANGE (TD_ENTRY_ID)
(
PARTITION P1 VALUES LESS THAN ( '12499999999999' ),
PARTITION P2 VALUES LESS THAN ( '24999999999999' ),
PARTITION P3 VALUES LESS THAN ( '37499999999999' ),
PARTITION P4 VALUES LESS THAN ( '49999999999999' ),
PARTITION P5 VALUES LESS THAN ( '62499999999999' ),
PARTITION P6 VALUES LESS THAN ( '74999999999999' ),
PARTITION P7 VALUES LESS THAN ( '87499999999999' ),
PARTITION P8 VALUES LESS THAN ( MAXVALUE )
)
COMPRESS ADVANCED LOW;

ALTER TABLE CI_TD_LOG ADD CONSTRAINT XT721P0 PRIMARY KEY(TD_ENTRY_ID,
SEQ_NUM) USING INDEX;

CREATE INDEX XT721S1 ON CI_TD_LOG ( LOG_DTTM,  USER_ID, LOG_TYPE_FLG,
TD_ENTRY_ID ) TABLESPACE CM_XT039_IND COMPRESS ADVANCED LOW;
```

Child Table: CI_TD_MSG_PARM

```
CREATE TABLE CI_TD_MSG_PARM
(
TD_ENTRY_ID CHAR(14) NOT NULL ENABLE,
SEQ_NUM     NUMBER(3,0) NOT NULL ENABLE,
MSG_PARM_VAL VARCHAR2(30) DEFAULT ' ' NOT NULL ENABLE,
VERSION     NUMBER(5,0) DEFAULT 1 NOT NULL ENABLE,
CONSTRAINT CI_TD_MSG_PARM_FK FOREIGN KEY(TD_ENTRY_ID) REFERENCES
CI_TD_ENTRY ON DELETE CASCADE
)
PARTITION BY REFERENCE (CI_TD_MSG_PARM_FK)
ENABLE ROW MOVEMENT;
```

INDEX

```
CREATE UNIQUE INDEX XT040P0 ON CI_TD_MSG_PARM ( TD_ENTRY_ID, SEQ_NUM )
TABLESPACE CM_XT039_IND
GLOBAL PARTITION BY RANGE (TD_ENTRY_ID)
 (
   PARTITION P1 VALUES LESS THAN ( '12499999999999' ),
   PARTITION P2 VALUES LESS THAN ( '24999999999999' ),
   PARTITION P3 VALUES LESS THAN ( '37499999999999' ),
   PARTITION P4 VALUES LESS THAN ( '49999999999999' ),
   PARTITION P5 VALUES LESS THAN ( '62499999999999' ),
```

```
   PARTITION P6 VALUES LESS THAN ( '74999999999999' ),
   PARTITION P7 VALUES LESS THAN ( '87499999999999' ),
   PARTITION P8 VALUES LESS THAN ( MAXVALUE )
 )
COMPRESS ADVANCED LOW;

ALTER TABLE CI_TD_MSG_PARM ADD CONSTRAINT XT040P0 PRIMARY
KEY(TD_ENTRY_ID, SEQ_NUM) USING INDEX;
```

Child Table: CI_TD_SRTKEY

```
CREATE TABLE CI_TD_SRTKEY
(
TD_ENTRY_ID CHAR(14) NOT NULL ENABLE,
SEQ_NUM    NUMBER(3,0) NOT NULL ENABLE,
KEY_VALUE  VARCHAR2(50) DEFAULT ' ' NOT NULL ENABLE,
VERSION    NUMBER(5,0) DEFAULT 1 NOT NULL ENABLE,
CONSTRAINT CI_TD_SRTKEY_FK FOREIGN KEY(TD_ENTRY_ID) REFERENCES
CI_TD_ENTRY ON DELETE CASCADE
)
PARTITION BY REFERENCE (CI_TD_SRTKEY_FK)
ENABLE ROW MOVEMENT;
```

INDEX

```
CREATE UNIQUE INDEX XT041P0 ON CI_TD_SRTKEY ( TD_ENTRY_ID, SEQ_NUM )
TABLESPACE CM_XT039_IND
GLOBAL PARTITION BY RANGE (TD_ENTRY_ID)
(
PARTITION P1 VALUES LESS THAN ( '12499999999999' ),
PARTITION P2 VALUES LESS THAN ( '24999999999999' ),
PARTITION P3 VALUES LESS THAN ( '37499999999999' ),
PARTITION P4 VALUES LESS THAN ( '49999999999999' ),
PARTITION P5 VALUES LESS THAN ( '62499999999999' ),
PARTITION P6 VALUES LESS THAN ( '74999999999999' ),
PARTITION P7 VALUES LESS THAN ( '87499999999999' ),
PARTITION P8 VALUES LESS THAN ( MAXVALUE )
)
COMPRESS ADVANCED LOW;

ALTER TABLE CI_TD_SRTKEY ADD CONSTRAINT XT041P0 PRIMARY
KEY(TD_ENTRY_ID, SEQ_NUM) USING INDEX;

CREATE INDEX XT041S1 ON CI_TD_SRTKEY ( KEY_VALUE, TD_ENTRY_ID )
TABLESPACE CM_XT039_IND COMPRESS ADVANCED LOW;
```

Maintenance Object:F1-SYNCREQIN

Parent Table: F1_SYNC_REQ_IN

```
CREATE BIGFILE TABLESPACE CM_F1T191_P2017JAN DATAFILE '+DATADG' SIZE
50M AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS
ADVANCED;
```

```
CREATE BIGFILE TABLESPACE CM_F1T191_P2017FEB DATAFILE '+DATADG' SIZE
50M AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS
ADVANCED;
CREATE BIGFILE TABLESPACE CM_F1T191_P2017MAR DATAFILE '+DATADG' SIZE
50M AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS
ADVANCED;
CREATE BIGFILE TABLESPACE CM_F1T191_P2017APR DATAFILE '+DATADG' SIZE
50M AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS
ADVANCED;
CREATE BIGFILE TABLESPACE CM_F1T191_P2017MAY DATAFILE '+DATADG' SIZE
50M AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS
ADVANCED;
CREATE BIGFILE TABLESPACE CM_F1T191_P2017JUN DATAFILE '+DATADG' SIZE
50M AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS
ADVANCED;
CREATE BIGFILE TABLESPACE CM_F1T191_P2017JUL DATAFILE '+DATADG' SIZE
50M AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS
ADVANCED;
CREATE BIGFILE TABLESPACE CM_F1T191_P2017AUG DATAFILE '+DATADG' SIZE
50M AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS
ADVANCED;
CREATE BIGFILE TABLESPACE CM_F1T191_P2017SEP DATAFILE '+DATADG' SIZE
50M AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS
ADVANCED;
CREATE BIGFILE TABLESPACE CM_F1T191_P2017OCT DATAFILE '+DATADG' SIZE
50M AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS
ADVANCED;
CREATE BIGFILE TABLESPACE CM_F1T191_P2017NOV DATAFILE '+DATADG' SIZE
50M AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS
ADVANCED;
CREATE BIGFILE TABLESPACE CM_F1T191_P2017DEC DATAFILE '+DATADG' SIZE
50M AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS
ADVANCED;
CREATE BIGFILE TABLESPACE CM_F1T191_PMAX DATAFILE '+DATADG' SIZE 50M
AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;

CREATE TABLE F1_SYNC_REQ_IN
  (
    F1_SYNC_REQ_IN_ID CHAR(14) NOT NULL ENABLE,
    BUS_OBJ_CD        CHAR(30) DEFAULT ' ' NOT NULL ENABLE,
    CRE_DTTM DATE NOT NULL ENABLE,
    BO_STATUS_CD CHAR(12) DEFAULT ' ' NOT NULL ENABLE,
    STATUS_UPD_DTTM DATE,
    MAINT_OBJ_CD  CHAR(12) DEFAULT ' ' NOT NULL ENABLE,
    NT_XID_CD     CHAR(30) DEFAULT ' ' NOT NULL ENABLE,
    EXT_PK_VALUE1 VARCHAR2(254) DEFAULT ' ' NOT NULL ENABLE,
    EXT_PK_VALUE2 VARCHAR2(254) DEFAULT ' ' NOT NULL ENABLE,
    EXT_PK_VALUE3 VARCHAR2(254) DEFAULT ' ' NOT NULL ENABLE,
    EXT_PK_VALUE4 VARCHAR2(254) DEFAULT ' ' NOT NULL ENABLE,
    EXT_PK_VALUE5 VARCHAR2(254) DEFAULT ' ' NOT NULL ENABLE,
    PK_VALUE1     VARCHAR2(254) DEFAULT ' ' NOT NULL ENABLE,
    BO_DATA_AREA CLOB,
    PRE_TRN_INIT_BO_DATA_AREA CLOB,
    PRE_TRN_FIN_BO_DATA_AREA CLOB,
    POST_TRN_BO_DATA_AREA CLOB,
    VERSION                  NUMBER(5,0) DEFAULT 1 NOT NULL ENABLE,
    EXT_REFERENCE_ID         CHAR(36) DEFAULT ' ' NOT NULL ENABLE,
    F1_INITIAL_LOAD_SYNC_FLG CHAR(4) DEFAULT ' ' NOT NULL ENABLE,
    F1_COMPOSITE_SYNC_FLG    CHAR(4) DEFAULT ' ' NOT NULL ENABLE,
    ILM_DT DATE,
    ILM_ARCH_SW CHAR(1)
   )
```

```
ENABLE ROW MOVEMENT
LOB   (BO_DATA_AREA)   STORE AS SECUREFILE (ENABLE STORAGE IN ROW
COMPRESS MEDIUM CACHE)
LOB   (PRE_TRN_INIT_BO_DATA_AREA)   STORE AS SECUREFILE (ENABLE STORAGE
IN ROW COMPRESS MEDIUM CACHE)
LOB   (PRE_TRN_FIN_BO_DATA_AREA)   STORE AS SECUREFILE (ENABLE STORAGE
IN ROW COMPRESS MEDIUM CACHE)
LOB   (POST_TRN_BO_DATA_AREA)   STORE AS SECUREFILE (ENABLE STORAGE IN
ROW COMPRESS MEDIUM CACHE)
PARTITION BY RANGE(ILM_DT)
SUBPARTITION BY RANGE(F1_SYNC_REQ_IN_ID)
SUBPARTITION TEMPLATE
(
    SUBPARTITION S01 VALUES LESS THAN ( '12499999999999' ),
    SUBPARTITION S02 VALUES LESS THAN ( '24999999999999' ),
    SUBPARTITION S03 VALUES LESS THAN ( '37499999999999' ),
    SUBPARTITION S04 VALUES LESS THAN ( '49999999999999' ),
    SUBPARTITION S05 VALUES LESS THAN ( '62499999999999' ),
    SUBPARTITION S06 VALUES LESS THAN ( '74999999999999' ),
    SUBPARTITION S07 VALUES LESS THAN ( '87499999999999' ),
    SUBPARTITION SMAX VALUES LESS THAN ( MAXVALUE )
 )
(

PARTITION "P2017JAN" VALUES LESS THAN (TO_DATE('2017-02-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
LOB(BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS
MEDIUM CACHE tablespace CM_F1T191_P2017JAN )
LOB(PRE_TRN_INIT_ BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN
ROW COMPRESS MEDIUM CACHE tablespace CM_F1T191_P2017JAN )
LOB(PRE_TRN_FIN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN
ROW COMPRESS MEDIUM CACHE tablespace CM_F1T191_P2017JAN )
LOB(POST_TRN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW
COMPRESS MEDIUM CACHE
tablespace CM_F1T191_P2017JAN )
tablespace CM_F1T191_P2017JAN,
PARTITION "P2017FEB" VALUES LESS THAN (TO_DATE('2017-03-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
LOB(BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS
MEDIUM CACHE tablespace CM_F1T191_P2017FEB )
LOB(PRE_TRN_INIT_ BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN
ROW COMPRESS MEDIUM CACHE tablespace CM_F1T191_P2017FEB )
LOB(PRE_TRN_FIN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN
ROW COMPRESS MEDIUM CACHE tablespace CM_F1T191_P2017FEB )
LOB(POST_TRN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW
COMPRESS MEDIUM CACHE tablespace CM_F1T191_P2017FEB )
tablespace CM_F1T191_P2017FEB,
PARTITION "P2017MAR" VALUES LESS THAN (TO_DATE('2017-04-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
LOB(BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS
MEDIUM CACHE tablespace CM_F1T191_P2017MAR )
LOB(PRE_TRN_INIT_ BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN
ROW COMPRESS MEDIUM CACHE tablespace CM_F1T191_P2017MAR )
LOB(PRE_TRN_FIN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN
ROW COMPRESS MEDIUM CACHE tablespace CM_F1T191_P2017MAR )
LOB(POST_TRN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW
COMPRESS MEDIUM CACHE tablespace CM_F1T191_P2017MAR )
tablespace CM_F1T191_P2017MAR,
PARTITION "P2017APR" VALUES LESS THAN (TO_DATE('2017-05-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
LOB(BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS
MEDIUM CACHE tablespace CM_F1T191_P2017APR )
```

```
LOB(PRE_TRN_INIT_ BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN
ROW COMPRESS MEDIUM CACHE tablespace CM_F1T191_P2017APR )
LOB(PRE_TRN_FIN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN
ROW COMPRESS MEDIUM CACHE tablespace CM_F1T191_P2017APR )
LOB(POST_TRN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW
COMPRESS MEDIUM CACHE
tablespace CM_F1T191_P2017APR )
tablespace CM_F1T191_P2017APR,
PARTITION "P2017MAY" VALUES LESS THAN (TO_DATE('2017-06-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
LOB(BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS
MEDIUM CACHE tablespace CM_F1T191_P2017MAY )
LOB(PRE_TRN_INIT_ BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN
ROW COMPRESS MEDIUM CACHE tablespace CM_F1T191_P2017MAY )
LOB(PRE_TRN_FIN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN
ROW COMPRESS MEDIUM CACHE tablespace CM_F1T191_P2017MAY )
LOB(POST_TRN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW
COMPRESS MEDIUM CACHE
tablespace CM_F1T191_P2017MAY )
tablespace CM_F1T191_P2017MAY,
PARTITION "P2017JUN" VALUES LESS THAN (TO_DATE('2017-07-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
LOB(BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS
MEDIUM CACHE tablespace CM_F1T191_P2017JUN )
LOB(PRE_TRN_INIT_ BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN
ROW COMPRESS MEDIUM CACHE tablespace CM_F1T191_P2017JUN )
LOB(PRE_TRN_FIN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN
ROW COMPRESS MEDIUM CACHE tablespace CM_F1T191_P2017JUN )
LOB(POST_TRN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW
COMPRESS MEDIUM CACHE tablespace CM_F1T191_P2017JUN )
tablespace CM_F1T191_P2017JUN,
PARTITION "P2017JUL" VALUES LESS THAN (TO_DATE('2017-08-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
LOB(BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS
MEDIUM CACHE tablespace CM_F1T191_P2017JUL )
LOB(PRE_TRN_INIT_ BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN
ROW COMPRESS MEDIUM CACHE tablespace CM_F1T191_P2017JUL )
LOB(PRE_TRN_FIN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN
ROW COMPRESS MEDIUM CACHE tablespace CM_F1T191_P2017JUL )
LOB(POST_TRN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW
COMPRESS MEDIUM CACHE
tablespace CM_F1T191_P2017JUL )
tablespace CM_F1T191_P2017JUL,
PARTITION "P2017AUG" VALUES LESS THAN (TO_DATE('2017-09-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
LOB(BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS
MEDIUM CACHE tablespace CM_F1T191_P2017AUG )
LOB(PRE_TRN_INIT_ BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN
ROW COMPRESS MEDIUM CACHE tablespace CM_F1T191_P2017AUG )
LOB(PRE_TRN_FIN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN
ROW COMPRESS MEDIUM CACHE tablespace CM_F1T191_P2017AUG )
LOB(POST_TRN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW
COMPRESS MEDIUM CACHE tablespace CM_F1T191_P2017AUG )
tablespace CM_F1T191_P2017AUG,
PARTITION "P2017SEP" VALUES LESS THAN (TO_DATE('2017-10-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
LOB(BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS
MEDIUM CACHE tablespace CM_F1T191_P2017SEP )
LOB(PRE_TRN_INIT_ BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN
ROW COMPRESS MEDIUM CACHE tablespace CM_F1T191_P2017SEP )
LOB(PRE_TRN_FIN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN
ROW COMPRESS MEDIUM CACHE tablespace CM_F1T191_P2017SEP )
```

```
LOB(POST_TRN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW
COMPRESS MEDIUM CACHE tablespace CM_F1T191_P2017SEP )
tablespace CM_F1T191_P2017SEP,
PARTITION "P2017OCT" VALUES LESS THAN (TO_DATE('2017-11-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
LOB(BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS
MEDIUM CACHE tablespace CM_F1T191_P2017OCT )
LOB(PRE_TRN_INIT_ BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN
ROW COMPRESS MEDIUM CACHE tablespace CM_F1T191_P2017OCT )
LOB(PRE_TRN_FIN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN
ROW COMPRESS MEDIUM CACHE tablespace CM_F1T191_P2017OCT )
LOB(POST_TRN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW
COMPRESS MEDIUM CACHE
tablespace CM_F1T191_P2017OCT )
tablespace CM_F1T191_P2017OCT,
PARTITION "P2017NOV" VALUES LESS THAN (TO_DATE('2017-12-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
LOB(BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS
MEDIUM CACHE tablespace CM_F1T191_P2017NOV )
LOB(PRE_TRN_INIT_ BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN
ROW COMPRESS MEDIUM CACHE tablespace CM_F1T191_P2017NOV )
LOB(PRE_TRN_FIN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN
ROW COMPRESS MEDIUM CACHE tablespace CM_F1T191_P2017NOV )
LOB(POST_TRN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW
COMPRESS MEDIUM CACHE tablespace CM_F1T191_P2017NOV )
tablespace CM_F1T191_P2017NOV,
PARTITION "P2017DEC" VALUES LESS THAN (TO_DATE('2018-01-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
LOB(BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS
MEDIUM CACHE tablespace CM_F1T191_P2017DEC )
LOB(PRE_TRN_INIT_ BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN
ROW COMPRESS MEDIUM CACHE tablespace CM_F1T191_P2017DEC )
LOB(PRE_TRN_FIN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN
ROW COMPRESS MEDIUM CACHE tablespace CM_F1T191_P2017DEC )
LOB(POST_TRN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW
COMPRESS MEDIUM CACHE tablespace CM_F1T191_P2017DEC )
tablespace CM_F1T191_P2017DEC,
PARTITION "PMAX" VALUES LESS THAN (MAXVALUE)
LOB(BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS
MEDIUM CACHE tablespace CM_F1T191_PMAX )
LOB(PRE_TRN_INIT_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN
ROW COMPRESS MEDIUM CACHE tablespace CM_F1T191_PMAX )
LOB(PRE_TRN_FIN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN
ROW COMPRESS MEDIUM CACHE tablespace CM_F1T191_PMAX )
LOB(POST_TRN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW
COMPRESS MEDIUM CACHE
tablespace CM_F1T191_PMAX )
tablespace CM_F1T191_PMAX
);
```

INDEX

```
CREATE BIGFILE TABLESPACE CM_F1T191_IND DATAFILE '+DATADG' SIZE 50M
AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;

CREATE UNIQUE INDEX F1T191P0 ON F1_SYNC_REQ_IN(F1_SYNC_REQ_IN_ID)
TABLESPACE CM_F1T191_IND
GLOBAL PARTITION BY RANGE (F1_SYNC_REQ_IN_ID)
(
PARTITION P1 VALUES LESS THAN ( '12499999999999' ),
PARTITION P2 VALUES LESS THAN ( '24999999999999' ),
```

```
PARTITION P3 VALUES LESS THAN ( '37499999999999' ),
PARTITION P4 VALUES LESS THAN ( '49999999999999' ),
PARTITION P5 VALUES LESS THAN ( '62499999999999' ),
PARTITION P6 VALUES LESS THAN ( '74999999999999' ),
PARTITION P7 VALUES LESS THAN ( '87499999999999' ),
PARTITION P8 VALUES LESS THAN ( MAXVALUE )
);

ALTER TABLE F1_SYNC_REQ_IN ADD CONSTRAINT F1T191P0 PRIMARY KEY
(F1_SYNC_REQ_IN_ID) USING INDEX;

CREATE UNIQUE INDEX F1T191S1 ON F1_SYNC_REQ_IN (BO_STATUS_CD,
BUS_OBJ_CD, F1_SYNC_REQ_IN_ID) TABLESPACE CM_F1T191_IND COMPRESS
ADVANCED LOW;

CREATE INDEX F1T191S2 ON
F1_SYNC_REQ_IN(MAINT_OBJ_CD,EXT_PK_VALUE1,NT_XID_CD,PK_VALUE1)
TABLESPACE CM_F1T191_IND COMPRESS ADVANCED LOW;

CREATE UNIQUE INDEX CM_ILM_F1T191S3 ON F1_SYNC_REQ_IN(ILM_DT,
ILM_ARCH_SW, F1_SYNC_REQ_IN_ID) LOCAL COMPRESS ADVANCED LOW;
```

Child Table: F1_SYNC_REQ_IN_CHAR

```
CREATE TABLE F1_SYNC_REQ_IN_CHAR
  (
    F1_SYNC_REQ_IN_ID CHAR(14) NOT NULL ENABLE,
    CHAR_TYPE_CD      CHAR(8) NOT NULL ENABLE,
    SEQ_NUM           NUMBER(3,0) NOT NULL ENABLE,
    CHAR_VAL          CHAR(16) DEFAULT ' ' NOT NULL ENABLE,
    ADHOC_CHAR_VAL    VARCHAR2(254) DEFAULT ' ' NOT NULL ENABLE,
    CHAR_VAL_FK1      VARCHAR2(50) DEFAULT ' ' NOT NULL ENABLE,
    CHAR_VAL_FK2      VARCHAR2(50) DEFAULT ' ' NOT NULL ENABLE,
    CHAR_VAL_FK3      VARCHAR2(50) DEFAULT ' ' NOT NULL ENABLE,
    CHAR_VAL_FK4      VARCHAR2(50) DEFAULT ' ' NOT NULL ENABLE,
    CHAR_VAL_FK5      VARCHAR2(50) DEFAULT ' ' NOT NULL ENABLE,
    SRCH_CHAR_VAL     VARCHAR2(50) DEFAULT ' ' NOT NULL ENABLE,
    VERSION           NUMBER(5,0) DEFAULT 1 NOT NULL ENABLE,
   CONSTRAINT F1_SYNC_REQ_IN_CHAR_FK FOREIGN KEY(F1_SYNC_REQ_IN_ID)
REFERENCES F1_SYNC_REQ_IN ON DELETE CASCADE
)
PARTITION BY REFERENCE (F1_SYNC_REQ_IN_CHAR_FK)
ENABLE ROW MOVEMENT;
```

INDEX

```
CREATE UNIQUE INDEX F1T193P0 ON F1_SYNC_REQ_IN_CHAR(F1_SYNC_REQ_IN_ID,
CHAR_TYPE_CD, SEQ_NUM) TABLESPACE CM_F1T191_IND
GLOBAL PARTITION BY RANGE (F1_SYNC_REQ_IN_ID)
(
PARTITION P1 VALUES LESS THAN ( '12499999999999' ),
PARTITION P2 VALUES LESS THAN ( '24999999999999' ),
PARTITION P3 VALUES LESS THAN ( '37499999999999' ),
PARTITION P4 VALUES LESS THAN ( '49999999999999' ),
PARTITION P5 VALUES LESS THAN ( '62499999999999' ),
PARTITION P6 VALUES LESS THAN ( '74999999999999' ),
PARTITION P7 VALUES LESS THAN ( '87499999999999' ),
PARTITION P8 VALUES LESS THAN ( MAXVALUE )
)
```

```
COMPRESS ADVANCED LOW;

ALTER TABLE F1_SYNC_REQ_IN_CHAR ADD CONSTRAINT F1T193P0 PRIMARY KEY
(F1_SYNC_REQ_IN_ID, CHAR_TYPE_CD, SEQ_NUM) USING INDEX;

CREATE INDEX F1T193S1 ON F1_SYNC_REQ_IN_CHAR(SRCH_CHAR_VAL) TABLESPACE
CM_F1T191_IND ;
```

Child Table: F1_SYNC_REQ_IN_EXCP

```
CREATE TABLE F1_SYNC_REQ_IN_EXCP
  (
    F1_SYNC_REQ_IN_ID CHAR(14) NOT NULL ENABLE,
    SEQNO             NUMBER(5,0) NOT NULL ENABLE,
    MESSAGE_CAT_NBR   NUMBER(5,0) DEFAULT 0 NOT NULL ENABLE,
    MESSAGE_NBR       NUMBER(5,0) DEFAULT 0 NOT NULL ENABLE,
    VERSION           NUMBER(5,0) DEFAULT 1 NOT NULL ENABLE,
   CONSTRAINT F1_SYNC_REQ_IN_EXCP_FK FOREIGN KEY(F1_SYNC_REQ_IN_ID)
REFERENCES F1_SYNC_REQ_IN ON DELETE CASCADE
)
PARTITION BY REFERENCE (F1_SYNC_REQ_IN_EXCP_FK)
ENABLE ROW MOVEMENT;
```

INDEX

```
CREATE UNIQUE INDEX F1T197P0 ON
F1_SYNC_REQ_IN_EXCP(F1_SYNC_REQ_IN_ID,SEQNO) TABLESPACE CM_F1T191_IND
GLOBAL PARTITION BY RANGE (F1_SYNC_REQ_IN_ID)
(
PARTITION P1 VALUES LESS THAN ( '12499999999999' ),
PARTITION P2 VALUES LESS THAN ( '24999999999999' ),
PARTITION P3 VALUES LESS THAN ( '37499999999999' ),
PARTITION P4 VALUES LESS THAN ( '49999999999999' ),
PARTITION P5 VALUES LESS THAN ( '62499999999999' ),
PARTITION P6 VALUES LESS THAN ( '74999999999999' ),
PARTITION P7 VALUES LESS THAN ( '87499999999999' ),
PARTITION P8 VALUES LESS THAN ( MAXVALUE )
)
COMPRESS ADVANCED LOW;

ALTER TABLE F1_SYNC_REQ_IN_EXCP ADD CONSTRAINT F1T197P0 PRIMARY KEY
(F1_SYNC_REQ_IN_ID,SEQNO) USING INDEX;
```

Child Table: F1_SYNC_REQ_IN_EXCP_PARM

```
CREATE TABLE F1_SYNC_REQ_IN_EXCP_PARM
  (
    F1_SYNC_REQ_IN_ID CHAR(14) NOT NULL ENABLE,
    SEQNO             NUMBER(5,0) NOT NULL ENABLE,
    PARM_SEQ          NUMBER(3,0) NOT NULL ENABLE,
    MSG_PARM_VAL      VARCHAR2(30) DEFAULT ' ' NOT NULL ENABLE,
    MSG_PARM_TYP_FLG  CHAR(4) DEFAULT ' ' NOT NULL ENABLE,
    VERSION           NUMBER(5,0) DEFAULT 1 NOT NULL ENABLE,
   CONSTRAINT F1_SYNC_REQ_IN_EXCP_PARM_FK FOREIGN
KEY(F1_SYNC_REQ_IN_ID) REFERENCES F1_SYNC_REQ_IN ON DELETE CASCADE
)
PARTITION BY REFERENCE (F1_SYNC_REQ_IN_EXCP_PARM_FK)
ENABLE ROW MOVEMENT;
```

INDEX

```
CREATE UNIQUE INDEX F1T198P0 ON
F1_SYNC_REQ_IN_EXCP_PARM(F1_SYNC_REQ_IN_ID,SEQNO,PARM_SEQ) TABLESPACE
CM_F1T191_IND
GLOBAL PARTITION BY RANGE (F1_SYNC_REQ_IN_ID)
(
PARTITION P1 VALUES LESS THAN ( '12499999999999' ),
PARTITION P2 VALUES LESS THAN ( '24999999999999' ),
PARTITION P3 VALUES LESS THAN ( '37499999999999' ),
PARTITION P4 VALUES LESS THAN ( '49999999999999' ),
PARTITION P5 VALUES LESS THAN ( '62499999999999' ),
PARTITION P6 VALUES LESS THAN ( '74999999999999' ),
PARTITION P7 VALUES LESS THAN ( '87499999999999' ),
PARTITION P8 VALUES LESS THAN ( MAXVALUE )
)
COMPRESS ADVANCED LOW;

ALTER TABLE F1_SYNC_REQ_IN_EXCP_PARM ADD CONSTRAINT F1T198P0 PRIMARY
KEY (F1_SYNC_REQ_IN_ID,SEQNO,PARM_SEQ) USING INDEX;
```

Child Table: F1_SYNC_REQ_IN_LOG

```
CREATE TABLE F1_SYNC_REQ_IN_LOG
  (
    F1_SYNC_REQ_IN_ID  CHAR(14) NOT NULL ENABLE,
    SEQNO              NUMBER(5,0) NOT NULL ENABLE,
    LOG_ENTRY_TYPE_FLG CHAR(4) DEFAULT ' ' NOT NULL ENABLE,
    LOG_DTTM DATE NOT NULL ENABLE,
    BO_STATUS_CD    CHAR(12) DEFAULT ' ' NOT NULL ENABLE,
    MESSAGE_CAT_NBR NUMBER(5,0) DEFAULT 0 NOT NULL ENABLE,
    MESSAGE_NBR     NUMBER(5,0) DEFAULT 0 NOT NULL ENABLE,
    CHAR_TYPE_CD    CHAR(8) DEFAULT ' ' NOT NULL ENABLE,
    CHAR_VAL        CHAR(16) DEFAULT ' ' NOT NULL ENABLE,
    ADHOC_CHAR_VAL  VARCHAR2(254) DEFAULT ' ' NOT NULL ENABLE,
    CHAR_VAL_FK1    VARCHAR2(50) DEFAULT ' ' NOT NULL ENABLE,
    CHAR_VAL_FK2    VARCHAR2(50) DEFAULT ' ' NOT NULL ENABLE,
    CHAR_VAL_FK3    VARCHAR2(50) DEFAULT ' ' NOT NULL ENABLE,
    CHAR_VAL_FK4    VARCHAR2(50) DEFAULT ' ' NOT NULL ENABLE,
    CHAR_VAL_FK5    VARCHAR2(50) DEFAULT ' ' NOT NULL ENABLE,
    DESCRLONG       VARCHAR2(4000) DEFAULT ' ' NOT NULL ENABLE,
    USER_ID         CHAR(8) DEFAULT ' ' NOT NULL ENABLE,
    VERSION         NUMBER(5,0) DEFAULT 1 NOT NULL ENABLE,
   CONSTRAINT F1_SYNC_REQ_IN_LOG_FK FOREIGN KEY(F1_SYNC_REQ_IN_ID)
REFERENCES F1_SYNC_REQ_IN ON DELETE CASCADE
)
PARTITION BY REFERENCE (F1_SYNC_REQ_IN_LOG_FK)
ENABLE ROW MOVEMENT;
```

INDEX

```
CREATE UNIQUE INDEX F1T194P0 ON
F1_SYNC_REQ_IN_LOG(F1_SYNC_REQ_IN_ID,SEQNO) TABLESPACE CM_F1T191_IND
GLOBAL PARTITION BY RANGE (F1_SYNC_REQ_IN_ID)
(
PARTITION P1 VALUES LESS THAN ( '12499999999999' ),
PARTITION P2 VALUES LESS THAN ( '24999999999999' ),
PARTITION P3 VALUES LESS THAN ( '37499999999999' ),
PARTITION P4 VALUES LESS THAN ( '49999999999999' ),
PARTITION P5 VALUES LESS THAN ( '62499999999999' ),
PARTITION P6 VALUES LESS THAN ( '74999999999999' ),
```

```
PARTITION P7 VALUES LESS THAN ( '87499999999999' ),
PARTITION P8 VALUES LESS THAN ( MAXVALUE )
)
COMPRESS ADVANCED LOW;

ALTER TABLE F1_SYNC_REQ_IN_LOG ADD CONSTRAINT F1T194P0 PRIMARY KEY
(F1_SYNC_REQ_IN_ID,SEQNO) USING INDEX;

CREATE INDEX F1T194S1 ON F1_SYNC_REQ_IN_LOG(CHAR_TYPE_CD,CHAR_VAL_FK1)
TABLESPACE CM_F1T191_IND COMPRESS ADVANCED LOW;

CREATE INDEX F1T194S2 ON F1_SYNC_REQ_IN_LOG(CHAR_TYPE_CD,CHAR_VAL)
TABLESPACE CM_F1T191_IND COMPRESS ADVANCED LOW;
```

Child Table: F1_SYNC_REQ_IN_LOG_PARM

```
CREATE TABLE F1_SYNC_REQ_IN_LOG_PARM
  (
    F1_SYNC_REQ_IN_ID CHAR(14) NOT NULL ENABLE,
    SEQNO             NUMBER(5,0) NOT NULL ENABLE,
    PARM_SEQ          NUMBER(3,0) NOT NULL ENABLE,
    MSG_PARM_VAL      VARCHAR2(30) DEFAULT ' ' NOT NULL ENABLE,
    MSG_PARM_TYP_FLG  CHAR(4) DEFAULT ' ' NOT NULL ENABLE,
    VERSION           NUMBER(5,0) DEFAULT 1 NOT NULL ENABLE,
    CONSTRAINT F1_SYNC_REQ_IN_LOG_PARM_FK FOREIGN
KEY(F1_SYNC_REQ_IN_ID) REFERENCES F1_SYNC_REQ_IN ON DELETE CASCADE
)
PARTITION BY REFERENCE (F1_SYNC_REQ_IN_LOG_PARM_FK)
ENABLE ROW MOVEMENT;
```

INDEX

```
CREATE UNIQUE INDEX F1T195P0 ON
F1_SYNC_REQ_IN_LOG_PARM(F1_SYNC_REQ_IN_ID,SEQNO,PARM_SEQ) TABLESPACE
CM_F1T191_IND
GLOBAL PARTITION BY RANGE (F1_SYNC_REQ_IN_ID)
(
PARTITION P1 VALUES LESS THAN ( '12499999999999' ),
PARTITION P2 VALUES LESS THAN ( '24999999999999' ),
PARTITION P3 VALUES LESS THAN ( '37499999999999' ),
PARTITION P4 VALUES LESS THAN ( '49999999999999' ),
PARTITION P5 VALUES LESS THAN ( '62499999999999' ),
PARTITION P6 VALUES LESS THAN ( '74999999999999' ),
PARTITION P7 VALUES LESS THAN ( '87499999999999' ),
PARTITION P8 VALUES LESS THAN ( MAXVALUE )
)
COMPRESS ADVANCED LOW;

ALTER TABLE F1_SYNC_REQ_IN_LOG_PARM ADD CONSTRAINT F1T195P0 PRIMARY
KEY (F1_SYNC_REQ_IN_ID,SEQNO,PARM_SEQ) USING INDEX;
```

Child Table: F1_SYNC_REQ_IN_REL_OBJ

```
CREATE TABLE F1_SYNC_REQ_IN_REL_OBJ
  (
    F1_SYNC_REQ_IN_ID CHAR(14) NOT NULL ENABLE,
    MAINT_OBJ_CD      CHAR(12) DEFAULT ' ' NOT NULL ENABLE,
    REL_OBJ_TYPE_FLG  CHAR(4) DEFAULT ' ' NOT NULL ENABLE,
    PK_VALUE1         VARCHAR2(254) DEFAULT ' ' NOT NULL ENABLE,
```

```
    PK_VALUE2         VARCHAR2(254) DEFAULT ' ' NOT NULL ENABLE,
    PK_VALUE3         VARCHAR2(254) DEFAULT ' ' NOT NULL ENABLE,
    PK_VALUE4         VARCHAR2(254) DEFAULT ' ' NOT NULL ENABLE,
    PK_VALUE5         VARCHAR2(254) DEFAULT ' ' NOT NULL ENABLE,
    VERSION           NUMBER(5,0) DEFAULT 1 NOT NULL ENABLE,
   CONSTRAINT F1_SYNC_REQ_IN_REL_OBJ_FK FOREIGN KEY(F1_SYNC_REQ_IN_ID)
REFERENCES F1_SYNC_REQ_IN ON DELETE CASCADE
)
PARTITION BY REFERENCE (F1_SYNC_REQ_IN_REL_OBJ_FK)
ENABLE ROW MOVEMENT;
```

INDEX

```
CREATE UNIQUE INDEX F1T192P0 ON
F1_SYNC_REQ_IN_REL_OBJ(F1_SYNC_REQ_IN_ID, MAINT_OBJ_CD,
REL_OBJ_TYPE_FLG) TABLESPACE CM_F1T191_IND
GLOBAL PARTITION BY RANGE (F1_SYNC_REQ_IN_ID)
(
PARTITION P1 VALUES LESS THAN ( '12499999999999' ),
PARTITION P2 VALUES LESS THAN ( '24999999999999' ),
PARTITION P3 VALUES LESS THAN ( '37499999999999' ),
PARTITION P4 VALUES LESS THAN ( '49999999999999' ),
PARTITION P5 VALUES LESS THAN ( '62499999999999' ),
PARTITION P6 VALUES LESS THAN ( '74999999999999' ),
PARTITION P7 VALUES LESS THAN ( '87499999999999' ),
PARTITION P8 VALUES LESS THAN ( MAXVALUE )
)
COMPRESS ADVANCED LOW;

ALTER TABLE F1_SYNC_REQ_IN_REL_OBJ ADD CONSTRAINT F1T192P0 PRIMARY KEY
(F1_SYNC_REQ_IN_ID, MAINT_OBJ_CD, REL_OBJ_TYPE_FLG) USING INDEX;

CREATE INDEX F1T192S1 ON F1_SYNC_REQ_IN_REL_OBJ(PK_VALUE1) TABLESPACE
CM_F1T191_IND;
```

Appendix B

Sample SQL For Enabling ILM in CCB (Existing Installation)

This section provides additional details related to supporting ILM in an existing installation. It includes the sample syntax for each step using the To Do Entry maintenance object as an example. Other maintenance object's implementations can follow a similar pattern.

1. Rename existing table CI_TD_ENTRY and primary key index as a backup. It is suggested to use an ILM_ prefix. The following are sample statements:

```
ALTER TABLE CI_TD_ENTRY RENAME TO ILM_TD_ENTRY;
ALTER INDEX XT039P0 RENAME TO ILM_XT039P0;
```

2. Generate DDL for the secondary index.

```
set heading off;
set echo off;
Set pages 999;
set long 90000;

spool ddl_list.sql
select dbms_metadata.get_ddl('INDEX','XT039S2','CISADM') from dual;
select dbms_metadata.get_ddl('INDEX','XT039S3','CISADM') from dual;
select dbms_metadata.get_ddl('INDEX','XT039S4','CISADM') from dual;
select dbms_metadata.get_ddl('INDEX','XT039S5','CISADM') from dual;
select dbms_metadata.get_ddl('INDEX','XT039S6','CISADM') from dual;
select dbms_metadata.get_ddl('INDEX','XT039S7','CISADM') from dual;
select dbms_metadata.get_ddl('INDEX','XT039S8','CISADM') from dual;
spool off;
```

3. Drop secondary indexes.

```
DROP INDEX CISADM.XT039S2;
DROP INDEX CISADM.XT039S3;
DROP INDEX CISADM.XT039S4;
DROP INDEX CISADM.XT039S5;
DROP INDEX CISADM.XT039S6;
DROP INDEX CISADM.XT039S7;
DROP INDEX CISADM.XT039S8;
```

4. Create Partitioned Table.

   In the following example ILM_DT value is inserted from column CRE_DTTM. The degree setting of 'parallel' in the DDL can be adjusted according to the table's data, its means and its size.

```
CREATE TABLE CI_TD_ENTRY (
TD_ENTRY_ID     CHAR(14) NOT NULL ENABLE,
BATCH_CD        CHAR(8) DEFAULT ' ' NOT NULL ENABLE,
BATCH_NBR       NUMBER(10,0) DEFAULT 0 NOT NULL ENABLE,
MESSAGE_CAT_NBR NUMBER(5,0) DEFAULT 0 NOT NULL ENABLE,
MESSAGE_NBR     NUMBER(5,0) DEFAULT 0 NOT NULL ENABLE,
```

```
ASSIGNED_TO     CHAR(8) DEFAULT ' ' NOT NULL ENABLE,
TD_TYPE_CD      CHAR(8) DEFAULT ' ' NOT NULL ENABLE,
ROLE_ID         CHAR(10) DEFAULT ' ' NOT NULL ENABLE,
ENTRY_STATUS_FLG CHAR(2) DEFAULT ' ' NOT NULL ENABLE,
VERSION         NUMBER(5,0) DEFAULT 1 NOT NULL ENABLE,
CRE_DTTM DATE,
ASSIGNED_DTTM DATE,
COMPLETE_DTTM DATE,
COMPLETE_USER_ID CHAR(8) DEFAULT ' ' NOT NULL ENABLE,
COMMENTS        VARCHAR2(254) DEFAULT ' ' NOT NULL ENABLE,
ASSIGNED_USER_ID CHAR(8) DEFAULT ' ' NOT NULL ENABLE,
TD_PRIORITY_FLG CHAR(4) DEFAULT ' ' NOT NULL ENABLE,
ILM_DT DATE,
ILM_ARCH_SW CHAR(1)
) NOLOGGING PARALLEL
ENABLE ROW MOVEMENT
PARTITION BY RANGE (ILM_DT)
SUBPARTITION BY RANGE (TD_ENTRY_ID) SUBPARTITION TEMPLATE
(
SUBPARTITION S01 VALUES LESS THAN ( '12499999999999' ),
SUBPARTITION S02 VALUES LESS THAN ( '24999999999999' ),
SUBPARTITION S03 VALUES LESS THAN ( '37499999999999' ),
SUBPARTITION S04 VALUES LESS THAN ( '49999999999999' ),
SUBPARTITION S05 VALUES LESS THAN ( '62499999999999' ),
SUBPARTITION S06 VALUES LESS THAN ( '74999999999999' ),
SUBPARTITION S07 VALUES LESS THAN ( '87499999999999' ),
SUBPARTITION SMAX VALUES LESS THAN ( MAXVALUE )
)
(
PARTITION "P2017JAN" VALUES LESS THAN (TO_DATE('2017-02-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT039_P2017JAN,
PARTITION "P2017FEB" VALUES LESS THAN (TO_DATE('2017-03-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT039_P2017FEB,
PARTITION "P2017MAR" VALUES LESS THAN (TO_DATE('2017-04-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT039_P2017MAR,
PARTITION "P2017APR" VALUES LESS THAN (TO_DATE('2017-05-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT039_P2017APR,
PARTITION "P2017MAY" VALUES LESS THAN (TO_DATE('2017-06-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT039_P2017MAY,
PARTITION "P2017JUN" VALUES LESS THAN (TO_DATE('2017-07-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT039_P2017JUN,
PARTITION "P2017JUL" VALUES LESS THAN (TO_DATE('2017-08-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT039_P2017JUL,
PARTITION "P2017AUG" VALUES LESS THAN (TO_DATE('2017-09-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT039_P2017AUG,
PARTITION "P2017SEP" VALUES LESS THAN (TO_DATE('2017-10-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT039_P2017SEP,
PARTITION "P2017OCT" VALUES LESS THAN (TO_DATE('2017-11-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT039_P2017OCT,
PARTITION "P2017NOV" VALUES LESS THAN (TO_DATE('2017-12-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT039_P2017NOV,
PARTITION "P2017DEC" VALUES LESS THAN (TO_DATE('2018-01-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT039_P2017DEC,
PARTITION "PMAX" VALUES LESS THAN (MAXVALUE)
tablespace CM_XT039_PMAX
)as select /* PARALLEL */
)as select /* PARALLEL */
TD_ENTRY_ID,
BATCH_CD,
BATCH_NBR,
MESSAGE_CAT_NBR,
MESSAGE_NBR,
ASSIGNED_TO,
TD_TYPE_CD,
ROLE_ID,
ENTRY_STATUS_FLG,
VERSION,
CRE_DTTM,
ASSIGNED_DTTM,
COMPLETE_DTTM,
COMPLETE_USER_ID,
COMMENTS,
ASSIGNED_USER_ID,
```

```
TD_PRIORITY_FLG,
CRE_DTTM as ILM_DT,
ILM_ARCH_SW
from ILM_TD_ENTRY
/
```

5. Enable logging option for table CI_TD_ENTRY.

```
ALTER TABLE CI_TD_ENTRY NOPARALLEL LOGGING;
```

6. Create Primary Index for Parent table CI_TD_ENTRY.

```
CREATE BIGFILE TABLESPACE CM_XT039_IND DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;

CREATE UNIQUE INDEX XT039P0 ON CI_TD_ENTRY NOLOGGING PARALLEL (
TD_ENTRY_ID
)
GLOBAL PARTITION BY RANGE (TD_ENTRY_ID) (
PARTITION P1 VALUES LESS THAN ( '12499999999999' ),
PARTITION P2 VALUES LESS THAN ( '24999999999999' ),
PARTITION P3 VALUES LESS THAN ( '37499999999999' ),
PARTITION P4 VALUES LESS THAN ( '49999999999999' ),
PARTITION P5 VALUES LESS THAN ( '62499999999999' ),
PARTITION P6 VALUES LESS THAN ( '74999999999999' ),
PARTITION P7 VALUES LESS THAN ( '87499999999999' ),
PARTITION P8 VALUES LESS THAN ( MAXVALUE )
) TABLESPACE CM_XT039_IND
/

ALTER INDEX XT039P0 LOGGING NOPARALLEL;
```

7. Add Primary Key for Parent table CI_TD_ENTRY.

```
ALTER TABLE CI_TD_ENTRY ADD CONSTRAINT XT039P0 PRIMARY KEY(TD_ENTRY_ID) USING INDEX
/
```

8. Create Secondary Indexes for Parent table CI_TD_ENTRY.

```
CREATE UNIQUE INDEX CM_ILM_XT039S8 ON CI_TD_ENTRY ( ILM_DT, ILM_ARCH_SW, TD_ENTRY_ID )
LOCAL COMPRESS ADVANCED LOW
/

CREATE UNIQUE INDEX XT039S2 ON CI_TD_ENTRY ( ASSIGNED_TO, TD_ENTRY_ID ) TABLESPACE
CM_XT039_IND COMPRESS ADVANCED LOW
/

CREATE INDEX XT039S3 ON CI_TD_ENTRY ( ENTRY_STATUS_FLG, ASSIGNED_TO ) TABLESPACE
CM_XT039_IND COMPRESS ADVANCED LOW
/

CREATE INDEX XT039S4 ON CI_TD_ENTRY ( ROLE_ID, TD_TYPE_CD, ENTRY_STATUS_FLG,
TD_PRIORITY_FLG, ASSIGNED_TO, CRE_DTTM ) TABLESPACE CM_XT039_IND COMPRESS ADVANCED LOW
/

CREATE INDEX XT039S5 ON CI_TD_ENTRY ( BATCH_CD, BATCH_NBR, ENTRY_STATUS_FLG )
TABLESPACE CM_XT039_IND COMPRESS ADVANCED LOW
/

CREATE UNIQUE INDEX XT039S6 ON CI_TD_ENTRY ( TD_ENTRY_ID, ASSIGNED_TO,
ENTRY_STATUS_FLG )TABLESPACE CM_XT039_IND COMPRESS ADVANCED LOW
/

CREATE UNIQUE INDEX XT039S7 ON CI_TD_ENTRY ( COMPLETE_USER_ID, COMPLETE_DTTM,
TD_ENTRY_ID ) TABLESPACE CM_XT039_IND COMPRESS ADVANCED LOW
/
```

9. After verification of the ILM based tables, user can drop the backup tables "ILM" renamed table.
10. Create all child Tables, Primary Key, Primary Indexes and Secondary Indexes as shown below.

Repeat the following steps for all child tables.

Create Child Table CI_TD_DRLKEY

```
CREATE TABLE CI_TD_DRLKEY
(
TD_ENTRY_ID NOT NULL ENABLE,
```

```
SEQ_NUM    NOT NULL ENABLE,
KEY_VALUE  DEFAULT ' ' NOT NULL ENABLE,
VERSION    DEFAULT 1 NOT NULL ENABLE,
CONSTRAINT CI_TD_DRLKEY_FK FOREIGN KEY(TD_ENTRY_ID) REFERENCES CI_TD_ENTRY ON DELETE
CASCADE
)
PARTITION BY REFERENCE (CI_TD_DRLKEY_FK)
ENABLE ROW MOVEMENT
AS SELECT /*+ PARALLEL */ * FROM ILM_CI_TD_DRLKEY;
```

Create Index

```
CREATE UNIQUE INDEX XT037P0 ON CI_TD_DRLKEY ( TD_ENTRY_ID, SEQ_NUM ) TABLESPACE
CM_XT039_IND NOLOGGING PARALLEL
GLOBAL PARTITION BY RANGE (TD_ENTRY_ID)
(
PARTITION P1 VALUES LESS THAN ( '124999999999' ),
PARTITION P2 VALUES LESS THAN ( '249999999999' ),
PARTITION P3 VALUES LESS THAN ( '374999999999' ),
PARTITION P4 VALUES LESS THAN ( '499999999999' ),
PARTITION P5 VALUES LESS THAN ( '624999999999' ),
PARTITION P6 VALUES LESS THAN ( '749999999999' ),
PARTITION P7 VALUES LESS THAN ( '874999999999' ),
PARTITION P8 VALUES LESS THAN ( MAXVALUE )
)
COMPRESS ADVANCED LOW;

ALTER INDEX XT037P0 LOGGING NOPARALLEL;

ALTER TABLE CI_TD_DRLKEY ADD CONSTRAINT XT037P0 PRIMARY KEY(TD_ENTRY_ID, SEQ_NUM) USING
INDEX;

CREATE INDEX XT037S1 ON CI_TD_DRLKEY ( KEY_VALUE, TD_ENTRY_ID ) TABLESPACE CM_XT039_IND
COMPRESS ADVANCED LOW;
```

Appendix C

Sample SQL for Periodic Maintenance

This appendix provides additional details related to creating new partitions over time as well as archiving and restoring partitions. The To Do Entry and Inbound Sync Request maintenance objects are used as examples. This section contains the following steps:

- Add Partition
- Archive Partition
- Restore Partition

Add Partition

1. Create separate tablespace for new partition

```
CREATE BIGFILE TABLESPACE CM_XT039_P2016JAN DATAFILE '+DATA' SIZE
50M AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS
ADVANCED;
```

2. Add partition using split operation on MAXVALUE Partition

```
ALTER TABLE CISADM.CI_TD_ENTRY SPLIT PARTITION PMAX AT
(TO_DATE('2016-02-01 00:00:01','SYYYY-MM-DD HH24:MI:SS'))
INTO
(
PARTITION P2016JAN TABLESPACE CM_XT039_P2016JAN, PARTITION PMAX
)
UPDATE INDEXES;
```

In case table contains LOBS like F1_SYNC_REQ_IN, there will be additional statement in split partition DDL indicating tablespace on which LOB should go.

```
ALTER TABLE CISADM.F1_SYNC_REQ_IN SPLIT PARTITION PMAX AT
(TO_DATE('2016-02-01 00:00:01','SYYYY-MM-DD HH24:MI:SS'))
 INTO
 (
 PARTITION P2016JAN TABLESPACE CM_F1T191_P2016JAN
 LOB(BO_DATA_AREA, POST_TRN_BO_DATA_AREA, PRE_TRN_FIN_BO_DATA_AREA,
PRE_TRN_INIT_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN
ROW COMPRESS MEDIUM CACHE TABLESPACE CM_F1T191_P2016JAN )
,
 PARTITION PMAX
 )
 UPDATE INDEXES;
```

3. Enable advanced compression after SPLIT partition as it will disable the compression.

```
ALTER TABLE CISADM.CI_TD_SRTKEY ROW STORE COMPRESS ADVANCED;
ALTER TABLE CISADM.CI_TD_MSG_PARM ROW STORE COMPRESS ADVANCED;
ALTER TABLE CISADM.CI_TD_DRLKEY ROW STORE COMPRESS ADVANCED;
ALTER TABLE CISADM.CI_TD_ENTRY_CHA ROW STORE COMPRESS ADVANCED;
ALTER TABLE CISADM.CI_TD_LOG ROW STORE COMPRESS ADVANCED;
```

Archive Partition

1. Make the tablespace to be archived READ ONLY.

```
ALTER TABLESPACE CM_XT039_P2017JAN READ ONLY;
```

2. Check the feasibility of archive using ILM_ARCH_SW = 'N'.

```
Select count(1) from CISADM.CI_TD_ENTRY PARTITION P2017JAN where
ILM_ARCH_SW = 'N';
```

   - If Yes (count of records of above query is ZERO), then proceed for further steps.
   - If No (count of records of above query is Non ZERO), then make the tablespace back to READ WRITE MODE as Archive is not Feasible at the time.

```
ALTER TABLESPACE CM_XT039_P2017JAN READ WRITE;
```

3. Create separate archive tablespace for partition need to be archived.

```
CREATE BIGFILE TABLESPACE CM_XT039_P2017JAN_ARC DATAFILE  '+DATA'
SIZE 50M AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS
ADVANCED;
```

4. Create staging tables and load data for all child tables for the MO first.

   a. CI_TD_ENTRY_CHA

```
CREATE TABLE CM_XT701_P2017JAN_ARC PARALLEL NOLOGGING
TABLESPACE CM_XT039_P2017JAN_ARC
AS
(
SELECT /*+ PARALLEL */ * FROM CISADM.CI_TD_ENTRY_CHA PARTITION
(P2017JAN_S01)
UNION ALL
SELECT /*+ PARALLEL */ * FROM CI_TD_ENTRY_CHA PARTITION
(P2017JAN_S02)
UNION ALL
.
.
.
.
UNION ALL
SELECT /*+ PARALLEL */ * FROM CI_TD_ENTRY_CHA PARTITION
(P2017JAN_S08)
);
ALTER TABLE CM_XT701_P2017JAN_ARC NOPARALLEL LOGGING;
```

   b. **CI_TD_MSG_PARM**

```
CREATE TABLE CM_XT04_P2017JAN_ARC PARALLEL NOLOGGING TABLESPACE
CM_XT039_P2017JAN_ARC
AS
(
SELECT /*+ PARALLEL */ * FROM CISADM.CI_TD_MSG_PARM PARTITION
(P2017JAN_S01)
UNION ALL
SELECT /*+ PARALLEL */ * FROM CI_TD_MSG_PARM  PARTITION
(P2017JAN_S02)
UNION ALL
.
.
.
.
UNION ALL
SELECT /*+ PARALLEL */ * FROM CI_TD_MSG_PARM  PARTITION
(P2017JAN_S08)
);
ALTER TABLE CM_XT04_P2017JAN_ARC NOPARALLEL LOGGING;
```

   c. CI_TD_LOG

```
CREATE TABLE CM_XT721_P2017JAN_ARC PARALLEL NOLOGGING
TABLESPACE CM_XT039_P2017JAN_ARC
AS
(
SELECT /*+ PARALLEL */ * FROM CISADM.CI_TD_LOG PARTITION
(P2017JAN_S01)
UNION ALL
```

```
SELECT /*+ PARALLEL */ * FROM CI_TD_LOG PARTITION (P2017JAN_S02)
UNION ALL
.
.
.
.
UNION ALL
SELECT /*+ PARALLEL */ * FROM CI_TD_LOG PARTITION (P2017JAN_S08)
);
ALTER TABLE CM_XT721_P2017JAN_ARC NOPARALLEL LOGGING;
```

d. CI_TD_SRTKEY

```
CREATE TABLE CM_XT041_P2017JAN_ARC PARALLEL NOLOGGING
TABLESPACE CM_XT039_P2017JAN_ARC
AS
(
SELECT /*+ PARALLEL */ * FROM CISADM.CI_TD_SRTKEY PARTITION
(P2017JAN_S01)
UNION ALL
SELECT /*+ PARALLEL */ * FROM CI_TD_SRTKEY PARTITION
(P2017JAN_S02)
UNION ALL
.
.
.
.
UNION ALL
SELECT /*+ PARALLEL */ * FROM CI_TD_SRTKEY PARTITION
(P2017JAN_S08)
);
ALTER TABLE CM_XT041_P2017JAN_ARC NOPARALLEL LOGGING;
```

e. CI_TD_DRLKEY

```
CREATE TABLE CM_XT037_P2017JAN_ARC PARALLEL NOLOGGING
TABLESPACE CM_XT039_P2017JAN_ARC
AS
(
SELECT /*+ PARALLEL */ * FROM CISADM.CI_TD_DRLKEY PARTITION
(P2017JAN_S01)
UNION ALL
SELECT /*+ PARALLEL */ * FROM CISADM.CI_TD_DRLKEY PARTITION
(P2017JAN_S02)
UNION ALL
.
.
.
.
UNION ALL
SELECT /*+ PARALLEL */ * FROM CISADM.CI_TD_DRLKEY PARTITION
(P2017JAN_S08)
);
ALTER TABLE CM_XT037_P2017JAN_ARC NOPARALLEL LOGGING;
```

5. Create staging table and load data for parent table.

```
CREATE TABLE CM_XT039_P2017JAN_ARC NOLOGGING PARALLEL
TABLESPACE CM_XT039_P2017JAN_ARC AS
SELECT /*+ PARALLEL */ * FROM CISADM.CI_TD_ENTRY PARTITION
(P2017JAN);

ALTER TABLE CM_XT039_P2017JAN_ARC NOPARALLEL LOGGING;
```

6. Export tablespace using TRANSPORT_TABLESPACES method.

```
ALTER TABLESPACE CM_XT039_P2017JAN_ARC READ ONLY;

expdp system/manager DIRECTORY=DUMP_DIR DUMPFILE=
CM_XT039_P2017JAN_ARC.DMP TRANSPORT_TABLESPACES =
CM_XT039_P2017JAN_ARC LOGFILE=EXP_ CM_XT039_P2017JAN_ARC.LOG
TRANSPORT_FULL_CHECK=Y
```

Ensure tablespace datafile required for further import should be preserved.

```
<<Transport THE FILE to LOCAL DB DIRECTORY DUMP_DIR like
connected to asmcmd and copied the file from cp
cm_xt039_p201701_tbs_ar.553.913864937  /tugbu_perf_02/BACKUPS/
test_verification/ >>
```

7. Drop the partition, partition tablespace and archive tablespace (as it is already exported).

```
ALTER TABLE CISADM.CI_TD_ENTRY  DROP PARTITION P2017JAN UPDATE
INDEXES;
DROP TABLESPACE CM_XT039_P2017JAN INCLUDING CONTENTS AND
DATAFILES;
DROP TABLESPACE CM_XT039_P2017JAN_ARC INCLUDING CONTENTS AND
DATAFILES;
```

Restore Partition

1. Create separate tablespace to restore the partition.

```
CREATE BIGFILE TABLESPACE CM_XT039_P2017JAN DATAFILE '+DATA'
SIZE 50M AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE
COMPRESS ADVANCED;
```

2. Add partition using split operation on next greater value partition

```
ALTER TABLE CISADM.CI_TD_ENTRY SPLIT PARTITION P2017FEB AT
(TO_DATE('2017-02-01 00:00:01','SYYYY-MM-DD HH24:MI:SS'))
 INTO
 (
 PARTITION P2017JAN TABLESPACE CM_XT039_P2017JAN , PARTITION
P2017FEB
 )
 UPDATE INDEXES;
```

In case table contains LOBS like F1_SYNC_REQ_IN, there will be additional statement in split partition DDL indicating tablespace on which LOB should go.

```
ALTER TABLE CISADM.F1_SYNC_REQ_IN SPLIT PARTITION P2017FEB AT
(TO_DATE('2017-02-01 00:00:01','SYYYY-MM-DD HH24:MI:SS'))
 INTO
 (
 PARTITION P2017JAN TABLESPACE CM_F1T191_P2017JAN
LOB(BO_DATA_AREA,PRE_TRN_INIT_BO_DATA_AREA,PRE_TRN_FIN_BO_DATA_
AREA,POST_TRN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE
IN ROW COMPRESS MEDIUM CACHE TABLESPACE
CM_F1T191_P2017JAN )
, PARTITION P2017FEB
```

```
)
UPDATE INDEXES;
```

3. Enable advanced compression after SPLIT partition as it will disable the compression.

```
ALTER TABLE CISADM.CI_TD_SRTKEY ROW STORE COMPRESS ADVANCED;
ALTER TABLE CISADM.CI_TD_MSG_PARM ROW STORE COMPRESS ADVANCED;
ALTER TABLE CISADM.CI_TD_DRLKEY ROW STORE COMPRESS ADVANCED;
ALTER TABLE CISADM.CI_TD_ENTRY_CHA ROW STORE COMPRESS ADVANCED;
ALTER TABLE CISADM.CI_TD_LOG ROW STORE COMPRESS ADVANCED;
```

4. Import tablespace using TRANSPORT_TABLESPACES method.

```
impdp system/manager DIRECTORY=DUMP_DIR DUMPFILE=
CM_XT039_P2017JAN_ARC.DMP PARTITION_OPTIONS=DEPARTITION
LOGFILE=IMP_CM_XT039_P2017JAN_ARC.LOG TRANSPORT_DATAFILES=/
tugbu_perf_02/BACKUPS/test_verification/
cm_xt039_p201701jan_ar.553.913864937
```

5. Load data into parent table first from the staging table

```
 ALTER SESSION ENABLE PARALLEL DML;

INSERT /*+ APPEND PARALLEL */   INTO CISADM.CI_TD_ENTRY SELECT /
*+ PARALLEL */  * FROM CM_XT039_P2017JAN_ARC;
COMMIT;
```

6. Load data into child table from the staging table

   For each Child IN LIST OF CHILD TABLES, perform the following:

```
INSERT /*+ APPEND PARALLEL */   INTO CISADM.CI_TD_ENTRY_CHA
SELECT /*+ PARALLEL */ * FROM CM_XT701_P2017JAN_ARC;
COMMIT;
INSERT /*+ APPEND PARALLEL */   INTO CISADM.CI_TD_MSG_PARM
SELECT /*+ PARALLEL */ * FROM CM_XT04_P2017JAN_ARC;
COMMIT;

INSERT /*+ APPEND PARALLEL */   INTO CISADM.CI_TD_LOG SELECT /*+
PARALLEL */ * FROM CM_XT721_P2017JAN_ARC;
COMMIT;

INSERT /*+ APPEND PARALLEL */   INTO CISADM.CI_TD_SRTKEY SELECT
/*+ PARALLEL */ * FROM CM_XT041_P2017JAN_ARC;
COMMIT;

INSERT /*+ APPEND PARALLEL */   INTO CISADM.CI_TD_DRLKEY SELECT
/*+ PARALLEL */ * FROM CM_XT037_P2017JAN_ARC;
COMMIT;
```

7. Drop the archive tablespace after import is import and data loading is successful.

```
DROP TABLESPACE CM_XT039_P2017JAN_ARC INCLUDING CONTENTS AND
DATAFILES;
```

Appendix D

Sample SQL for ILM in CCB

These are the sample SQL scripts which has the recommended way to partition the Bill Segment and Adjustment tables and Indexes. Implementations can further customize these scripts and update the partition names, tablespace names and date ranges to make sure they are suited to the implementation.

This appendix consists:

Maintenance Object: Adjustment

This section contains the sample SQL for the following tables:

Parent Table: CI_ADJ

```
CREATE BIGFILE TABLESPACE CM_XT012_P2017JAN DATAFILE '+DATA' SIZE 100M
AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED
/
CREATE BIGFILE TABLESPACE CM_XT012_P2017FEB DATAFILE '+DATA' SIZE 100M
AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED
/
CREATE BIGFILE TABLESPACE CM_XT012_P2017MAR DATAFILE '+DATA' SIZE 100M
AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED
/
CREATE BIGFILE TABLESPACE CM_XT012_P2017APR DATAFILE '+DATA' SIZE 100M
AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED
/
CREATE BIGFILE TABLESPACE CM_XT012_P2017MAY DATAFILE '+DATA' SIZE 100M
AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED
/
CREATE BIGFILE TABLESPACE CM_XT012_P2017JUN DATAFILE '+DATA' SIZE 100M
AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED
/
CREATE BIGFILE TABLESPACE CM_XT012_P2017JUL DATAFILE '+DATA' SIZE 100M
AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED
/
CREATE BIGFILE TABLESPACE CM_XT012_P2017AUG DATAFILE '+DATA' SIZE 100M
AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED
/
CREATE BIGFILE TABLESPACE CM_XT012_P2017SEP DATAFILE '+DATA' SIZE 100M
AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED
/
CREATE BIGFILE TABLESPACE CM_XT012_P2017OCT DATAFILE '+DATA' SIZE 100M
AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED
/
CREATE BIGFILE TABLESPACE CM_XT012_P2017NOV DATAFILE '+DATA' SIZE 100M
AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED
/
CREATE BIGFILE TABLESPACE CM_XT012_P2017DEC DATAFILE '+DATA' SIZE 100M
AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED
/
CREATE BIGFILE TABLESPACE CM_XT012_PMAX DATAFILE '+DATA' SIZE 100M
AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED
/

CREATE TABLE CI_ADJ
(  CHAR(12) NOT NULL ENABLE,
ADJ_ID
SA_IDCHAR(10) DEFAULT ' ' NOT NULL ENABLE,
ADJ_TYPE_CDCHAR(8) DEFAULT' ' NOT NULL ENABLE,
ADJ_STATUS_FLG CHAR(2) DEFAULT' ' NOT NULL ENABLE,
```

```
CRE_DT DATE,CHAR(4) DEFAULT '' NOT NULL ENABLE,
CAN_RSN_CD
ADJ_AMTNUMBER(15,2) DEFAULT 0NOT NULL ENABLE,
XFER_ADJ_IDCHAR(12) DEFAULT ' ' NOT NULL ENABLE,
CURRENCY_CDCHAR(3) DEFAULT '' NOT NULL ENABLE,
COMMENTSVARCHAR2(254) DEFAULT ' ' NOT NULL ENABLE,
VERSIONNUMBER(5,0) DEFAULT 1 NOT NULL ENABLE,
BEHALF_SA_IDCHAR(10) DEFAULT ' ' NOT NULL ENABLE,
BASE_AMTNUMBER(15,2) DEFAULT 0NOT NULL ENABLE,
GEN_REF_DT DATE,

APPR_REQ_ID CHAR(12) DEFAULT ' ' NOT NULL ENABLE,
ADJ_DATA_AREA CLOB,
ILM_DT DATE,
ILM_ARCH_SW CHAR(1),
)
ENABLE ROW MOVEMENT
PARTITION BY RANGE (ILM_DT)
SUBPARTITION BY RANGE (ADJ_ID) SUBPARTITION TEMPLATE (
SUBPARTITION S01 VALUES LESS THAN ( '124999999999' ),
SUBPARTITION S02 VALUES LESS THAN ( '249999999999' ),
SUBPARTITION S03 VALUES LESS THAN ( '374999999999' ),
SUBPARTITION S04 VALUES LESS THAN ( '499999999999' ),
SUBPARTITION S05 VALUES LESS THAN ( '624999999999' ),
SUBPARTITION S06 VALUES LESS THAN ( '749999999999' ),
SUBPARTITION S07 VALUES LESS THAN ( '874999999999' ),
SUBPARTITION S08 VALUES LESS THAN ( MAXVALUE )
)

(
PARTITION "P2017JAN" VALUES LESS THAN (TO_DATE('2017-02-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT012_P2017JAN,
PARTITION "P2017FEB" VALUES LESS THAN (TO_DATE('2017-03-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT012_P2017FEB,
PARTITION "P2017MAR" VALUES LESS THAN (TO_DATE('2017-04-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT012_P2017MAR,

PARTITION "P2017APR" VALUES LESS THAN (TO_DATE('2017-05-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT012_P2017APR,
PARTITION "P2017MAY" VALUES LESS THAN (TO_DATE('2017-06-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT012_P2017MAY,
PARTITION "P2017JUN" VALUES LESS THAN (TO_DATE('2017-07-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT012_P2017JUN,
PARTITION "P2017JUL" VALUES LESS THAN (TO_DATE('2017-08-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT012_P2017JUL,
PARTITION "P2017AUG" VALUES LESS THAN (TO_DATE('2017-09-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT012_P2017AUG,
PARTITION "P2017SEP" VALUES LESS THAN (TO_DATE('2017-10-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT012_P2017SEP,
PARTITION "P2017OCT" VALUES LESS THAN (TO_DATE('2017-11-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT012_P2017OCT,
```

```
PARTITION "P2017NOV" VALUES LESS THAN (TO_DATE('2017-12-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT012_P2017NOV,
PARTITION "P2017DEC" VALUES LESS THAN (TO_DATE('2018-01-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT012_P2017DEC,
PARTITION "PMAX" VALUES LESS THAN (MAXVALUE)
tablespace CM_XT012_PMAX
)

/
```

INDEX

```
CREATE BIGFILE TABLESPACE CM_XT012_IND DATAFILE '+DATA' SIZE 50M
AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED
/

CREATE UNIQUE INDEX XT012P0 ON CI_ADJ ( ADJ_ID ) TABLESPACE
CM_XT012_IND
GLOBAL PARTITION BY RANGE (ADJ_ID)
(
PARTITION P1 VALUES LESS THAN ( '124999999999' ),
PARTITION P2 VALUES LESS THAN ( '249999999999' ),
PARTITION P3 VALUES LESS THAN ( '374999999999' ),
PARTITION P4 VALUES LESS THAN ( '499999999999' ),
PARTITION P5 VALUES LESS THAN ( '624999999999' ),
PARTITION P6 VALUES LESS THAN ( '749999999999' ),
PARTITION P7 VALUES LESS THAN ( '874999999999' ),
PARTITION P8 VALUES LESS THAN ( MAXVALUE )
)
/

ALTER TABLE CI_ADJ ADD CONSTRAINT XT012P0 PRIMARY KEY(ADJ_ID) USING
INDEX
/
CREATE INDEX XT012S1 ON CI_ADJ ( SA_ID, ADJ_TYPE_CD ) TABLESPACE
CM_XT012_IND COMPRESS ADVANCED LOW
/
CREATE UNIQUE INDEX XT012S2 ON CI_ADJ ( XFER_ADJ_ID, ADJ_ID )
TABLESPACE CM_XT012_IND COMPRESS ADVANCED LOW
/
CREATE UNIQUE INDEX XT012S3 ON CI_ADJ ( ILM_DT, ILM_ARCH_SW, ADJ_ID )
TABLESPACE CM_XT012_IND COMPRESS ADVANCED LOW
/
```

Child Table: CI_ADJ_APREQ

```
CREATE TABLE CI_ADJ_APREQ
  (
    AP_REQ_ID      CHAR(12) NOT NULL ENABLE,
    COUNTRY        CHAR(3) DEFAULT ' ' NOT NULL ENABLE,
    ADDRESS1       VARCHAR2(254) DEFAULT ' ' NOT NULL ENABLE,
    ADJ_ID         CHAR(12) DEFAULT ' ' NOT NULL ENABLE,
    ADDRESS2       VARCHAR2(254) DEFAULT ' ' NOT NULL ENABLE,
    ADDRESS3       VARCHAR2(254) DEFAULT ' ' NOT NULL ENABLE,
    ADDRESS4       VARCHAR2(254) DEFAULT ' ' NOT NULL ENABLE,
    CITY           VARCHAR2(90) DEFAULT ' ' NOT NULL ENABLE,
    NUM1           CHAR(6) DEFAULT ' ' NOT NULL ENABLE,
```

```
    NUM2           CHAR(4) DEFAULT ' ' NOT NULL ENABLE,
    COUNTY         VARCHAR2(90) DEFAULT ' ' NOT NULL ENABLE,
    HOUSE_TYPE     CHAR(2) DEFAULT ' ' NOT NULL ENABLE,
    STATE          CHAR(6) DEFAULT ' ' NOT NULL ENABLE,
    POSTAL         CHAR(12) DEFAULT ' ' NOT NULL ENABLE,
    CURRENCY_PYMNT CHAR(3) DEFAULT ' ' NOT NULL ENABLE,
    GEO_CODE       CHAR(11) DEFAULT ' ' NOT NULL ENABLE,
    IN_CITY_LIMIT  CHAR(1) DEFAULT ' ' NOT NULL ENABLE,
    PAID_AMT       NUMBER(15,2) DEFAULT 0 NOT NULL ENABLE,
    SCHEDULED_PAY_DT DATE,
    PYMNT_DT DATE,
    ENTITY_NAME VARCHAR2(254) DEFAULT ' ' NOT NULL ENABLE,
    PAY_DOC_ID  VARCHAR2(20) DEFAULT ' ' NOT NULL ENABLE,
    PAY_DOC_DT DATE,
    PYMNT_ID           CHAR(36) DEFAULT ' ' NOT NULL ENABLE,
    PYMNT_METHOD_FLG   CHAR(3) DEFAULT ' ' NOT NULL ENABLE,
    PYMNT_SEL_STAT_FLG CHAR(1) DEFAULT ' ' NOT NULL ENABLE,
    VERSION            NUMBER(5,0) DEFAULT 1 NOT NULL ENABLE,
    BATCH_CD           CHAR(8) DEFAULT ' ' NOT NULL ENABLE,
    BATCH_NBR          NUMBER(10,0) DEFAULT 0 NOT NULL ENABLE,
CONSTRAINT CI_ADJ_APREQ_FK FOREIGN KEY(ADJ_ID) REFERENCES CI_ADJ ON
DELETE CASCADE
)
PARTITION BY REFERENCE (CI_ADJ_APREQ_FK)
ENABLE ROW MOVEMENT
/
```

INDEX

```
CREATE UNIQUE INDEX XT160P0 ON CI_ADJ_APREQ ( AP_REQ_ID ) TABLESPACE
CM_XT012_IND
GLOBAL PARTITION BY RANGE (AP_REQ_ID)
(
PARTITION P1 VALUES LESS THAN ( '124999999999' ),
PARTITION P2 VALUES LESS THAN ( '249999999999' ),
PARTITION P3 VALUES LESS THAN ( '374999999999' ),
PARTITION P4 VALUES LESS THAN ( '499999999999' ),
PARTITION P5 VALUES LESS THAN ( '624999999999' ),
PARTITION P6 VALUES LESS THAN ( '749999999999' ),
PARTITION P7 VALUES LESS THAN ( '874999999999' ),
PARTITION P8 VALUES LESS THAN ( MAXVALUE )
)
COMPRESS ADVANCED LOW
/

ALTER TABLE CI_ADJ_APREQ ADD CONSTRAINT XT160P0 PRIMARY KEY(AP_REQ_ID)
USING INDEX
/
CREATE INDEX XT160S1 ON CI_ADJ_APREQ ( ADJ_ID ) TABLESPACE
CM_XT012_IND
/
CREATE INDEX XT160S2 ON CI_ADJ_APREQ ( BATCH_CD, BATCH_NBR )
TABLESPACE CM_XT012_IND COMPRESS ADVANCED LOW
/
```

Child Table: CI_ADJ_CALC_LN

```
CREATE TABLE CI_ADJ_CALC_LN
  (
    ADJ_ID CHAR(12) NOT NULL ENABLE,
    SEQNO  NUMBER(5,0) NOT NULL ENABLE,
```

```
    TOU_CD CHAR(8) DEFAULT ' ' NOT NULL ENABLE,
    UOM_CD CHAR(4) DEFAULT ' ' NOT NULL ENABLE,
    SQI_CD CHAR(8) DEFAULT ' ' NOT NULL ENABLE,
    RS_CD  CHAR(8) DEFAULT ' ' NOT NULL ENABLE,
    EFFDT DATE,
    RC_SEQ          NUMBER(4,0) DEFAULT 0 NOT NULL ENABLE,
    DST_ID          CHAR(10) DEFAULT ' ' NOT NULL ENABLE,
    CURRENCY_CD     CHAR(3) DEFAULT ' ' NOT NULL ENABLE,
    CHAR_TYPE_CD    CHAR(8) DEFAULT ' ' NOT NULL ENABLE,
    CHAR_VAL        CHAR(16) DEFAULT ' ' NOT NULL ENABLE,
    PRT_SW          CHAR(1) DEFAULT ' ' NOT NULL ENABLE,
    APP_IN_SUMM_SW  CHAR(1) DEFAULT ' ' NOT NULL ENABLE,
    CALC_AMT        NUMBER(15,2) DEFAULT 0 NOT NULL ENABLE,
    EXEMPT_AMT      NUMBER(15,2) DEFAULT 0 NOT NULL ENABLE,
    BASE_AMT        NUMBER(15,2) DEFAULT 0 NOT NULL ENABLE,
    MSR_PEAK_QTY_SW CHAR(1) DEFAULT ' ' NOT NULL ENABLE,
    VERSION         NUMBER(5,0) DEFAULT 1 NOT NULL ENABLE,
    DESCR_ON_BILL   VARCHAR2(254) DEFAULT ' ' NOT NULL ENABLE,
    BILL_SQ         NUMBER(18,6) DEFAULT 0 NOT NULL ENABLE,
    AUDIT_CALC_AMT  NUMBER(18,5) DEFAULT 0 NOT NULL ENABLE,
    CALC_GRP_CD     VARCHAR2(30) DEFAULT ' ' NOT NULL ENABLE,
    CALC_RULE_CD    VARCHAR2(30) DEFAULT ' ' NOT NULL ENABLE,
CONSTRAINT CI_ADJ_CALC_LN_FK FOREIGN KEY(ADJ_ID) REFERENCES CI_ADJ ON
DELETE CASCADE
)
PARTITION BY REFERENCE (CI_ADJ_CALC_LN_FK)
ENABLE ROW MOVEMENT
/
```

INDEX

```
CREATE UNIQUE INDEX XT310P0 ON CI_ADJ_CALC_LN ( ADJ_ID, SEQNO )
TABLESPACE CM_XT012_IND
GLOBAL PARTITION BY RANGE (ADJ_ID)
(
PARTITION P1 VALUES LESS THAN ( '124999999999' ),
PARTITION P2 VALUES LESS THAN ( '249999999999' ),
PARTITION P3 VALUES LESS THAN ( '374999999999' ),
PARTITION P4 VALUES LESS THAN ( '499999999999' ),
PARTITION P5 VALUES LESS THAN ( '624999999999' ),
PARTITION P6 VALUES LESS THAN ( '749999999999' ),
PARTITION P7 VALUES LESS THAN ( '874999999999' ),
PARTITION P8 VALUES LESS THAN ( MAXVALUE )
)
COMPRESS ADVANCED LOW
/

ALTER TABLE CI_ADJ_CALC_LN ADD CONSTRAINT XT310P0 PRIMARY KEY(ADJ_ID,
SEQNO) USING INDEX
/
```

Child Table: CI_ADJ_CL_CHAR

```
CREATE TABLE CI_ADJ_CL_CHAR
  (
    ADJ_ID         CHAR(12) NOT NULL ENABLE,
    SEQNO          NUMBER(5,0) NOT NULL ENABLE,
    CHAR_TYPE_CD   CHAR(8) NOT NULL ENABLE,
    VERSION        NUMBER(5,0) DEFAULT 1 NOT NULL ENABLE,
    CHAR_VAL       CHAR(16) DEFAULT ' ' NOT NULL ENABLE,
    ADHOC_CHAR_VAL VARCHAR2(254) DEFAULT ' ' NOT NULL ENABLE,
```

```
    CHAR_VAL_FK1   VARCHAR2(50) DEFAULT ' ' NOT NULL ENABLE,
    CHAR_VAL_FK2   VARCHAR2(50) DEFAULT ' ' NOT NULL ENABLE,
    CHAR_VAL_FK3   VARCHAR2(50) DEFAULT ' ' NOT NULL ENABLE,
    CHAR_VAL_FK4   VARCHAR2(50) DEFAULT ' ' NOT NULL ENABLE,
    CHAR_VAL_FK5   VARCHAR2(50) DEFAULT ' ' NOT NULL ENABLE,
CONSTRAINT CI_ADJ_CL_CHAR_FK FOREIGN KEY(ADJ_ID) REFERENCES CI_ADJ ON
DELETE CASCADE
)
PARTITION BY REFERENCE (CI_ADJ_CL_CHAR_FK)
ENABLE ROW MOVEMENT
/
```

INDEX

```
CREATE UNIQUE INDEX XT309P0 ON CI_ADJ_CL_CHAR ( ADJ_ID, SEQNO,
CHAR_TYPE_CD ) TABLESPACE CM_XT012_IND
GLOBAL PARTITION BY RANGE (ADJ_ID)
(
PARTITION P1 VALUES LESS THAN ( '124999999999' ),
PARTITION P2 VALUES LESS THAN ( '249999999999' ),
PARTITION P3 VALUES LESS THAN ( '374999999999' ),
PARTITION P4 VALUES LESS THAN ( '499999999999' ),
PARTITION P5 VALUES LESS THAN ( '624999999999' ),
PARTITION P6 VALUES LESS THAN ( '749999999999' ),
PARTITION P7 VALUES LESS THAN ( '874999999999' ),
PARTITION P8 VALUES LESS THAN ( MAXVALUE )
)
COMPRESS ADVANCED LOW
/

ALTER TABLE CI_ADJ_CL_CHAR ADD CONSTRAINT XT309P0 PRIMARY KEY(ADJ_ID,
SEQNO, CHAR_TYPE_CD) USING INDEX
/
```

Child Table: CI_ADJ_CHAR

```
CREATE TABLE CI_ADJ_CHAR
  (
    ADJ_ID         CHAR(12) NOT NULL ENABLE,
    CHAR_TYPE_CD   CHAR(8) NOT NULL ENABLE,
    SEQ_NUM        NUMBER(3,0) NOT NULL ENABLE,
    VERSION        NUMBER(5,0) DEFAULT 1 NOT NULL ENABLE,
    CHAR_VAL       CHAR(16) DEFAULT ' ' NOT NULL ENABLE,
    ADHOC_CHAR_VAL VARCHAR2(254) DEFAULT ' ' NOT NULL ENABLE,
    CHAR_VAL_FK1   VARCHAR2(50) DEFAULT ' ' NOT NULL ENABLE,
    CHAR_VAL_FK2   VARCHAR2(50) DEFAULT ' ' NOT NULL ENABLE,
    CHAR_VAL_FK3   VARCHAR2(50) DEFAULT ' ' NOT NULL ENABLE,
    CHAR_VAL_FK4   VARCHAR2(50) DEFAULT ' ' NOT NULL ENABLE,
    CHAR_VAL_FK5   VARCHAR2(50) DEFAULT ' ' NOT NULL ENABLE,
    SRCH_CHAR_VAL  VARCHAR2(50) DEFAULT ' ' NOT NULL ENABLE,
CONSTRAINT CI_ADJ_CHAR_FK FOREIGN KEY(ADJ_ID) REFERENCES CI_ADJ ON
DELETE CASCADE
)
PARTITION BY REFERENCE (CI_ADJ_CHAR_FK)
ENABLE ROW MOVEMENT
/
```

INDEX

```
CREATE UNIQUE INDEX XC781P0 ON CI_ADJ_CHAR (ADJ_ID, CHAR_TYPE_CD,
SEQ_NUM) TABLESPACE CM_XT012_IND
GLOBAL PARTITION BY RANGE(ADJ_ID)
 (
 PARTITION PART1 VALUES LESS THAN ('124999999999'),
 PARTITION PART2 VALUES LESS THAN ('249999999999'),
 PARTITION PART3 VALUES LESS THAN ('374999999999'),
 PARTITION PART4 VALUES LESS THAN ('499999999999'),
 PARTITION PART5 VALUES LESS THAN ('624999999999'),
 PARTITION PART6 VALUES LESS THAN ('749999999999'),
 PARTITION PART7 VALUES LESS THAN ('874999999999'),
 PARTITION PART8 VALUES LESS THAN (MAXVALUE)
 )
COMPRESS ADVANCED LOW
/

ALTER TABLE CI_ADJ_CHAR ADD CONSTRAINT XC781P0 PRIMARY KEY (ADJ_ID,
CHAR_TYPE_CD, SEQ_NUM) USING INDEX;

CREATE INDEX XC781S1 ON CI_ADJ_CHAR(SRCH_CHAR_VAL)
GLOBAL PARTITION BY HASH(SRCH_CHAR_VAL)
 (
 PARTITION PART1 TABLESPACE CM_XT012_IND,
 PARTITION PART2 TABLESPACE CM_XT012_IND,
 PARTITION PART3 TABLESPACE CM_XT012_IND,
 PARTITION PART4 TABLESPACE CM_XT012_IND,
 PARTITION PART5 TABLESPACE CM_XT012_IND,
 PARTITION PART6 TABLESPACE CM_XT012_IND,
 PARTITION PART7 TABLESPACE CM_XT012_IND,
 PARTITION PART8 TABLESPACE CM_XT012_IND
)
/
```

Maintenance Object: Bill Segment

This section contains the sample SQL for the following tables:

Parent Table: CI_BSEG

```
CREATE BIGFILE TABLESPACE CM_XT048_P2017JAN DATAFILE '+DATA' SIZE 100M
AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED
/
CREATE BIGFILE TABLESPACE CM_XT048_P2017FEB DATAFILE '+DATA' SIZE 100M
AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED
/
CREATE BIGFILE TABLESPACE CM_XT048_P2017MAR DATAFILE '+DATA' SIZE 100M
AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED
/
CREATE BIGFILE TABLESPACE CM_XT048_P2017APR DATAFILE '+DATA' SIZE 100M
AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED
/
CREATE BIGFILE TABLESPACE CM_XT048_P2017MAY DATAFILE '+DATA' SIZE 100M
AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED
/
CREATE BIGFILE TABLESPACE CM_XT048_P2017JUN DATAFILE '+DATA' SIZE 100M
AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED
/
CREATE BIGFILE TABLESPACE CM_XT048_P2017JUL DATAFILE '+DATA' SIZE 100M
AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED
/
CREATE BIGFILE TABLESPACE CM_XT048_P2017AUG DATAFILE '+DATA' SIZE 100M
AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED
/
CREATE BIGFILE TABLESPACE CM_XT048_P2017SEP DATAFILE '+DATA' SIZE 100M
AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED
/
CREATE BIGFILE TABLESPACE CM_XT048_P2017OCT DATAFILE '+DATA' SIZE 100M
AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED
/
CREATE BIGFILE TABLESPACE CM_XT048_P2017NOV DATAFILE '+DATA' SIZE 100M
AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED
/
CREATE BIGFILE TABLESPACE CM_XT048_P2017DEC DATAFILE '+DATA' SIZE 100M
AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED
/
CREATE BIGFILE TABLESPACE CM_XT048_PMAX DATAFILE '+DATA' SIZE 100M
AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED
/
```

```
CREATE TABLE CI_BSEG
(  CHAR(12) NOTNULL ENABLE,
BSEG_ID
BILL_CYC_CDCHAR(4) DEFAULT ' ' NOT NULL ENABLE,
WIN_START_DT DATE,
CAN_RSN_CDCHAR(4) DEFAULT ' ' NOT NULL ENABLE,
CAN_BSEG_IDCHAR(12) DEFAULT ' 'NOT NULL ENABLE,
SA_IDCHAR(10) DEFAULT ' 'NOT NULL ENABLE,
BILL_IDCHAR(12) DEFAULT ' 'NOT NULL ENABLE,
START_DT DATE,
END_DT DATE,CHAR(1)DEFAULT' ' NOT NULL ENABLE,
EST_SW
CLOSING_BSEG_SWCHAR(1)DEFAULT' ' NOT NULL ENABLE,
SQ_OVERRIDE_SWCHAR(1)DEFAULT' ' NOT NULL ENABLE,
ITEM_OVERRIDE_SW CHAR(1)DEFAULT' ' NOT NULL ENABLE,
PREM_IDCHAR(10) DEFAULT ' ' NOT NULL ENABLE,
BSEG_STAT_FLGCHAR(2)DEFAULT' ' NOT NULL ENABLE,
CRE_DTTM DATE,
STAT_CHG_DTTM DATE,
REBILL_SEG_IDCHAR(12) DEFAULT ' 'NOT NULL ENABLE,
VERSIONNUMBER(5,0) DEFAULT 1NOTNULL ENABLE,
MASTER_BSEG_IDCHAR(12) DEFAULT ' 'NOT NULL ENABLE,
QUOTE_DTL_IDCHAR(12)DEFAULT ''NOT NULL ENABLE,
BILL_SCNR_IDCHAR(12)DEFAULT''NOT NULL ENABLE,
MDM_START_DTTM DATE,
MDM_END_DTTM DATE,
BSEG_DATA_AREA CLOB,
ILM_DT DATE,
ILM_ARCH_SW CHAR(1)
)
ENABLE ROW MOVEMENT
PARTITION BY RANGE (ILM_DT)
SUBPARTITION BY RANGE (BSEG_ID) SUBPARTITION TEMPLATE (
SUBPARTITION S01 VALUES LESS THAN ( '124999999999' ),
SUBPARTITION S02 VALUES LESS THAN ( '249999999999' ),
SUBPARTITION S03 VALUES LESS THAN ( '374999999999' ),
SUBPARTITION S04 VALUES LESS THAN ( '499999999999' ),
SUBPARTITION S05 VALUES LESS THAN ( '624999999999' ),
SUBPARTITION S06 VALUES LESS THAN ( '749999999999' ),
SUBPARTITION S07 VALUES LESS THAN ( '874999999999' ),
SUBPARTITION S08 VALUES LESS THAN ( MAXVALUE )
)

(
PARTITION "P2017JAN" VALUES LESS THAN (TO_DATE('2017-02-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT048_P2017JAN,
PARTITION "P2017FEB" VALUES LESS THAN (TO_DATE('2017-03-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT048_P2017FEB,
PARTITION "P2017MAR" VALUES LESS THAN (TO_DATE('2017-04-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT048_P2017MAR,
PARTITION "P2017APR" VALUES LESS THAN (TO_DATE('2017-05-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT048_P2017APR,
PARTITION "P2017MAY" VALUES LESS THAN (TO_DATE('2017-06-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT048_P2017MAY,
PARTITION "P2017JUN" VALUES LESS THAN (TO_DATE('2017-07-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
```

```
tablespace CM_XT048_P2017JUN,
PARTITION "P2017JUL" VALUES LESS THAN (TO_DATE('2017-08-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT048_P2017JUL,
PARTITION "P2017AUG" VALUES LESS THAN (TO_DATE('2017-09-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT048_P2017AUG,
PARTITION "P2017SEP" VALUES LESS THAN (TO_DATE('2017-10-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT048_P2017SEP,
PARTITION "P2017OCT" VALUES LESS THAN (TO_DATE('2017-11-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT048_P2017OCT,
PARTITION "P2017NOV" VALUES LESS THAN (TO_DATE('2017-12-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT048_P2017NOV,
PARTITION "P2017DEC" VALUES LESS THAN (TO_DATE('2018-01-01 00:00:01',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
tablespace CM_XT048_P2017DEC,
PARTITION "PMAX" VALUES LESS THAN (MAXVALUE)
tablespace CM_XT048_PMAX
)

/
CREATE BIGFILE TABLESPACE CM_XT048_IND DATAFILE '+DATA' SIZE 100M
AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
```

INDEX

```
CREATE UNIQUE INDEX XT048P0 ON CI_BSEG ( BSEG_ID ) TABLESPACE
CM_XT048_IND
GLOBAL PARTITION BY RANGE (BSEG_ID)
(
PARTITION P1 VALUES LESS THAN ( '124999999999' ),
PARTITION P2 VALUES LESS THAN ( '249999999999' ),
PARTITION P3 VALUES LESS THAN ( '374999999999' ),
PARTITION P4 VALUES LESS THAN ( '499999999999' ),
PARTITION P5 VALUES LESS THAN ( '624999999999' ),
PARTITION P6 VALUES LESS THAN ( '749999999999' ),
PARTITION P7 VALUES LESS THAN ( '874999999999' ),
PARTITION P8 VALUES LESS THAN ( MAXVALUE )
)
/
ALTER TABLE CI_BSEG ADD CONSTRAINT XT048P0 PRIMARY KEY(BSEG_ID) USING
INDEX
/
CREATE INDEX XT048S1 ON CI_BSEG ( BILL_ID ) TABLESPACE CM_XT048_IND
/
CREATE INDEX XT048S2 ON CI_BSEG ( SA_ID ) TABLESPACE CM_XT048_IND
/
CREATE UNIQUE INDEX XT048S3 ON CI_BSEG ( QUOTE_DTL_ID, BSEG_ID
)TABLESPACE CM_XT048_IND COMPRESS ADVANCED LOW
/
CREATE UNIQUE INDEX XT048S4 ON CI_BSEG ( ILM_DT, ILM_ARCH_SW, BSEG_ID )
TABLESPACE CM_XT048_IND COMPRESS ADVANCED LOW
/
```

Child Table: CI_BSEG_CALC

```
CREATE TABLE CI_BSEG_CALC
(
BSEG_ID    CHAR(12) NOT NULL ENABLE,
```

```
HEADER_SEQ NUMBER(3,0) NOT NULL ENABLE,
START_DT DATE NOT NULL ENABLE,
CURRENCY_CD CHAR(3) DEFAULT ' ' NOT NULL ENABLE,
END_DT DATE NOT NULL ENABLE,
RS_CD CHAR(8) DEFAULT ' ' NOT NULL ENABLE,
EFFDT DATE,
BILLABLE_CHG_ID CHAR(12) DEFAULT ' ' NOT NULL ENABLE,
CALC_AMT        NUMBER(15,2) DEFAULT 0 NOT NULL ENABLE,
DESCR_ON_BILL   VARCHAR2(254) DEFAULT ' ' NOT NULL ENABLE,
VERSION         NUMBER(5,0) DEFAULT 1 NOT NULL ENABLE,
CONSTRAINT CI_BSEG_CALC_FK FOREIGN KEY(BSEG_ID) REFERENCES CI_BSEG ON
DELETE CASCADE
)
PARTITION BY REFERENCE (CI_BSEG_CALC_FK)
ENABLE ROW MOVEMENT
/
```

INDEX

```
CREATE UNIQUE INDEX XT072P0 ON CI_BSEG_CALC ( BSEG_ID, HEADER_SEQ )
TABLESPACE CM_XT048_IND
GLOBAL PARTITION BY RANGE (BSEG_ID)
(
PARTITION P1 VALUES LESS THAN ( '124999999999' ),
PARTITION P2 VALUES LESS THAN ( '249999999999' ),
PARTITION P3 VALUES LESS THAN ( '374999999999' ),
PARTITION P4 VALUES LESS THAN ( '499999999999' ),
PARTITION P5 VALUES LESS THAN ( '624999999999' ),
PARTITION P6 VALUES LESS THAN ( '749999999999' ),
PARTITION P7 VALUES LESS THAN ( '874999999999' ),
PARTITION P8 VALUES LESS THAN ( MAXVALUE )
)
COMPRESS ADVANCED LOW
/

ALTER TABLE CI_BSEG_CALC ADD CONSTRAINT XT072P0 PRIMARY KEY(BSEG_ID,
HEADER_SEQ) USING INDEX
/
CREATE INDEX XT072S1 ON CI_BSEG_CALC ( BILLABLE_CHG_ID,  BSEG_ID )
TABLESPACE CM_XT048_IND COMPRESS ADVANCED LOW
/
```

Child Table: CI_BSEG_CALC_LN

```
CREATE TABLE CI_BSEG_CALC_LN
  (
    BSEG_ID         CHAR(12) NOT NULL ENABLE,
    HEADER_SEQ      NUMBER(3,0) NOT NULL ENABLE,
    SEQNO           NUMBER(5,0) NOT NULL ENABLE,
    CHAR_TYPE_CD    CHAR(8) DEFAULT ' ' NOT NULL ENABLE,
    CURRENCY_CD     CHAR(3) DEFAULT ' ' NOT NULL ENABLE,
    CHAR_VAL        CHAR(16) DEFAULT ' ' NOT NULL ENABLE,
    DST_ID          CHAR(10) DEFAULT ' ' NOT NULL ENABLE,
    UOM_CD          CHAR(4) DEFAULT ' ' NOT NULL ENABLE,
    TOU_CD          CHAR(8) DEFAULT ' ' NOT NULL ENABLE,
    RC_SEQ          NUMBER(4,0) DEFAULT 0 NOT NULL ENABLE,
    PRT_SW          CHAR(1) DEFAULT ' ' NOT NULL ENABLE,
    APP_IN_SUMM_SW  CHAR(1) DEFAULT ' ' NOT NULL ENABLE,
    CALC_AMT        NUMBER(15,2) DEFAULT 0 NOT NULL ENABLE,
    EXEMPT_AMT      NUMBER(15,2) DEFAULT 0 NOT NULL ENABLE,
    BASE_AMT        NUMBER(15,2) DEFAULT 0 NOT NULL ENABLE,
    SQI_CD          CHAR(8) DEFAULT ' ' NOT NULL ENABLE,
```

```
    BILL_SQ         NUMBER(18,6) DEFAULT 0 NOT NULL ENABLE,
    MSR_PEAK_QTY_SW CHAR(1) DEFAULT ' ' NOT NULL ENABLE,
    DESCR_ON_BILL   VARCHAR2(254) DEFAULT ' ' NOT NULL ENABLE,
    VERSION         NUMBER(5,0) DEFAULT 1 NOT NULL ENABLE,
    AUDIT_CALC_AMT  NUMBER(18,5) DEFAULT 0 NOT NULL ENABLE,
    CALC_GRP_CD     VARCHAR2(30) DEFAULT ' ' NOT NULL ENABLE,
    CALC_RULE_CD    VARCHAR2(30) DEFAULT ' ' NOT NULL ENABLE,
 CONSTRAINT CI_BSEG_CALC_LN_FK FOREIGN KEY(BSEG_ID) REFERENCES CI_BSEG
ON DELETE CASCADE
)
PARTITION BY REFERENCE (CI_BSEG_CALC_LN_FK)
ENABLE ROW MOVEMENT
/
```

INDEX

```
CREATE UNIQUE INDEX XT050P0 ON CM_BSEG_CALC_LN ( BSEG_ID,
HEADER_SEQ,SEQNO) TABLESPACE CM_XT048_IND
GLOBAL PARTITION BY RANGE (BSEG_ID)
(
PARTITION P1 VALUES LESS THAN ( '124999999999' ),
PARTITION P2 VALUES LESS THAN ( '249999999999' ),
PARTITION P3 VALUES LESS THAN ( '374999999999' ),
PARTITION P4 VALUES LESS THAN ( '499999999999' ),
PARTITION P5 VALUES LESS THAN ( '624999999999' ),
PARTITION P6 VALUES LESS THAN ( '749999999999' ),
PARTITION P7 VALUES LESS THAN ( '874999999999' ),
PARTITION P8 VALUES LESS THAN ( MAXVALUE )
)
COMPRESS ADVANCED LOW
/

ALTER TABLE CI_BSEG_CALC_LN ADD CONSTRAINT XT050P0 PRIMARY
KEY(BSEG_ID, HEADER_SEQ,SEQNO) USING INDEX
/
```

Child Table: CI_BSEG_CL_CHAR

```
CREATE TABLE CI_BSEG_CL_CHAR
  (
    BSEG_ID        CHAR(12) NOT NULL ENABLE,
    HEADER_SEQ     NUMBER(3,0) NOT NULL ENABLE,
    SEQNO          NUMBER(5,0) NOT NULL ENABLE,
    CHAR_TYPE_CD   CHAR(8) NOT NULL ENABLE,
    CHAR_VAL       CHAR(16) DEFAULT ' ' NOT NULL ENABLE,
    ADHOC_CHAR_VAL VARCHAR2(254) DEFAULT ' ' NOT NULL ENABLE,
    CHAR_VAL_FK1   VARCHAR2(50) DEFAULT ' ' NOT NULL ENABLE,
    CHAR_VAL_FK2   VARCHAR2(50) DEFAULT ' ' NOT NULL ENABLE,
    CHAR_VAL_FK3   VARCHAR2(50) DEFAULT ' ' NOT NULL ENABLE,
    CHAR_VAL_FK4   VARCHAR2(50) DEFAULT ' ' NOT NULL ENABLE,
    CHAR_VAL_FK5   VARCHAR2(50) DEFAULT ' ' NOT NULL ENABLE,
    VERSION        NUMBER(5,0) DEFAULT 1 NOT NULL ENABLE,
 CONSTRAINT CI_BSEG_CL_CHAR_FK FOREIGN KEY(BSEG_ID) REFERENCES CI_BSEG
ON DELETE CASCADE
)
PARTITION BY REFERENCE (CI_BSEG_CL_CHAR_FK)
ENABLE ROW MOVEMENT
/
```

INDEX

```
CREATE UNIQUE INDEX XT056P0 ON CI_BSEG_CL_CHAR ( BSEG_ID, HEADER_SEQ,
SEQNO, CHAR_TYPE_CD ) TABLESPACE CM_XT048_IND
GLOBAL PARTITION BY RANGE (BSEG_ID)
(
PARTITION P1 VALUES LESS THAN ( '124999999999' ),
PARTITION P2 VALUES LESS THAN ( '249999999999' ),
PARTITION P3 VALUES LESS THAN ( '374999999999' ),
PARTITION P4 VALUES LESS THAN ( '499999999999' ),
PARTITION P5 VALUES LESS THAN ( '624999999999' ),
PARTITION P6 VALUES LESS THAN ( '749999999999' ),
PARTITION P7 VALUES LESS THAN ( '874999999999' ),
PARTITION P8 VALUES LESS THAN ( MAXVALUE )
)
COMPRESS ADVANCED LOW
/

ALTER TABLE CI_BSEG_CL_CHAR ADD CONSTRAINT XT056P0 PRIMARY
KEY(BSEG_ID, HEADER_SEQ, SEQNO, CHAR_TYPE_CD) USING INDEX
/
```

Child Table: CI_BSEG_EXCP

```
CREATE TABLE CI_BSEG_EXCP
  (
    BSEG_ID         CHAR(12) NOT NULL ENABLE,
    MESSAGE_CAT_NBR NUMBER(5,0) DEFAULT 0 NOT NULL ENABLE,
    MESSAGE_NBR     NUMBER(5,0) DEFAULT 0 NOT NULL ENABLE,
    BSEG_EXCP_FLG   CHAR(2) DEFAULT ' ' NOT NULL ENABLE,
    EXP_MSG         VARCHAR2(254) DEFAULT ' ' NOT NULL ENABLE,
    MESSAGE_PARM1   VARCHAR2(2000) DEFAULT ' ' NOT NULL ENABLE,
    MESSAGE_PARM2   VARCHAR2(2000) DEFAULT ' ' NOT NULL ENABLE,
    MESSAGE_PARM3   VARCHAR2(2000) DEFAULT ' ' NOT NULL ENABLE,
    MESSAGE_PARM4   VARCHAR2(2000) DEFAULT ' ' NOT NULL ENABLE,
    MESSAGE_PARM5   VARCHAR2(2000) DEFAULT ' ' NOT NULL ENABLE,
    MESSAGE_PARM6   VARCHAR2(2000) DEFAULT ' ' NOT NULL ENABLE,
    MESSAGE_PARM7   VARCHAR2(2000) DEFAULT ' ' NOT NULL ENABLE,
    MESSAGE_PARM8   VARCHAR2(2000) DEFAULT ' ' NOT NULL ENABLE,
    MESSAGE_PARM9   VARCHAR2(2000) DEFAULT ' ' NOT NULL ENABLE,
    CALL_SEQ        VARCHAR2(254) DEFAULT ' ' NOT NULL ENABLE,
    USER_ID         CHAR(8) DEFAULT ' ' NOT NULL ENABLE,
    CRE_DTTM DATE,
    REVIEW_COMP    CHAR(1) DEFAULT ' ' NOT NULL ENABLE,
    REVIEW_USER_ID CHAR(8) DEFAULT ' ' NOT NULL ENABLE,
    REVIEW_DT DATE,
    COMMENTS VARCHAR2(254) DEFAULT ' ' NOT NULL ENABLE,
    VERSION  NUMBER(5,0) DEFAULT 1 NOT NULL ENABLE,
  CONSTRAINT CI_BSEG_EXCP_FK FOREIGN KEY(BSEG_ID) REFERENCES CI_BSEG
ON DELETE CASCADE
)
PARTITION BY REFERENCE (CI_BSEG_EXCP_FK)
ENABLE ROW MOVEMENT
/
```

INDEX

```
CREATE UNIQUE INDEX XT051P0 ON CI_BSEG_EXCP ( BSEG_ID ) TABLESPACE
CM_XT048_IND
GLOBAL PARTITION BY RANGE (BSEG_ID)
```

```
(
PARTITION P1 VALUES LESS THAN ( '124999999999' ),
PARTITION P2 VALUES LESS THAN ( '249999999999' ),
PARTITION P3 VALUES LESS THAN ( '374999999999' ),
PARTITION P4 VALUES LESS THAN ( '499999999999' ),
PARTITION P5 VALUES LESS THAN ( '624999999999' ),
PARTITION P6 VALUES LESS THAN ( '749999999999' ),
PARTITION P7 VALUES LESS THAN ( '874999999999' ),
PARTITION P8 VALUES LESS THAN ( MAXVALUE )
)
COMPRESS ADVANCED LOW
/

ALTER TABLE CI_BSEG_EXCP ADD CONSTRAINT XT051P0 PRIMARY KEY(BSEG_ID)
USING INDEX
/
```

Child Table: CI_BSEG_MSG

```
CREATE TABLE CI_BSEG_MSG
  (
    BSEG_ID     CHAR(12) NOT NULL ENABLE,
    BILL_MSG_CD CHAR(4) NOT NULL ENABLE,
    VERSION     NUMBER(5,0) DEFAULT 1 NOT NULL ENABLE,
 CONSTRAINT CI_BSEG_MSG_FK FOREIGN KEY(BSEG_ID) REFERENCES CI_BSEG ON
DELETE CASCADE
)
PARTITION BY REFERENCE (CI_BSEG_MSG_FK)
ENABLE ROW MOVEMENT
/
```

INDEX

```
CREATE UNIQUE INDEX XT080P0 ON CI_BSEG_MSG ( BSEG_ID, BILL_MSG_CD )
TABLESPACE CM_XT048_IND
GLOBAL PARTITION BY RANGE (BSEG_ID)
(
PARTITION P1 VALUES LESS THAN ( '124999999999' ),
PARTITION P2 VALUES LESS THAN ( '249999999999' ),
PARTITION P3 VALUES LESS THAN ( '374999999999' ),
PARTITION P4 VALUES LESS THAN ( '499999999999' ),
PARTITION P5 VALUES LESS THAN ( '624999999999' ),
PARTITION P6 VALUES LESS THAN ( '749999999999' ),
PARTITION P7 VALUES LESS THAN ( '874999999999' ),
PARTITION P8 VALUES LESS THAN ( MAXVALUE )
)
COMPRESS ADVANCED LOW
/

ALTER TABLE CI_BSEG_MSG ADD CONSTRAINT XT080P0 PRIMARY KEY(BSEG_ID,
BILL_MSG_CD) USING INDEX
/
```

Child Table: CI_BSEG_READ

```
CREATE TABLE CI_BSEG_READ
  (
    BSEG_ID           CHAR(12) NOT NULL ENABLE,
    SP_ID             CHAR(10) NOT NULL ENABLE,
    REG_CONST         NUMBER(12,6) DEFAULT 0 NOT NULL ENABLE,
```

```
    SEQNO             NUMBER(5,0) NOT NULL ENABLE,
    USAGE_FLG         CHAR(2) DEFAULT ' ' NOT NULL ENABLE,
    USE_PCT           NUMBER(3,0) DEFAULT 0 NOT NULL ENABLE,
    HOW_TO_USE_FLG    CHAR(2) DEFAULT ' ' NOT NULL ENABLE,
    MSR_PEAK_QTY_SW   CHAR(1) DEFAULT ' ' NOT NULL ENABLE,
    UOM_CD            CHAR(4) DEFAULT ' ' NOT NULL ENABLE,
    TOU_CD            CHAR(8) DEFAULT ' ' NOT NULL ENABLE,
    START_REG_READ_ID CHAR(12) DEFAULT ' ' NOT NULL ENABLE,
    START_READ_DTTM DATE NOT NULL ENABLE,
    START_REG_READING NUMBER(15,6) DEFAULT 0 NOT NULL ENABLE,
    END_REG_READ_ID   CHAR(12) DEFAULT ' ' NOT NULL ENABLE,
    END_READ_DTTM DATE NOT NULL ENABLE,
    END_REG_READING NUMBER(15,6) DEFAULT 0 NOT NULL ENABLE,
    MSR_QTY         NUMBER(18,6) DEFAULT 0 NOT NULL ENABLE,
    FINAL_UOM_CD    CHAR(4) DEFAULT ' ' NOT NULL ENABLE,
    FINAL_TOU_CD    CHAR(8) DEFAULT ' ' NOT NULL ENABLE,
    FINAL_REG_QTY   NUMBER(18,6) DEFAULT 0 NOT NULL ENABLE,
    VERSION         NUMBER(5,0) DEFAULT 1 NOT NULL ENABLE,
    SQI_CD          CHAR(8) DEFAULT ' ' NOT NULL ENABLE,
    FINAL_SQI_CD    CHAR(8) DEFAULT ' ' NOT NULL ENABLE,
 CONSTRAINT CI_BSEG_READ_FK FOREIGN KEY(BSEG_ID) REFERENCES CI_BSEG ON
DELETE CASCADE
)
PARTITION BY REFERENCE (CI_BSEG_READ_FK)
ENABLE ROW MOVEMENT
/
```

INDEX

```
CREATE UNIQUE INDEX XT054P0 ON CI_BSEG_READ ( BSEG_ID, SP_ID, SEQNO )
TABLESPACE CM_XT048_IND
GLOBAL PARTITION BY RANGE (BSEG_ID)
(
PARTITION P1 VALUES LESS THAN ( '124999999999' ),
PARTITION P2 VALUES LESS THAN ( '249999999999' ),
PARTITION P3 VALUES LESS THAN ( '374999999999' ),
PARTITION P4 VALUES LESS THAN ( '499999999999' ),
PARTITION P5 VALUES LESS THAN ( '624999999999' ),
PARTITION P6 VALUES LESS THAN ( '749999999999' ),
PARTITION P7 VALUES LESS THAN ( '874999999999' ),
PARTITION P8 VALUES LESS THAN ( MAXVALUE )
)
COMPRESS ADVANCED LOW
/

ALTER TABLE CI_BSEG_READ ADD CONSTRAINT XT054P0 PRIMARY KEY(BSEG_ID,
SP_ID, SEQNO) USING INDEX
/
CREATE INDEX XT054S1 ON CI_BSEG_READ ( SP_ID ) TABLESPACE CM_XT048_IND
/
CREATE INDEX XT054S2 ON CI_BSEG_READ ( START_REG_READ_ID ) TABLESPACE
CM_XT048_IND
/
CREATE INDEX XT054S3 ON CI_BSEG_READ ( END_REG_READ_ID ) TABLESPACE
CM_XT048_IND
/
```

Child Table: CI_BSEG_SQ

```
CREATE TABLE CI_BSEG_SQ
  (
```

```
    BSEG_ID CHAR(12) NOT NULL ENABLE,
    UOM_CD  CHAR(4) NOT NULL ENABLE,
    TOU_CD  CHAR(8) NOT NULL ENABLE,
    SQI_CD  CHAR(8) NOT NULL ENABLE,
    INIT_SQ NUMBER(18,6) DEFAULT 0 NOT NULL ENABLE,
    BILL_SQ NUMBER(18,6) DEFAULT 0 NOT NULL ENABLE,
    VERSION NUMBER(5,0) DEFAULT 1 NOT NULL ENABLE,
 CONSTRAINT CI_BSEG_SQ_FK FOREIGN KEY(BSEG_ID) REFERENCES CI_BSEG ON
DELETE CASCADE
)
PARTITION BY REFERENCE (CI_BSEG_SQ_FK)
ENABLE ROW MOVEMENT
/
```

INDEX

```
CREATE UNIQUE INDEX XT055P0 ON CI_BSEG_SQ ( BSEG_ID, UOM_CD, TOU_CD,
SQI_CD ) TABLESPACE CM_XT048_IND
GLOBAL PARTITION BY RANGE (BSEG_ID)
(
PARTITION P1 VALUES LESS THAN ( '124999999999' ),
PARTITION P2 VALUES LESS THAN ( '249999999999' ),
PARTITION P3 VALUES LESS THAN ( '374999999999' ),
PARTITION P4 VALUES LESS THAN ( '499999999999' ),
PARTITION P5 VALUES LESS THAN ( '624999999999' ),
PARTITION P6 VALUES LESS THAN ( '749999999999' ),
PARTITION P7 VALUES LESS THAN ( '874999999999' ),
PARTITION P8 VALUES LESS THAN ( MAXVALUE )
)
COMPRESS ADVANCED LOW
/
ALTER TABLE CI_BSEG_SQ ADD CONSTRAINT XT055P0 PRIMARY KEY(BSEG_ID,
UOM_CD, TOU_CD, SQI_CD) USING INDEX
/
```

Child Table: CI_BSEG_ITEM

```
CREATE TABLE CI_BSEG_ITEM
  (
    BSEG_ID      CHAR(12) NOT NULL ENABLE,
    SEQNO        NUMBER(5,0) NOT NULL ENABLE,
    ITEM_TYPE_CD VARCHAR(30) DEFAULT ' ' NOT NULL ENABLE,
    ITEM_ID      CHAR(10) DEFAULT ' ' NOT NULL ENABLE,
    START_DT DATE NOT NULL ENABLE,
    END_DT DATE NOT NULL ENABLE,
    ITEM_CNT NUMBER(11,2) DEFAULT 0 NOT NULL ENABLE,
    UOM_CD   CHAR(4) DEFAULT ' ' NOT NULL ENABLE,
    SVC_QTY  NUMBER(18,6) DEFAULT 0 NOT NULL ENABLE,
    VERSION  NUMBER(5,0) DEFAULT 1 NOT NULL ENABLE,
CONSTRAINT CI_BSEG_ITEM_FK FOREIGN KEY(BSEG_ID) REFERENCES CI_BSEG ON
DELETE CASCADE
)
PARTITION BY REFERENCE (CI_BSEG_ITEM_FK)
ENABLE ROW MOVEMENT
/
```

INDEX

```
CREATE UNIQUE INDEX XT053P0 ON CI_BSEG_ITEM ( BSEG_ID, SEQNO )
TABLESPACE CM_XT048_IND
GLOBAL PARTITION BY RANGE (BSEG_ID)
```

```
(
PARTITION P1 VALUES LESS THAN ( '124999999999' ),
PARTITION P2 VALUES LESS THAN ( '249999999999' ),
PARTITION P3 VALUES LESS THAN ( '374999999999' ),
PARTITION P4 VALUES LESS THAN ( '499999999999' ),
PARTITION P5 VALUES LESS THAN ( '624999999999' ),
PARTITION P6 VALUES LESS THAN ( '749999999999' ),
PARTITION P7 VALUES LESS THAN ( '874999999999' ),
PARTITION P8 VALUES LESS THAN ( MAXVALUE )
)
COMPRESS ADVANCED LOW
/
ALTER TABLE CI_BSEG_ITEM ADD CONSTRAINT XT053P0 PRIMARY KEY(BSEG_ID,
SEQNO) USING INDEX
/
```

Appendix E

Sample Scripts for Customer Contact Enhancement

In this release of Oracle Utilities Customer Care and Billing, there are enhancements that change the behavior of some objects. Upgrading customers may wish to update existing data to align with how newly created data will behave.

The following sample scripts are provided to help upgrading clients. Customers are advised to review, edit, and test these sample script to ensure they meet business and data requirement.

All scripts are delivered with logic to run in threads. In the format that the script is delivered, these scripts will not process any records. The scripts must be edited to set specific key ranges so that it can be run in parallel or process all records.

This appendix consists of the following:

Updating Customer Contact Account and Premise

In this release of Oracle Utilities Customer Care and Billing, the customer contact capability has been enhanced so that customer contacts can be linked to a person, account and/or premise. This section contains sample SQL that upgrading implementations can use to update the account ID and premise ID fields on existing customer contacts.

All Oracle Utilities Customer Care and Billing system processes that create customer contacts have been enhanced to populate account and/or premise. Not all of these areas retain a link to the customer contact created and the SQL provided is just a sample.

These scripts update Customer Contact Account and Premise from the following sources:

- Customer Contact Characteristics
- Collection Events
- Severance Events
- Write off Events
- Overdue Process Logs

The other Oracle Utilities Customer Care and Billing process enhanced are algorithm types delivered for the below plug-in spots. Examine your organization's customer contacts to determine if you should update customer contacts created from the following:

- FA Remark Algorithm
- Meter read Remark Algorithm
- Case Type Enter Status Algorithm
- Customer Class Order Completion
- SA Type SA Stop
- SA Type - SA Activation
- Service Credit Membership Type - Membership Activation
- Service Credit Membership Type - Membership Creation
- Campaign - Order Completion
- Lead Event Type (BO) - Lead Event Completion
- Notification Type (BO) - Create Notification

Since existing customer contact records are all person-based, all account updates should occur before premise updates to ensure the premise is associated with the account.

These are intended to be run in the following order:

1. Update Account ID from characteristic.

   Note the following about this SQL:

   - This cannot handle two characteristic types that are both linked to Acct FK Ref with different char values. If this exists such as FROM_ACCT and TO_ACCT with different values, this will not update.
   - Limited to accounts linked to CC person to not violate new CC validations.

- Change FK Ref if your implementation has introduced new FK Ref values for ACCT.

```
UPDATE CI_CC X
SET X.ACCT_ID =
  (SELECT DISTINCT(a.CHAR_VAL_FK1)
  FROM CI_CC_CHAR A,
    CI_CHAR_TYPE B,
    CI_CC C
  WHERE a.CHAR_TYPE_CD = B.CHAR_TYPE_CD
  AND A.CC_ID          = C.CC_ID
  AND B.FK_REF_CD     <> ' '
  AND b.fk_ref_cd      = 'ACCT'
  AND a.CC_ID          = X.CC_ID
  AND EXISTS
    (SELECT'x'
    FROM CI_ACCT_PER D
    WHERE D.PER_ID = C.PER_ID
    AND D.ACCT_ID  = a.CHAR_VAL_FK1
    )
  )
WHERE CC_ID =
  (SELECT A.CC_ID
  FROM CI_CC_CHAR A,
    CI_CHAR_TYPE B,
    CI_CC C
  WHERE a.CHAR_TYPE_CD = B.CHAR_TYPE_CD
  AND A.CC_ID          = C.CC_ID
  AND B.FK_REF_CD     <> ' '
  AND b.fk_ref_cd      = 'ACCT'
  AND a.CC_ID          = X.CC_ID
  AND EXISTS
    (SELECT'x'
    FROM CI_ACCT_PER D
    WHERE D.PER_ID = C.PER_ID
    AND D.ACCT_ID  = a.CHAR_VAL_FK1
    )
  )
AND 1 =
  (SELECT COUNT (DISTINCT a.CHAR_VAL_FK1)
  FROM CI_CC_CHAR A,
    CI_CHAR_TYPE B,
    CI_CC C
  WHERE a.CHAR_TYPE_CD = B.CHAR_TYPE_CD
  AND A.CC_ID          = C.CC_ID
  AND B.FK_REF_CD     <> ' '
  AND b.fk_ref_cd      = 'ACCT'
  AND a.CC_ID          = X.CC_ID
  )
AND X.ACCT_ID IS NULL
--AND X.CC_ID BETWEEN '0000000000' AND '9999999999';
AND X.CC_ID BETWEEN '0000000000' AND '0000000000';
```

2. Update Account ID from Coll event.

```
UPDATE CI_CC X
SET X.ACCT_ID =
  (SELECT a.acct_id
  FROM CI_COLL_PROC A,
    CI_COLL_EVT_CC C
  WHERE a.coll_proc_id = c.coll_proc_ID
  AND c.cc_id          = x.cc_id
```

```
  )
WHERE X.CC_ID IN
  (SELECT CC_ID FROM CI_COLL_EVT_CC
  )
AND X.ACCT_ID IS NULL
--AND X.CC_ID BETWEEN '0000000000' AND '9999999999';
AND X.CC_ID BETWEEN '0000000000' AND '0000000000';
```

3. Update Account ID from Sev event.

```
UPDATE CI_CC X
SET X.ACCT_ID =
  (SELECT a.acct_id
  FROM CI_SA A,
    CI_SEV_PROC B,
    CI_SEV_EVT_CC C
  WHERE a.SA_ID     = B.SA_ID
  AND B.sev_proc_id = c.sev_proc_id
  AND c.cc_id       = x.cc_id
  )
WHERE X.CC_ID IN
  (SELECT CC_ID FROM CI_SEV_EVT_CC
  )
AND X.ACCT_ID IS NULL
--AND X.CC_ID BETWEEN '0000000000' AND '9999999999';
AND X.CC_ID BETWEEN '0000000000' AND '0000000000';
```

4. Update Premise ID from Sev event.

   Using premise linked to SP linked to FA linked to same Sev Proc First.

```
UPDATE CI_CC X
SET X.PREM_ID =
  (SELECT DISTINCT h.prem_id
  FROM CI_SEV_EVT_CC D,
    CI_SEV_EVT_FA E,
    CI_FA G,
    CI_SP H
  WHERE D.SEV_PROC_ID = E.SEV_PROC_ID
  AND e.fa_id         = g.fa_id
  AND g.sp_id         = h.sp_id
  AND d.cc_id         = x.cc_id
  )
WHERE X.CC_ID IN
  (SELECT CC_ID
  FROM CI_SEV_EVT_CC D,
    CI_SEV_EVT_FA E
  WHERE D.SEV_PROC_ID = E.SEV_PROC_ID
  )
AND X.PREM_ID IS NULL
--AND X.CC_ID BETWEEN '0000000000' AND '9999999999';
AND X.CC_ID BETWEEN '0000000000' AND '0000000000';
```

5. Update Premise ID from Sev event.

   Using premise linked to SA via Char Premise second.

   Note: There is no SQL that will try to find a premise via SASP. The likelihood of more than one premise is too high.

```
UPDATE CI_CC X
SET X.PREM_ID =
  (SELECT NVL(a.char_prem_id,NULL)
```

```
  FROM CI_SA A,
    CI_SEV_PROC B,
    CI_SEV_EVT_CC C
  WHERE a.SA_ID     = B.SA_ID
  AND B.sev_proc_id = c.sev_proc_id
  AND A.CHAR_PREM_ID <> ' '
  AND c.cc_id       = x.cc_id
  )
WHERE X.CC_ID IN
  (SELECT CC_ID FROM CI_SEV_EVT_CC
  )
AND X.PREM_ID IS NULL
--AND X.CC_ID BETWEEN '0000000000' AND '9999999999';
AND X.CC_ID BETWEEN '0000000000' AND '0000000000';
```

6. Update Account ID from WO event.

```
UPDATE CI_CC X
SET X.ACCT_ID =
  (SELECT a.acct_id
  FROM CI_WO_PROC A,
    CI_WO_EVT_CC C
  WHERE a.wo_proc_id = c.wo_proc_ID
  AND c.cc_id        = x.cc_id
  )
WHERE X.CC_ID IN
  (SELECT CC_ID FROM CI_WO_EVT_CC
  )
AND X.ACCT_ID IS NULL
--AND X.CC_ID BETWEEN '0000000000' AND '9999999999';
AND X.CC_ID BETWEEN '0000000000' AND '0000000000';
```

7. Update Account ID from Overdue and Cut Processes via the Overdue Process Logs.

   NOTE: You must hardcode the Char Type Code your implementation has introduced.

```
UPDATE CI_CC X
SET X.ACCT_ID =
  (SELECT b.acct_id
  FROM CI_OD_PROC_LOG A,
    CI_OD_PROC B
  WHERE a.char_type_cd = 'CCID'
  AND a.OD_PROC_ID     = B.OD_PROC_ID
  AND trim(a.char_val_fk1)   = x.cc_id
  )
WHERE X.CC_ID IN
  (SELECT a.char_val_fk1
  FROM CI_OD_PROC_LOG A,
    CI_OD_PROC B
  WHERE a.char_type_cd = 'CCID'
  AND a.OD_PROC_ID     = B.OD_PROC_ID
  )
AND X.ACCT_ID IS NULL
--AND X.CC_ID BETWEEN '0000000000' AND '9999999999';
AND X.CC_ID BETWEEN '0000000000' AND '0000000000';
```

8. Update Premise ID from CC Char.

   Some notes about this SQL:

- This cannot handle two characteristic types that are both linked to Prem FK Ref with different char values. If this exists such as OLD_PREM and NEW_PREM with different values, this will not update.
- Limited to premise associated with account if acct is populated on the CC to not violate new CC validations.
- Change FK Ref if your implementation has introduced new FK Ref values for PREM.

```
UPDATE CI_CC X
SET X.PREM_ID =
  (SELECT DISTINCT(a.CHAR_VAL_FK1)
  FROM CI_CC_CHAR A,
    CI_CHAR_TYPE B,
    CI_CC C
  WHERE a.CHAR_TYPE_CD = B.CHAR_TYPE_CD
  AND A.CC_ID          = C.CC_ID
  AND B.FK_REF_CD     <> ' '
  AND b.fk_ref_cd      = 'PREM'
  AND a.CC_ID          = X.CC_ID
  AND (C.ACCT_ID      IS NULL
  OR C.ACCT_ID        IS NOT NULL
  AND a.CHAR_VAL_FK1  IN
    (SELECT E.char_prem_id FROM CI_SA E WHERE E.ACCT_ID = C.ACCT_ID
    UNION
    SELECT H.PREM_ID
    FROM CI_SA F,
      CI_SA_SP G,
      CI_SP H
    WHERE f.ACCT_ID = C.ACCT_ID
    AND F.SA_ID     = G.SA_ID
    AND G.SP_ID     = H.SP_ID
    ) )
  )
WHERE CC_ID =
  (SELECT A.CC_ID
  FROM CI_CC_CHAR A,
    CI_CHAR_TYPE B,
    CI_CC C
  WHERE a.CHAR_TYPE_CD = B.CHAR_TYPE_CD
  AND A.CC_ID          = C.CC_ID
  AND B.FK_REF_CD     <> ' '
  AND b.fk_ref_cd      = 'PREM'
  AND a.CC_ID          = X.CC_ID
  AND (C.ACCT_ID      IS NULL
  OR C.ACCT_ID        IS NOT NULL
  AND a.CHAR_VAL_FK1  IN
    (SELECT E.char_prem_id FROM CI_SA E WHERE E.ACCT_ID = C.ACCT_ID
    UNION
    SELECT H.PREM_ID
    FROM CI_SA F,
      CI_SA_SP G,
      CI_SP H
    WHERE f.ACCT_ID = C.ACCT_ID
    AND F.SA_ID     = G.SA_ID
    AND G.SP_ID     = H.SP_ID
    ) )
  )
AND 1 =
  (SELECT COUNT (DISTINCT a.CHAR_VAL_FK1)
  FROM CI_CC_CHAR A,
```

```
    CI_CHAR_TYPE B,
    CI_CC C
  WHERE a.CHAR_TYPE_CD = B.CHAR_TYPE_CD
  AND A.CC_ID          = C.CC_ID
  AND B.FK_REF_CD     <> ' '
  AND b.fk_ref_cd      = 'PREM'
  AND a.CC_ID          = X.CC_ID
  )
AND X.PREM_ID IS NULL
--AND X.CC_ID BETWEEN '0000000000' AND '9999999999';
AND X.CC_ID BETWEEN '0000000000' AND '0000000000';
```

Updating Preferred Contact Method on Legacy Values

In this release of Oracle Utilities Customer Care and Billing, a feature is introduced that some system processes use to determine if person phone and email or person contacts is being used.

If moving to using person contacts, some contact method values on case and customer contact are suppressed and existing records show as <invalid value>.

These scripts update the preferred contact method on cases and customer contacts from legacy values to person contact.

1. Update Case Preferred Contact Method from Email to Primary Email Person Contact:

```
UPDATE CI_CASE X
SET X.CONTACT_METH_FLG = 'C1PC',
  X.C1_CONTACT_ID      =
  (SELECT B.C1_CONTACT_ID
  FROM CI_CASE A,
    CISADM.C1_PER_CONTDET B,
    CISADM.C1_COMM_RTE_TYPE C
  WHERE CONTACT_METH_FLG  = 'EM'
  AND A.CONTACT_PER_ID    = B.PER_ID
  AND B.COMM_RTE_TYPE_CD  = C.COMM_RTE_TYPE_CD
  AND B.CND_PRIMARY_FLG   = 'C1YS'
  AND C.COMM_RTE_METH_FLG = 'EMAIL'
  AND X.CASE_ID           = A.CASE_ID
  )
WHERE X.CASE_ID IN
  (SELECT A.CASE_ID
  FROM CI_CASE A,
    CISADM.C1_PER_CONTDET B,
    CISADM.C1_COMM_RTE_TYPE C
  WHERE CONTACT_METH_FLG  = 'EM'
  AND A.CONTACT_PER_ID    = B.PER_ID
  AND B.COMM_RTE_TYPE_CD  = C.COMM_RTE_TYPE_CD
  AND B.CND_PRIMARY_FLG   = 'C1YS'
  AND C.COMM_RTE_METH_FLG = 'EMAIL'
  )
--AND X.CASE_ID BETWEEN '0000000000' AND '9999999999';
AND X.CASE_ID BETWEEN '0000000000' AND '0000000000';
```

2. Update Case Preferred Contact Method from Fax to Primary Fax Person Contact:

```
UPDATE CI_CASE X
SET X.CONTACT_METH_FLG = 'C1PC',
  X.C1_CONTACT_ID      =
```

```
  (SELECT B.C1_CONTACT_ID
  FROM CI_CASE A,
    CISADM.C1_PER_CONTDET B,
    CISADM.C1_COMM_RTE_TYPE C
  WHERE CONTACT_METH_FLG  = 'FAX'
  AND A.CONTACT_PER_ID    = B.PER_ID
  AND B.COMM_RTE_TYPE_CD  = C.COMM_RTE_TYPE_CD
  AND B.CND_PRIMARY_FLG   = 'C1YS'
  AND C.COMM_RTE_METH_FLG = 'FAX'
  AND X.CASE_ID           = A.CASE_ID
  )
WHERE X.CASE_ID IN
  (SELECT A.CASE_ID
  FROM CI_CASE A,
    CISADM.C1_PER_CONTDET B,
    CISADM.C1_COMM_RTE_TYPE C
  WHERE CONTACT_METH_FLG  = 'FAX'
  AND A.CONTACT_PER_ID    = B.PER_ID
  AND B.COMM_RTE_TYPE_CD  = C.COMM_RTE_TYPE_CD
  AND B.CND_PRIMARY_FLG   = 'C1YS'
  AND C.COMM_RTE_METH_FLG = 'FAX'
  )
--AND X.CASE_ID BETWEEN '0000000000' AND '9999999999';
AND X.CASE_ID BETWEEN '0000000000' AND '0000000000';
```

3. Update Case Preferred Contact Method from Phone to Primary Phone Person Contact:

```
UPDATE CI_CASE X
SET X.CONTACT_METH_FLG = 'C1PC',
  X.C1_CONTACT_ID      =
  (SELECT B.C1_CONTACT_ID
  FROM CI_CASE A,
    CISADM.C1_PER_CONTDET B,
    CISADM.C1_COMM_RTE_TYPE C
  WHERE CONTACT_METH_FLG  = 'PH'
  AND A.CONTACT_PER_ID    = B.PER_ID
  AND B.COMM_RTE_TYPE_CD  = C.COMM_RTE_TYPE_CD
  AND B.CND_PRIMARY_FLG   = 'C1YS'
  AND C.COMM_RTE_METH_FLG = 'PHONE'
  AND X.CASE_ID           = A.CASE_ID
  )
WHERE X.CASE_ID IN
  (SELECT A.CASE_ID
  FROM CI_CASE A,
    CISADM.C1_PER_CONTDET B,
    CISADM.C1_COMM_RTE_TYPE C
  WHERE CONTACT_METH_FLG  = 'PH'
  AND A.CONTACT_PER_ID    = B.PER_ID
  AND B.COMM_RTE_TYPE_CD  = C.COMM_RTE_TYPE_CD
  AND B.CND_PRIMARY_FLG   = 'C1YS'
  AND C.COMM_RTE_METH_FLG = 'PHONE'
  )
--AND X.CASE_ID BETWEEN '0000000000' AND '9999999999';
AND X.CASE_ID BETWEEN '0000000000' AND '0000000000';
```

4. Update Customer Contact Preferred Contact Method from Email to Primary Email Person Contact:

```
UPDATE CI_CC X
SET X.CONTACT_METH_FLG = 'C1PC',
  X.C1_CONTACT_ID      =
```

```
  (SELECT B.C1_CONTACT_ID
  FROM CI_CC A,
    CISADM.C1_PER_CONTDET B,
    CISADM.C1_COMM_RTE_TYPE C
  WHERE CONTACT_METH_FLG  = 'EM'
  AND A.PER_ID    = B.PER_ID
  AND B.COMM_RTE_TYPE_CD  = C.COMM_RTE_TYPE_CD
  AND B.CND_PRIMARY_FLG   = 'C1YS'
  AND C.COMM_RTE_METH_FLG = 'EMAIL'
  AND X.CC_ID           = A.CC_ID
  )
WHERE X.CC_ID IN
  (SELECT A.CC_ID
  FROM CI_CC A,
    CISADM.C1_PER_CONTDET B,
    CISADM.C1_COMM_RTE_TYPE C
  WHERE CONTACT_METH_FLG  = 'EM'
  AND A.PER_ID    = B.PER_ID
  AND B.COMM_RTE_TYPE_CD  = C.COMM_RTE_TYPE_CD
  AND B.CND_PRIMARY_FLG   = 'C1YS'
  AND C.COMM_RTE_METH_FLG = 'EMAIL'
  )
--AND X.CC_ID BETWEEN '0000000000' AND '9999999999';
AND X.CC_ID BETWEEN '0000000000' AND '0000000000';
```

5. Update Customer Contact Preferred Contact Method from Fax to Primary Fax Person Contact:

```
UPDATE CI_CC X
SET X.CONTACT_METH_FLG = 'C1PC',
  X.C1_CONTACT_ID      =
  (SELECT B.C1_CONTACT_ID
  FROM CI_CC A,
    CISADM.C1_PER_CONTDET B,
    CISADM.C1_COMM_RTE_TYPE C
  WHERE CONTACT_METH_FLG  = 'FAX'
  AND A.PER_ID    = B.PER_ID
  AND B.COMM_RTE_TYPE_CD  = C.COMM_RTE_TYPE_CD
  AND B.CND_PRIMARY_FLG   = 'C1YS'
  AND C.COMM_RTE_METH_FLG = 'FAX'
  AND X.CC_ID           = A.CC_ID
  )
WHERE X.CC_ID IN
  (SELECT A.CC_ID
  FROM CI_CC A,
    CISADM.C1_PER_CONTDET B,
    CISADM.C1_COMM_RTE_TYPE C
  WHERE CONTACT_METH_FLG  = 'FAX'
  AND A.PER_ID    = B.PER_ID
  AND B.COMM_RTE_TYPE_CD  = C.COMM_RTE_TYPE_CD
  AND B.CND_PRIMARY_FLG   = 'C1YS'
  AND C.COMM_RTE_METH_FLG = 'FAX'
  )
--AND X.CC_ID BETWEEN '0000000000' AND '9999999999';
AND X.CC_ID BETWEEN '0000000000' AND '0000000000';
```

6. Update Customer Contact Preferred Contact Method from Phone to Primary Phone Person Contact:

```
UPDATE CI_CC X
SET X.CONTACT_METH_FLG = 'C1PC',
  X.C1_CONTACT_ID      =
```

```
  (SELECT B.C1_CONTACT_ID
  FROM CI_CC A,
    CISADM.C1_PER_CONTDET B,
    CISADM.C1_COMM_RTE_TYPE C
  WHERE CONTACT_METH_FLG  = 'PH'
  AND A.PER_ID    = B.PER_ID
  AND B.COMM_RTE_TYPE_CD  = C.COMM_RTE_TYPE_CD
  AND B.CND_PRIMARY_FLG   = 'C1YS'
  AND C.COMM_RTE_METH_FLG = 'PHONE'
  AND X.CC_ID           = A.CC_ID
  )
WHERE X.CC_ID IN
  (SELECT A.CC_ID
  FROM CI_CC A,
    CISADM.C1_PER_CONTDET B,
    CISADM.C1_COMM_RTE_TYPE C
  WHERE CONTACT_METH_FLG  = 'PH'
  AND A.PER_ID    = B.PER_ID
  AND B.COMM_RTE_TYPE_CD  = C.COMM_RTE_TYPE_CD
  AND B.CND_PRIMARY_FLG   = 'C1YS'
  AND C.COMM_RTE_METH_FLG = 'PHONE'
  )
--AND X.CC_ID BETWEEN '0000000000' AND '9999999999';
AND X.CC_ID BETWEEN '0000000000' AND '0000000000';
```

7. The following shows remaining cases that need to be investigated manually:

```
SELECT *
FROM CI_CASE
WHERE CONTACT_METH_FLG <> ' '
 AND CONTACT_METH_FLG NOT IN ('N/A','POST','C1PC');
```

8. The following shows remaining Customer Contacts that need to be investigated manually:

```
SELECT *
FROM CI_CC
WHERE CONTACT_METH_FLG <> ' '
 AND CONTACT_METH_FLG NOT IN ('N/A','POST','C1PC');
```

Appendix F

Upgrades to the Oracle Utilities Customer Care and Billing 2.7.0.1.0 Database

This document describes the database upgrade process for the Oracle Utilities Customer Care and Billing V2.7.0.1.0. It highlights changes made to the administrative tables and how those changes should be applied to the data in order for your current database to work with the V2.7.0.1.0 application, and to preserve the business logic implemented in the previous version of the application. The changes that do not require data upgrade are not described in this section of the document. The tasks that need to be performed after running the upgrade scripts are included.

The added functionality of V2.7.0.1.0 is not the scope of this documentation. The upgrade scripts do not turn on the newly added functionality by default. For new functionality, refer the V2.7.0.1.0 User Guides. In the last section of this document, you will find a list of the new tables that are added in V2.7.0.1.0.

This section includes:

Schema Changes

Column Format Changes

The following column is modified in this Oracle Utilities Customer Care and Billing release.

Table_Name	Column_Name	From	To
CI_CC_TYPE	CUSTOM_ARCH_ELIG_CRIT_FLG	NULL	NOT NULL

New System Data

This section lists the new system data that are added for business process configuration.

New Tables

The new tables added to Oracle Utilities Customer Care and Billing in this release are:

Table_Name	Description
C1_PA_RQST	Payment Arrangement Request
C1_PA_RQST_CHAR	Payment Arrangement Request - Characteristics
C1_PA_RQST_ELIG_RSLT	Payment Arrangement Request - Eligibility Results
C1_PA_RQST_K	Payment Arrangement Request - Key
C1_PA_RQST_LOG	Payment Arrangement Request - Log
C1_PA_RQST_LOG_PARM	Payment Arrangement Request - Log Parameters
C1_PA_RQST_REL_OBJ	Payment Arrangement Request - Related Object
C1_PA_RQST_TYPE	Payment Arrangement Request Type
C1_PA_RQST_TYPE_ALG	Payment Arrangement Request Type Algorithm Table
C1_PA_RQST_TYPE_CHAR	Payment Arrangement Request Type Characteristic Table
C1_PA_RQST_TYPE_DOWN_PAY_OPT	Payment Arrangement Request Type Down Payment Option
C1_PA_RQST_TYPE_DOWN_PAY_OPT_L	Payment Arrangement Request Type Down Payment Option Language
C1_PA_RQST_TYPE_L	Payment Arrangement Request Type Language

New Columns

The columns added to Oracle Utilities Customer Care and Billing in this release are as below:

Table_Name	Column_Name	Description
C1_PA_RQST	ACCT_ID	CHAR(10)
C1_PA_RQST	BO_STATUS_CD	CHAR(12)
C1_PA_RQST	BO_STATUS_REASON_CD	VARCHAR2(30)
C1_PA_RQST	BO_XML_DATA_AREA	XMLTYPE
C1_PA_RQST	BUS_OBJ_CD	CHAR(30)
C1_PA_RQST	CRE_DTTM	DATE
C1_PA_RQST	ILM_ARCH_SW	CHAR(1)
C1_PA_RQST	ILM_DT	DATE
C1_PA_RQST	PA_RQST_DOWN_PAY_AMT	NUMBER(15,2)
C1_PA_RQST	PA_RQST_DOWN_PAY_DUE_DT	DATE
C1_PA_RQST	PA_RQST_DT	DATE
C1_PA_RQST	PA_RQST_ID	CHAR(14)
C1_PA_RQST	PA_RQST_INSTALLMENT_AMT	NUMBER(15,2)
C1_PA_RQST	PA_RQST_NBR_INSTALLMENT	NUMBER(3,0)
C1_PA_RQST	PA_RQST_REASON	VARCHAR2(254)
C1_PA_RQST	PA_RQST_RSLT_FLG	CHAR(4)
C1_PA_RQST	PA_RQST_TOT_AMT	NUMBER(15,2)
C1_PA_RQST	PA_RQST_TYPE_CD	VARCHAR2(30)
C1_PA_RQST	STATUS_UPD_DTTM	DATE
C1_PA_RQST	VERSION	NUMBER(5,0)
C1_PA_RQST_CHAR	ADHOC_CHAR_VAL	VARCHAR2(254)
C1_PA_RQST_CHAR	CHAR_TYPE_CD	CHAR(8)
C1_PA_RQST_CHAR	CHAR_VAL	CHAR(16)
C1_PA_RQST_CHAR	CHAR_VAL_FK1	VARCHAR2(50)
C1_PA_RQST_CHAR	CHAR_VAL_FK2	VARCHAR2(50)
C1_PA_RQST_CHAR	CHAR_VAL_FK3	VARCHAR2(50)
C1_PA_RQST_CHAR	CHAR_VAL_FK4	VARCHAR2(50)
C1_PA_RQST_CHAR	CHAR_VAL_FK5	VARCHAR2(50)

Table_Name	Column_Name	Description
C1_PA_RQST_CHAR	PA_RQST_ID	CHAR(14)
C1_PA_RQST_CHAR	SEQ_NUM	NUMBER(3,0)
C1_PA_RQST_CHAR	SRCH_CHAR_VAL	VARCHAR2(50)
C1_PA_RQST_CHAR	VERSION	NUMBER(5,0)
C1_PA_RQST_ELIG_RSLT	BO_XML_DATA_AREA	XMLTYPE
C1_PA_RQST_ELIG_RSLT	PA_RQST_CRIT_RSLT_FLG	CHAR(4)
C1_PA_RQST_ELIG_RSLT	PA_RQST_ELIG_CRIT_SEQ	NUMBER(3,0)
C1_PA_RQST_ELIG_RSLT	PA_RQST_ID	CHAR(14)
C1_PA_RQST_ELIG_RSLT	VERSION	NUMBER(5,0)
C1_PA_RQST_K	ENV_ID	NUMBER(6,0)
C1_PA_RQST_K	PA_RQST_ID	CHAR(14)
C1_PA_RQST_LOG	ADHOC_CHAR_VAL	VARCHAR2(254)
C1_PA_RQST_LOG	BO_STATUS_CD	CHAR(12)
C1_PA_RQST_LOG	BO_STATUS_REASON_CD	VARCHAR2(30)
C1_PA_RQST_LOG	CHAR_TYPE_CD	CHAR(8)
C1_PA_RQST_LOG	CHAR_VAL	CHAR(16)
C1_PA_RQST_LOG	CHAR_VAL_FK1	VARCHAR2(50)
C1_PA_RQST_LOG	CHAR_VAL_FK2	VARCHAR2(50)
C1_PA_RQST_LOG	CHAR_VAL_FK3	VARCHAR2(50)
C1_PA_RQST_LOG	CHAR_VAL_FK4	VARCHAR2(50)
C1_PA_RQST_LOG	CHAR_VAL_FK5	VARCHAR2(50)
C1_PA_RQST_LOG	DESCRLONG	VARCHAR2(4000)
C1_PA_RQST_LOG	LOG_DTTM	DATE
C1_PA_RQST_LOG	LOG_ENTRY_TYPE_FLG	CHAR(4)
C1_PA_RQST_LOG	MESSAGE_CAT_NBR	NUMBER(5,0)
C1_PA_RQST_LOG	MESSAGE_NBR	NUMBER(5,0)
C1_PA_RQST_LOG	PA_RQST_ID	CHAR(14)
C1_PA_RQST_LOG	SEQNO	NUMBER(5,0)
C1_PA_RQST_LOG	USER_ID	CHAR(8)
C1_PA_RQST_LOG	VERSION	NUMBER(5,0)
C1_PA_RQST_LOG_PARM	MSG_PARM_TYP_FLG	CHAR(4)

Table_Name	Column_Name	Description
C1_PA_RQST_LOG_PARM	MSG_PARM_VAL	VARCHAR2(2000)
C1_PA_RQST_LOG_PARM	PA_RQST_ID	CHAR(14)
C1_PA_RQST_LOG_PARM	PARM_SEQ	NUMBER(3,0)
C1_PA_RQST_LOG_PARM	SEQNO	NUMBER(5,0)
C1_PA_RQST_LOG_PARM	VERSION	NUMBER(5,0)
C1_PA_RQST_REL_OBJ	BO_XML_DATA_AREA	XMLTYPE
C1_PA_RQST_REL_OBJ	MAINT_OBJ_CD	CHAR(12)
C1_PA_RQST_REL_OBJ	PA_RQST_ID	CHAR(14)
C1_PA_RQST_REL_OBJ	PA_RQST_REL_OBJ_TYPE_FLG	CHAR(4)
C1_PA_RQST_REL_OBJ	PK_VALUE1	VARCHAR2(254)
C1_PA_RQST_REL_OBJ	PK_VALUE2	VARCHAR2(254)
C1_PA_RQST_REL_OBJ	PK_VALUE3	VARCHAR2(254)
C1_PA_RQST_REL_OBJ	PK_VALUE4	VARCHAR2(254)
C1_PA_RQST_REL_OBJ	PK_VALUE5	VARCHAR2(254)
C1_PA_RQST_REL_OBJ	SEQ_NUM	NUMBER(3,0)
C1_PA_RQST_REL_OBJ	VERSION	NUMBER(5,0)
C1_PA_RQST_TYPE	BO_XML_DATA_AREA	XMLTYPE
C1_PA_RQST_TYPE	BUS_OBJ_CD	CHAR(30)
C1_PA_RQST_TYPE	CIS_DIVISION	CHAR(5)
C1_PA_RQST_TYPE	PA_RQST_CRIT_FAIL_INSTR_FLG	CHAR(4)
C1_PA_RQST_TYPE	PA_RQST_CRIT_RSLT_DISP_FLG	CHAR(4)
C1_PA_RQST_TYPE	PA_RQST_DOWN_PAY_APPL_FLG	CHAR(4)
C1_PA_RQST_TYPE	PA_RQST_DOWN_PAY_DAYS_TIL_DUE	NUMBER(5,0)
C1_PA_RQST_TYPE	PA_RQST_DOWN_PAY_DT_CALC_FLG	CHAR(4)
C1_PA_RQST_TYPE	PA_RQST_DOWN_PAY_GRACE_DAYS	NUMBER(3,0)
C1_PA_RQST_TYPE	PA_RQST_FLAT_DOWN_PAY_OPT_FLG	CHAR(4)
C1_PA_RQST_TYPE	PA_RQST_MIN_INSTALLMENT_AMT	NUMBER(15,2)

Table_Name	Column_Name	Description
C1_PA_RQST_TYPE	PA_RQST_MNL_SEL_CAND_SAS_FLG	CHAR(4)
C1_PA_RQST_TYPE	PA_RQST_TYPE_CD	VARCHAR2(30)
C1_PA_RQST_TYPE	PA_RQST_TYPE_STATUS_FLG	CHAR(4)
C1_PA_RQST_TYPE	TRANS_BUS_OBJ_CD	CHAR(30)
C1_PA_RQST_TYPE	VERSION	NUMBER(5,0)
C1_PA_RQST_TYPE_ALG	ALG_CD	CHAR(12)
C1_PA_RQST_TYPE_ALG	PA_RQST_ALG_ENTITY_FLG	CHAR(4)
C1_PA_RQST_TYPE_ALG	PA_RQST_TYPE_CD	VARCHAR2(30)
C1_PA_RQST_TYPE_ALG	SEQ_NUM	NUMBER(3,0)
C1_PA_RQST_TYPE_ALG	VERSION	NUMBER(5,0)
C1_PA_RQST_TYPE_CHAR	ADHOC_CHAR_VAL	VARCHAR2(254)
C1_PA_RQST_TYPE_CHAR	CHAR_TYPE_CD	CHAR(8)
C1_PA_RQST_TYPE_CHAR	CHAR_VAL	CHAR(16)
C1_PA_RQST_TYPE_CHAR	CHAR_VAL_FK1	VARCHAR2(50)
C1_PA_RQST_TYPE_CHAR	CHAR_VAL_FK2	VARCHAR2(50)
C1_PA_RQST_TYPE_CHAR	CHAR_VAL_FK3	VARCHAR2(50)
C1_PA_RQST_TYPE_CHAR	CHAR_VAL_FK4	VARCHAR2(50)
C1_PA_RQST_TYPE_CHAR	CHAR_VAL_FK5	VARCHAR2(50)
C1_PA_RQST_TYPE_CHAR	PA_RQST_TYPE_CD	VARCHAR2(30)
C1_PA_RQST_TYPE_CHAR	SEQ_NUM	NUMBER(3,0)
C1_PA_RQST_TYPE_CHAR	SRCH_CHAR_VAL	VARCHAR2(50)
C1_PA_RQST_TYPE_CHAR	VERSION	NUMBER(5,0)
C1_PA_RQST_TYPE_DOWN_PAY_OPT	PA_RQST_DOWN_PAY_PCT	NUMBER(2,0)
C1_PA_RQST_TYPE_DOWN_PAY_OPT	PA_RQST_TYPE_CD	VARCHAR2(30)
C1_PA_RQST_TYPE_DOWN_PAY_OPT	SEQ_NUM	NUMBER(3,0)
C1_PA_RQST_TYPE_DOWN_PAY_OPT	VERSION	NUMBER(5,0)
C1_PA_RQST_TYPE_DOWN_PAY_OPT_L	DESCR100	VARCHAR2(100)
C1_PA_RQST_TYPE_DOWN_PAY_OPT_L	LANGUAGE_CD	CHAR(3)

Table_Name	Column_Name	Description
C1_PA_RQST_TYPE_DOWN_PAY_OPT_L	PA_RQST_TYPE_CD	VARCHAR2(30)
C1_PA_RQST_TYPE_DOWN_PAY_OPT_L	SEQ_NUM	NUMBER(3,0)
C1_PA_RQST_TYPE_DOWN_PAY_OPT_L	VERSION	NUMBER(5,0)
C1_PA_RQST_TYPE_L	DESCR100	VARCHAR2(100)
C1_PA_RQST_TYPE_L	DESCRLONG	VARCHAR2(4000)
C1_PA_RQST_TYPE_L	LANGUAGE_CD	CHAR(3)
C1_PA_RQST_TYPE_L	PA_RQST_TYPE_CD	VARCHAR2(30)
C1_PA_RQST_TYPE_L	VERSION	NUMBER(5,0)
CI_ID_TYPE	VAL_ID_TYPE_FOR_DUPS_FLG	CHAR(4)
CI_SA_TYPE	PA_ELIG_FLG	CHAR(4)

New Indexes

The indexes added in this release of Oracle Utilities Customer Care and Billing are as below:

Table_Name	Index_Name
C1_PA_RQST_TYPE	C1C080P0
C1_PA_RQST_TYPE	C1C080S1
C1_PA_RQST_TYPE	C1C080S2
C1_PA_RQST_TYPE_L	C1C081P0
C1_PA_RQST_TYPE_DOWN_PAY_OPT	C1C082P0
C1_PA_RQST_TYPE_DOWN_PAY_OPT_L	C1C083P0
C1_PA_RQST_TYPE_ALG	C1C084P0
C1_PA_RQST_TYPE_CHAR	C1C085P0
C1_PA_RQST_TYPE_CHAR	C1C085S1
C1_PA_RQST	C1T020P0
C1_PA_RQST	C1T020S1
C1_PA_RQST_ELIG_RSLT	C1T022P0
C1_PA_RQST_REL_OBJ	C1T023P0
C1_PA_RQST_CHAR	C1T024P0

Table_Name	Index_Name
C1_PA_RQST_CHAR	C1T024S1
C1_PA_RQST_LOG	C1T025P0
C1_PA_RQST_LOG	C1T025S1
C1_PA_RQST_LOG	C1T025S2
C1_PA_RQST_LOG_PARM	C1T026P0
C1_PA_RQST_K	C1T027P0

New Functions

There are no new functions added to Oracle Utilities Customer Care and Billing in this release.

New Views

The view added to Oracle Utilities Customer Care and Billing in this release is:

View_Name
CI_TD_ENTRY_OPEN_VW

New Batch Control Application Services

The batch controls listed below use the designated application services:

Batch Control	Description	New Application Service
ACTVTAPY	Activate auto-pay	C1-APAY
ADM	Account debt monitor	ADM
ADM2	Account debt monitor, minimum days review	ADM2
ANLYZSAR	Analyze SA relationship	ANLYZSAR
APAYACH	Auto pay extract - ACH	C1-APAY
APAYCRET	Create autopay on extract date	C1-APAY
APAYDSFR	Distribute and freeze autopay	C1-APAY
APDL	Accounts payable download	APDL
ASSGNSBN	Assign sequential bill numbers	ASSGNSBN
BALAPY	Create autopay tender controls	BALAPY
BCASSIGN	Assign balance control id to FTs	C1-BCG
BCGNEW	Create new balance control group	C1-BCG

Batch Control	Description	New Application Service
BCGSNAP	Create/validate balance control snapshot	C1-BCG
BCU1	Billable Charge Upload 1 - Validate Staging	C1-BCU
BCU2	Billable Charge Upload 2 - Populate	C1-BCU
BILLING	Create bills using bill cycle	BILLING
BUDMON	Monitor budgets	BUDMON
BUDTRUP	True up budgets	BUDTRUP
C1-ACAMT	Update Bill Segment Audit Calc Amount	C1-ACAMT
C1-ACTRQ	Activity Request Monitor (Deferred)	C1-ACTREQ
C1-ADAMT	Update Adjustment Audit Calc Amount	C1-ADAMT
C1-ADMOV	Overdue Monitor	C1-ADMOV
C1-ADUP1	Adjustment Upload Preprocessor	C1-ADUP
C1-ADUP2	Adjustment Upload	C1-ADUP
C1-ADURS	Resolve Suspense Adjustments	C1-ADURS
C1-APACH	Auto pay extract - ACH (with offset days parameter)	C1-APAY
C1-APRPR	Approval Request Monitor	C1-APRREQ
C1-APRTR	Approval Request Monitor (Deferred)	C1-APRREQ
C1-ARQPR	Activity Request Monitor	C1-ACTREQ
C1-BILL	Billing	BILLING
C1-BLAPC	Balance Autopay Control Tables	BALAPY
C1-BLCMP	Freeze and Complete Pending Bills	C1-BLCMP
C1-BLEIL	Billing Data Initial Load for DataConnect	C1-DATACON
C1-BLRVI	Bill Review Insertion	C1-BLRV
C1-BLRVV	Bill Review Validation	C1-BLRV
C1-BLRVW	Review Bills for Settlement of Held GL Amounts	C1-BLRVW
C1-BNBAS	Assign Bill Document Numbers	C1-DOCNBR
C1-BSYEX	Billing Data Extract for DataConnect	C1-DATACON
C1-CPRPR	Conservation Program Monitor Process	C1-CONSRVMON
C1-CPRQ	Conservation Program Monitor Process (Deferred)	C1-CONSRVMON
C1-CRRMP	Customer Relationship Request Periodic Monitor Process	C1-CRRMP

Batch Control	Description	New Application Service
C1-CSTRS	Case Scheduled Transition	C1-CASETRAN
C1-CTCOE	Complete To Do Entry of Contract Option Event Exception	C1-CTCOE
C1-CTIDS	Complete To Do Entry of Interval Data Set Exception	C1-CTIDS
C1-CTRDS	Complete To Do Entry of Interval Register Data Set Exception	C1-CTRDS
C1-CTTDS	Complete To Do Entry of TOU Data Set Exception	C1-CTTDS
C1-DVTSC	Cancel Device Test Selection	C1-DVTSC
C1-ECRVL	Encrypt Legacy Order Field Col Ref Value	C1-ENCR
C1-INITR	Initiative Periodic Monitor Process	C1-INITIATIVE
C1-INPUS	Create Person Contact from Person Phone/ Email	C1-INPUS
C1-ISSCT	Issuing Center Monitor	C1-ISSCT
C1-LDEVT	Lead Event Periodic Monitor Process	C1-INITIATIVE
C1-LDEXT	Advanced Analysis System Lead - Extract	C1-LEAD
C1-LDRTY	Advanced Analysis System Lead - Retry	C1-LEAD
C1-LDTR	Lead Periodic Monitor Process	C1-INITIATIVE
C1-LEADD	Lead Disposition	C1-INITIATIVBOAS
C1-LEADG	Lead Generation	C1-INITIATIVBOAS
C1-MMTPR	Market Message Type Monitor	C1-MMTPR
C1-NEMDF	NEM Scheduled Monitor Process (Deferred)	C1-NEMDF
C1-OCBG	Off Cycle Bill Generator Monitor	C1-OCBG
C1-ODET	Overdue and Cut Event Manager	C1-ODET
C1-PAYTP	Payment Template Monitor	C1-PAYTP
C1-PEPL1	Payment Event Upload Stage 1	C1-PEPL
C1-PEPL2	Payment Event Upload Stage 2	C1-PEPL
C1-PEPL3	Balance Tender Controls	C1-PEPL
C1-PNBAS	Assign Payment Event Document Numbers	C1-DOCNBR
C1-PPBER	Prepay Biller Task - Error	C1-PPB
C1-PPBTR	Prepay Biller Task	C1-PPB

Batch Control	Description	New Application Service
C1-PUBAL	Payment upload balance control tables	C1-PUPL
C1-RCTRQ	Rebate Claim Monitor	C1-RCTRQ
C1-RDFPR	Conservation Program Rebate Definition Monitor	C1-CONSRVMON
C1-SAEIL	SA-Based Initial Load for DataConnect	C1-DATACON
C1SAFTCT	SAFT-PT Audit Extract Concatenator	C1-SAFTPT
C1SAFTPT	SAFT-PT Audit Extract	C1-SAFTPT
C1-SASYX	SA-Based Extract for DataConnect	C1-DATACON
C1-SDDCE	SEPA Direct Debit Payment Extract	C1-SDDCE
C1-SMEIL	Meter History Initial Load for DataConnect	C1-DATACON
C1-SMISL	Synchronize Meter/Item Stock Location	C1-SMISL
C1-SMSYX	Meter History Extract for DataConnect	C1-DATACON
C1-SPEIL	SP-Based Initial Load for DataConnect	C1-DATACON
C1-SPSYX	SP-Based Extract for DataConnect	C1-DATACON
C1-SRDS	Service Route Download Staging	C1-SRDS
C1-SYNIL	Generic Sync Request Initial Load	C1-SYNIL
C1-TCRNB	Transition OCBG for Replacement Read Process	C1-TCRNB
C1-TDCOE	Create To Do Entry for Contract Option Event Exception	C1-TDCOE
C1-TDIDS	Create To Do Entry for Interval Data Set Exception	C1-TDIDS
C1-TDRDS	Create To Do Entry for Interval Register Data Set Exception	C1-TDRDS
C1-TDTDS	Create To Do Entry for TOU Data Set Exception	C1-TDTDS
C1-TOUTR	TOU Map Data Generation Monitor	C1-TOUTR
C1-UPDBF	Update Calc Rules Mapping	C1-UPDBF
C1-UPDNT	Update Notification Tasks	C1-UPDNT
C1-UPDPT	Update Payment Tender Alternate Currency	C1-UPDPT
C1-USGDF	Usage Scheduled Monitor Process (Deferred)	C1-USAGE
C1-USGTR	Usage Periodic Monitor Process	C1-USAGE
C1-WAMAS	Extract asset data for work and asset management	C1-WAMEXTRACT

Batch Control	Description	New Application Service
C1-WAMEX	Extract customer data for work and asset management	C1-WAMEXTRACT
C1-WFSUB	Workflow event trigger - Batch scheduler	C1-WORKFLOW
CAREPROG	Create customer contact for expiring SA char	CAREPROG
CASETRAN	Case Status Automatic Transition Process	C1-CASETRAN
CET	Collection event trigger	CET
CLOSEQTE	Close expired quotes	CLOSEQTE
CPCRALOC	Capital credit allocation	C1-CAPCRE
CPCRRETR	Capital credit retirement	C1-CAPCRE
CPM	Collection process monitor	CPM
DEPINTRF	Deposit interest refund	C1-DEPOSIT
DEPRFND	Deposit refund	C1-DEPOSIT
DEPRVW	Deposit review	C1-DEPOSIT
DSGPFODL	Dispatch Group Field Order Print	C1-FODWNLD
DWLDBILC	Download billable charge	DWLDBILC
DWLDCOLL	Download collection agency ref	DWLDCOLL
DWLDCONS	Download consumption	DWLDCONS
F1-ENCRS	Encrypt Legacy Schema Field Data	C1-ENCR
F1-ENCRT	Encrypt Legacy Table Field Data	C1-ENCR
FACOMPL	Field activity completion	C1-FAUPLD
FACT	Field activity remark activation	C1-FAUPLD
FANRMRCO	Complete FA using a recent MR	FANRMRCO
FAXROUT	Fax Routing	FAXROUT
FDS	Field order download staging	C1-FODWNLD
FOD	Automatic dispatch of FA's	C1-FODWNLD
FODL	Field order download extract	C1-FODWNLD
GLASSIGN	Populates GL_ACCT on CI_FT_GL	C1-GL
GLDL	GL download extract	C1-GL
GLS	Create GL download staging	C1-GL
IB-SPDB	Interval data set derivation	C1-INTERVAL
IB-STDB	SA specific TOU data creation	C1-INTERVAL
IPDSDVB	Interval prof data validation	C1-INTERVAL

Batch Control	Description	New Application Service
IPDSIDB	Determine profile for datasets	C1-INTERVAL
IREGDVB	Interval reg. data validation	C1-INTERVAL
IREGIDB	Determine register for datasets	C1-INTERVAL
LATEPYMT	Create late payment charges	LATEPYMT
LTRPRT	Letter Extract	LTRPRT
MASSCNCL	Mass bill cancellation	MASSCNCL
MASSROBL	Mass re-open of bills	MASSROBL
MDL	Meter read download extract	MDL
MDS	Create MR download staging	C1-MRDWNLD
MRRA	Execute MR remark algorithms	MRRA
MSR	Create MR schedule routes	C1-MRDWNLD
MUP1	Meter Read upload 1 - populate meter confg	C1-MRUPLD
MUP2	Meter Read upload 2 - populate meter read	C1-MRUPLD
NBBAPAY	Create autopay for NBB's	C1-NBB
NBBPS	Process NBB scheduled payments	C1-NBB
NDSXTR	Notif download staging extract	NDSXTR
PAYSPR	Pay service provider	PAYSPR
POSTROUT	Postal bill routing	C1-BILLPRINT
PPAPAY	Generate autopay for pay plans	C1-PAYPLAN
PPM	Pay plan monitor	C1-PAYPLAN
PSASPM	Pending SA/SP monitor	PSASPM
PUPL	Payment upload	C1-PUPL
PY-RPE	Resolve payments in error	PY-RPE
QUOTROUT	Quote routing	QUOTROUT
REACH	YTD charitable contributions	REACH
REDSAAMT	Update old FT's as redundant	REDSAAMT
REGCNST	Register constant validation	REGCNST
RTTYPOST	Postal bill routing using bill print software	C1-BILLPRINT
SAACT	Activate pending start/stop SA	SAACT
SAEXPIRE	Stop expired SAs	SAEXPIRE
SARENEW	Renew Service Agreement	SARENEW
SASP	Find read for SA/SP	SASP

Batch Control	Description	New Application Service
SEC	Severance event completion	C1-SEVEVT
SED	Severance event set dependency date	C1-SEVEVT
SET	Severance event trigger	C1-SEVEVT
STMDWLD	Download statements	STMDWLD
STMPRD	Create statements	STMPRD
TD-BCUPL	To Do for Billable Charge in Error	C1-BILLERR
TD-BIERR	To Do for Bills in Error	C1-BILLERR
TD-BSERR	To Do for Bill Segment in Error	C1-BILLERR
TD-BTERR	To Do for Batch Errors	TD-BTERR
TD-CCCB	To Do for Customer Contact	TD-CCCB
TD-CEVT	To Do for C&C Events	TD-CEVT
TD-DTCS1	Create To Do for Deposit/Tender Upload Error	TD-DTCS1
TD-DTCS2	Complete To Do for Deposit/Tender Upload Error	TD-DTCS2
TD-DTCST	To Do for Deposit/Tender Upload Error	TD-DTCST
TD-ECBK	To Do for Callback Orders	TD-ECBK
TD-EPND	To Do for Pending Orders	TD-EPND
TD-FACT	To Do for Field Activity Remark Exception	TD-FACT
TD-FAUPL	To Do for Field Activity Upload in Error	TD-FAUPL
TD-HILO	To Do for Meter Read High/Low Errors	TD-HILO
TD-MODTL	To Do for Open-Dispute Match Event	TD-MODTL
TD-MONTL	To Do for Open/non-Dispute Match Event	TD-MONTL
TD-MRRER	To Do for Meter Read Remarks in Error	TD-MRRER
TD-MRUPL	To Do for Meter Read Upload in Error	TD-MRUPL
TD-NCDEX	To Do for Non Cash Deposit	TD-NCDEX
TD-NOBC	To Do for Account without a Bill Cycle	TD-NOBC
TD-NOMR	To Do for SP without Meter Read Cycle	TD-NOMR
TD-NTDWN	To Do for NT Download in Error	TD-NTDWN
TD-NTUPL	To Do for Notification Upload in Error	TD-NTUPL
TD-PYERR	To Do for Payments in Error/Unfrozen	TD-PYERR
TD-PYUPL	To Do for Payment Staging Error	TD-PYUPL
TD-SEVT	To Do for Severance Events	TD-SEVT

Batch Control	Description	New Application Service
TD-SPRO	To Do for Severance Processes	TD-SPRO
TD-SSFTL	To Do for Old Pending Start/Stops	TD-SSFTL
TD-UNBAL	To Do for Unbalanced Pay Event	TD-UNBAL
TD-WEXTL	To Do for Workflow Events in Error	TD-WEXTL
TD-WOEVT	To Do for Write-Off Events	TD-WOEVT
TD-XAIDN	To Do for XAI Download Staging in error	TD-XAIDN
TD-XAIUP	To Do for XAI Upload Staging in error	TD-XAIUP
TREND	Trend update	TREND
UARENEW	Umbrella Agreement Renewal	UARENEW
UPDERR	Update Batch Run/Thread Status	UPDERR
WAITCOM	Completes a WF Event in Waiting state	C1-WORKFLOW
WAITFA	Wait for field actv completion	C1-WORKFLOW
WAITMAN	Workflow timeout manual wait	C1-WORKFLOW
WAITNT	Wait for notification response	C1-WORKFLOW
WET	Write off event trigger	WET
WFET	Workflow event trigger	C1-WORKFLOW
WFPRINIT	Workflow process initiation	C1-WORKFLOW
WPM	Write off monitor process	WPM
WX-NOTIF	Self-Service Notification Monitor	WX-NOTIF

Conversion Batch Control Application Services

The following batch controls use the C1-CONVERSION application service:

Batch Control	Description
CIPVAAPI	Insert CI_ACCT_APAY
CIPVAAPK	Generate CI_ACCT_APAY keys
CIPVAAPV	Foreign Key validation for CI_ACCT_APAY
CIPVACCI	Insert CI_ACCT
CIPVACCK	Generate CI_ACCT keys
CIPVACHI	Insert CI_ACCT_CHAR
CIPVACHV	Foreign Key validation for CI_ACCT_CHAR
CIPVACPI	Insert CI_ACCT_PER

Batch Control	Description
CIPVACPV	Foreign Key validation for CI_ACCT_PER
CIPVADCI	Insert CI_ADJ_CHAR
CIPVADCV	Foreign Key validation for CI_ADJ_CHAR
CIPVADJI	Insert CI_ADJ
CIPVADJK	Generate CI_ADJ keys
CIPVADJV	Foreign Key validation for CI_ADJ
CIPVAPAI	Insert CI_PRM_ALT_ADDR
CIPVAPAK	Generate CI_PRM_ALT_ADDR keys
CIPVAPAV	Foreign Key validation for CI_PRM_ALT_ADDR
CIPVAPRI	Insert CI_ADJ_APREQ
CIPVAPRK	Generate CI_ADJ_APREQ keys
CIPVAPRV	Foreign Key validation for CI_ADJ_APREQ
CIPVARHI	Insert CI_COLL_AGY_HIS
CIPVARHV	Foreign Key validation for CI_COLL_AGY_HIS
CIPVARSI	Insert CI_ADM_RVW_SCH
CIPVARSV	Foreign Key validation for CI_ADM_RVW_SCH
CIPVBCCI	Insert CI_BSEG_CL_CHAR
CIPVBCCV	Foreign Key validation for CI_BSEG_CL_CHAR
CIPVBCGI	Insert CI_BILL_CHG
CIPVBCGK	Generate CI_BILL_CHG keys
CIPVBCGV	Foreign Key validation for CI_BILL_CHG
CIPVBCHI	Insert CI_BILL_CHAR
CIPVBCHV	Foreign Key validation for CI_BILL_CHAR
CIPVBCLI	Insert CI_B_CHG_LINE
CIPVBCLV	Foreign Key validation for CI_B_CHG_LINE
CIPVBFVI	Insert CI_BF_VAL
CIPVBFVV	Foreign Key validation for CI_BF_VAL
CIPVBILK	Generate CI_BILL keys
CIPVBLLI	Insert CI_BILL
CIPVBLLV	Foreign Key validation for CI_BILL
CIPVBLMI	Insert CI_BILL_MSGS
CIPVBLMV	Foreign Key validation for CI_BILL_MSGS
CIPVBLRI	Insert CI_BILL_ROUTING

Batch Control	Description
CIPVBLRV	Foreign Key validation for CI_BILL_ROUTING
CIPVBRVI	Insert CI_BILL_RVW_SCH
CIPVBRVV	Foreign Key validation for CI_BILL_RVW_SCH
CIPVBSAI	Insert CI_BILL_SA
CIPVBSAV	Foreign Key validation for CI_BILL_SA
CIPVBSCI	Insert CI_BSEG_CALC
CIPVBSCV	Foreign Key validation for CI_BILL_CALC
CIPVBSGK	Generate CI_BSEG keys
CIPVBSII	Insert CI_BSEG_ITEM
CIPVBSIV	Foreign Key validation for CI_BSEG_ITEM
CIPVBSLI	Insert CI_BSEG_CALC_LN
CIPVBSLV	Foreign Key validation for CI_BSEG_CALC_LN
CIPVCARI	Insert CI_COLL_AGY_REF
CIPVCARK	Generate CI_COLL_AGY_REF keys
CIPVCARV	Foreign Key validation for CI_COLL_AGY_REF
CIPVCCAI	Insert CI_COP_CHAR
CIPVCCFV	Foreign Key validation for CI_COP_CHAR
CIPVCCHI	Insert CI_CC_CHAR
CIPVCCHV	Foreign Key validation for CI_CC_CHAR
CIPVCCLI	Insert CI_CC_LOG
CIPVCCLV	Foreign Key validation for CI_CC_LOG
CIPVCCTK	Generate CI_CC keys
CIPVCECI	Insert CI_COLL_EVT_CC
CIPVCECV	Foreign Key validation for CI_COLL_EVT_CC
CIPVCEVI	Insert CI_COP_EVT
CIPVCEVK	Generate CI_COP_EVT keys
CIPVCLPI	Insert CI_COLL_PROC
CIPVCLPK	Generate CI_COLL_PROC keys
CIPVCLPV	Foreign Key validation for CI_COLL_PROC
CIPVCLSI	Insert CI_COLL_PROC_SA
CIPVCLSV	Foreign Key validation for CI_COLL_PROC_SA
CIPVCOLI	Insert CI_COP_L
CIPVCOLV	Foreign Key validation for CI_COP_L

Batch Control	Description
CIPVCOPI	Insert CI_COP
CIPVCOPK	Generate CI_COP keys
CIPVCRRK	Generate CI_CR_RAT_HIST keys
CIPVCRTI	Insert CI_CR_RAT_HIST
CIPVCRTV	Foreign Key validation for CI_CR_RAT_HIST
CIPVCSCI	Insert CI_CC
CIPVCSCV	Foreign Key validation for CI_CC
CIPVCVCI	Insert CI_COP_EVT_CHAR
CIPVCVCV	Foreign Key validation for CI_COP_EVT_CHAR
CIPVCVNI	Insert CI_COLL_EVT
CIPVCVNV	Foreign Key validation for CI_COLL_EVT
CIPVDCRI	Insert CI_DCL
CIPVDCRK	Generate CI_DCL keys
CIPVDTCI	Insert CI_DV_TEST_CHAR
CIPVDTCV	Foreign Key validation for CI_DV_TEST_CHAR
CIPVDTMI	Insert CI_DV_TEST_COMP
CIPVDTMV	Foreign Key validation for CI_DV_TEST_COMP
CIPVDTRI	Insert CI_DV_TEST_RES
CIPVDTRV	Foreign Key validation for CI_DV_TEST_RES
CIPVDTTI	Insert CI_DV_TEST
CIPVDTTK	Generate CI_DV_TEST keys
CIPVDTTV	Foreign Key validation for CI_DV_TEST
CIPVFACI	Insert CI_FA
CIPVFACK	Generate CI_FA keys
CIPVFAFV	Foreign Key validation for CI_FA
CIPVFAHI	Insert CI_FA_CHAR
CIPVFAHV	Foreign Key validation for CI_FA_CHAR
CIPVFALI	Insert CI_FA_LOG
CIPVFALV	Foreign Key validation for CI_FA_LOG
CIPVFARI	Insert CI_FA_REM
CIPVFARV	Foreign Key validation for CI_FA_REM
CIPVFORI	Insert CI_FO
CIPVFORK	Generate CI_FO keys

Batch Control	Description
CIPVFORV	Foreign Key validation for CI_FO
CIPVFSTI	Insert CI_FA_STEP
CIPVFSTV	Foreign Key validation for CI_FA_STEP
CIPVFTFI	Insert CI_FT
CIPVFTFV	Foreign Key validation for CI_FT
CIPVFTGI	Insert CI_FT_GL
CIPVFTGV	Foreign Key validation for CI_FT_GL
CIPVFTPI	Insert CI_FT_PROC
CIPVFTPV	Foreign Key validation for CI_FT_PROC
CIPVFTXK	Generate CI_FT keys
CIPVIDSI	Insert CI_INTV_DATA_SET
CIPVIDSK	Generate CI_INTV_DATA_SET keys
CIPVIEQI	Insert CI_ITEM_EQ
CIPVIEQV	Foreign Key validation for CI_ITEM_EQ
CIPVILHI	Insert CI_ITEM_LOC_HIS
CIPVILHK	Generate CI_ITEM_LOC_HIS keys
CIPVILHV	Foreign Key validation for CI_ITEM_LOC_HIS
CIPVINLI	Insert CI_INTV_PF_L
CIPVINLV	Foreign Key validation for CI_INTV_PF_L
CIPVINPI	Insert CI_INTV_PF
CIPVINPK	Generate CI_INTV_PF keys
CIPVIRSI	Insert CI_REG_DATA_SET
CIPVIRSK	Generate CI_REG_DATA_SET keys
CIPVITCI	Insert CI_ITEM_CHAR
CIPVITCV	Foreign Key validation for CI_ITEM_CHAR
CIPVITDI	Insert CI_INTV_DATA
CIPVITFV	Foreign Key validation for CI_INTV_DATA
CIPVITMI	Insert CI_ITEM
CIPVITMK	Generate CI_ITEM keys
CIPVITVI	Insert CI_INTV_VAL
CIPVITVV	Foreign Key validation for CI_INTV_VAL
CIPVIVSI	Insert CI_INTV_VAL_SET
CIPVIVSK	Generate CI_INTV_VAL_SET keys

Batch Control	Description
CIPVLLDI	Insert CI_LL_DETAIL
CIPVLLDV	Foreign Key validation for CI_LL_DETAIL
CIPVLNDI	Insert CI_LANDLORD
CIPVLNDK	Generate CI_LANDLORD keys
CIPVMEQI	Insert CI_MTR_EQ
CIPVMEQV	Foreign Key validation for CI_MTR_EQ
CIPVMIDI	Insert CI_MTR_ID
CIPVMIDV	Foreign Key validation for CI_MTR_ID
CIPVMLHI	Insert CI_MTR_LOC_HIS
CIPVMLHK	Generate CI_MTR_LOC_HIS keys
CIPVMLHV	Foreign Key validation for CI_MTR_LOC_HIS
CIPVMRCI	Insert CI_MR_CHAR
CIPVMRCV	Foreign Key validation for CI_MR_CHAR
CIPVMRDI	Insert CI_MR
CIPVMRDK	Generate CI_MR keys
CIPVMRDV	Foreign Key validation for CI_MR
CIPVMRMI	Insert CI_MR_REM
CIPVMRMV	Foreign Key validation for CI_MR_REM
CIPVMSGI	Insert CI_ACCT_MSG
CIPVMSGV	Foreign Key validation for CI_ACCT_MSG
CIPVMTCI	Insert CI_MTR_CHAR
CIPVMTCV	Foreign Key validation for CI_MTR_CHAR
CIPVMTGI	Insert CI_MTR_CONFIG
CIPVMTGK	Generate CI_MTR_CONFIG keys
CIPVMTRI	Insert CI_MTR
CIPVMTRK	Generate CI_MTR keys
CIPVNBSI	Insert CI_NB_SA
CIPVNBSV	Foreign Key validation for CI_NB_SA
CIPVNCDI	Insert CI_NCD
CIPVNCDV	Foreign Key validation for CI_NCD
CIPVNPMI	Insert CI_SA_NB_PARM
CIPVNPMV	Foreign Key validation for CI_SA_NB_PARM
CIPVNSPI	Insert CI_NB_SCHED_PAY

Batch Control	Description
CIPVNSPK	Generate CI_NB_SCHED_PAY keys
CIPVNSPV	Foreign Key validation for CI_NB_SCHED_PAY
CIPVPAOI	Insert CI_PER_ADDR_OVRD
CIPVPAOV	Foreign Key validation for CI_PER_ADDR_OVRD
CIPVPAYI	Insert CI_PAY
CIPVPAYK	Generate CI_PAY keys
CIPVPAYV	Foreign Key validation for CI_PAY
CIPVPCHI	Insert CI_PREM_CHAR
CIPVPCHV	Foreign Key validation for CI_PREM_CHAR
CIPVPCNI	Insert C1_PER_CONTDET
CIPVPCNK	Generate C1_PER_CONTDET keys
CIPVPCNV	Foreign Key validation for C1_PER_CONTDET
CIPVPECI	Insert CI_PAY_EVT_CHAR
CIPVPECV	Foreign Key validation for CI_PAY_EVT_CHAR
CIPVPERI	Insert CI_PER
CIPVPERK	Generate CI_PERSON keys
CIPVPGOI	Insert CI_PREM_GEO
CIPVPGOV	Foreign Key validation for CI_PREM_GEO
CIPVPIDI	Insert CI_PER_ID
CIPVPIDV	Foreign Key validation for CI_PER_ID
CIPVPNMI	Insert CI_PER_NAME
CIPVPNMV	Foreign Key validation for CI_PER_NAME
CIPVPPEI	Insert CI_PER_PER
CIPVPPEV	Foreign Key validation for CI_PER_PER
CIPVPPHI	Insert CI_PER_PHONE
CIPVPPHV	Foreign Key validation for CI_PER_PHONE
CIPVPRCI	Insert CI_PER_CHAR
CIPVPRCV	Foreign Key validation for CI_PER_CHAR
CIPVPRMI	Insert CI_PREM
CIPVPRMK	Generate CI_PREM keys
CIPVPSAI	Insert CI_PER_ADDR_SEAS
CIPVPSAV	Foreign Key validation for CI_PER_ADDR_SEAS
CIPVPSGI	Insert CI_PAY_SEG

Batch Control	Description
CIPVPSGK	Generate CI_PAY_SEG keys
CIPVPSGV	Foreign Key validation for CI_PAY_SEG
CIPVPYCI	Insert CI_PAY_CHAR
CIPVPYCV	Foreign Key validation for CI_PAY_CHAR
CIPVPYEI	Insert CI_PAY_EVENT
CIPVPYEK	Generate CI_PAY_EVENT keys
CIPVREDI	Insert CI_REG_DATA
CIPVREFV	Foreign Key validation for CI_REG_DATA
CIPVREGI	Insert CI_REG
CIPVREGK	Generate CI_REG keys
CIPVREGV	Foreign Key validation for CI_REG
CIPVRGCI	Insert CI_REG_CHAR
CIPVRGCV	Foreign Key validation for CI_REG_CHAR
CIPVRGRI	Insert CI_REG_READ
CIPVRGRV	Foreign Key validation for CI_REG_READ
CIPVRRDK	Generate CI_REG_READ keys
CIPVSACI	Insert CI_SA_CHAR
CIPVSACV	Foreign Key validation for CI_SA_CHAR
CIPVSAHI	Insert CI_SA_RS_HIST
CIPVSAHV	Foreign Key validation for CI_SA_RS_HIST
CIPVSAOI	Insert CI_SA_CONTERM
CIPVSAOV	Foreign Key validation for CI_SA_CONTERM
CIPVSAPI	Insert CI_SA_SP
CIPVSAPV	Foreign Key validation for CI_SA_SP
CIPVSAQI	Insert CI_SA_CONT_QTY
CIPVSAQV	Foreign Key validation for CI_SA_CONT_QTY
CIPVSARI	Insert CI_SA_RCHG_HIST
CIPVSARV	Foreign Key validation for CI_SA_RCHG_HIST
CIPVSCAI	Insert CI_SCM_ACCT
CIPVSCAV	Foreign Key validation for CI_SCM_ACCT
CIPVSCBV	Foreign Key validation for CI_SCM
CIPVSCCI	Insert CI_SCM_CHAR
CIPVSCCV	Foreign Key validation for CI_SCM_CHAR

Batch Control	Description
CIPVSCFI	Insert CI_SC_EVT_FT
CIPVSCFK	Generate CI_SC_EVT_FT keys
CIPVSCFV	Foreign Key validation for CI_SC_EVT_FT
CIPVSCMI	Insert CI_SCM
CIPVSCMK	Generate CI_SCM keys
CIPVSCOI	Insert CI_SA_COP_OVRD
CIPVSCOV	Foreign Key validation for CI_SA_COP_OVRD
CIPVSCPI	Insert CI_SA_COP
CIPVSCPK	Generate CI_SA_COP keys
CIPVSCPV	Foreign Key validation for CI_SA_COP
CIPVSCVI	Insert CI_SC_EVT
CIPVSCVK	Generate CI_SC_EVT keys
CIPVSCVV	Foreign Key validation for CI_SC_EVT
CIPVSECI	Insert CI_SEV_EVT_CC
CIPVSECV	Foreign Key validation for CI_SEV_EVT_CC
CIPVSEDI	Insert CI_SEV_EVT_DEP
CIPVSEDV	Foreign Key validation for CI_SEV_EVT_DEP
CIPVSEFI	Insert CI_SEV_EVT_FA
CIPVSEFV	Foreign Key validation for CI_SEV_EVT_FA
CIPVSEGI	Insert CI_BSEG
CIPVSEGV	Foreign Key validation for CI_BSEG
CIPVSEPI	Insert CI_SEV_PROC
CIPVSEPK	Generate CI_SEV_PROC keys
CIPVSEPV	Foreign Key validation for CI_SEV_PROC
CIPVSEQI	Insert CI_SP_EQ
CIPVSEQV	Foreign Key validation for CI_SP_EQ
CIPVSEVI	Insert CI_SEV_EVT
CIPVSEVV	Foreign Key validation for CI_SEV_EVT
CIPVSIEI	Insert CI_SP_ITEM_EVT
CIPVSIEV	Foreign Key validation for CI_SP_ITEM_EVT
CIPVSIFV	Foreign Key validation for CI_SA_INTV_PF
CIPVSIHI	Insert CI_SP_ITEM_HIST
CIPVSIHK	Generate CI_SP_ITEM_HIST keys

Batch Control	Description
CIPVSIHV	Foreign Key validation for CI_SP_ITEM_HIST
CIPVSIPI	Insert CI_SA_INTV_PF
CIPVSMEI	Insert CI_SP_MTR_EVT
CIPVSMEV	Foreign Key validation for CI_SP_MTR_EVT
CIPVSMGI	Insert CI_SA_MSG
CIPVSMGV	Foreign Key validation for CI_SA_MSG
CIPVSMHI	Insert CI_SP_MTR_HIST
CIPVSMHK	Generate CI_SP_MTR_HIST keys
CIPVSMHV	Foreign Key validation for CI_SP_MTR_HIST
CIPVSMII	Insert CI_MULT_ITEM
CIPVSMIV	Foreign Key validation for CI_MULT_ITEM
CIPVSPCI	Insert CI_SP_CHAR
CIPVSPCV	Foreign Key validation for CI_SP_CHAR
CIPVSPGI	Insert CI_SP_GEO
CIPVSPGV	Foreign Key validation for CI_SP_GEO
CIPVSPMI	Insert CI_SP_MULT_ITEM
CIPVSPMV	Foreign Key validation for CI_SP_MULT_ITEM
CIPVSPOI	Insert CI_SP_OP_AREA
CIPVSPOV	Foreign Key validation for CI_SP_OP_AREA
CIPVSPPI	Insert CI_SP
CIPVSPPK	Generate CI_SP keys
CIPVSPRI	Insert CI_SP_RTE
CIPVSPRV	Foreign Key validation for CI_SP_RTE
CIPVSQTI	Insert CI_BSEG_SQ
CIPVSQTV	Foreign Key validation for CI_BSEG_SQ
CIPVSRCI	Insert CI_SP_RTE_CFG
CIPVSRCV	Foreign Key validation for CI_SP_RTE_CFG
CIPVSRLI	Insert CI_SA_REL
CIPVSRLK	Generate CI_SA_REL keys
CIPVSRLV	Foreign Key validation for CI_SA_REL
CIPVSRRI	Insert CI_BSEG_READ
CIPVSRRV	Foreign Key validation for CI_BSEG_READ
CIPVSSCI	Insert CI_SA_SP_CHAR

Batch Control	Description
CIPVSSCV	Foreign Key validation for CI_SA_SP_CHAR
CIPVSSFI	Insert CI_SA_SP_FA
CIPVSSFV	Foreign Key validation for CI_SA_SP_FA
CIPVSSPK	Generate CI_SA_SP keys
CIPVSTMI	Insert CI_SA_TOU_MAP
CIPVSTMV	Foreign Key validation for CI_SA_TOU_MAP
CIPVSVAI	Insert CI_SA
CIPVSVAK	Generate CI_SA keys
CIPVTBVI	Insert CI_TOU_BF_VAL
CIPVTBVV	Foreign Key validation for CI_TOU_BF_VAL
CIPVTCVI	Insert CI_TOU_CONT_VAL
CIPVTCVV	Foreign Key validation for CI_TOU_CONT_VAL
CIPVTDSI	Insert CI_TOU_DATA_SET
CIPVTDSK	Generate CI_TOU_DATA_SET keys
CIPVTMAI	Insert CI_TOU_MAP
CIPVTMAK	Generate CI_TOU_MAP keys
CIPVTMLI	Insert CI_TOU_MAP_L
CIPVTMLV	Foreign Key validation for CI_TOU_MAP_L
CIPVTNCI	Insert CI_PAY_TNDR_CHAR
CIPVTNCV	Foreign Key validation for CI_PAY_TNDR_CHAR
CIPVTNDI	Insert CI_PAY_TNDR
CIPVTNDK	Generate CI_PAY_TNDR keys
CIPVTNDV	Foreign Key validation for CI_PAY_TNDR
CIPVTODI	Insert CI_TOU_DATA
CIPVTOFV	Foreign Key validation for CI_TOU_DATA
CIPVTRNI	Insert CI_TREND
CIPVTRNV	Foreign Key validation for CI_TREND
CIPVWECI	Insert CI_WF_EVT_CTXT
CIPVWECV	Foreign Key validation for CI_WF_EVT_CTXT
CIPVWEDI	Insert CI_WF_EVT_DEP
CIPVWEDV	Foreign Key validation for CI_WF_EVT_DEP
CIPVWEVI	Insert CI_WF_EVT
CIPVWEVV	Foreign Key validation for CI_WF_EVT

Batch Control	Description
CIPVWOCI	Insert CI_WO_EVT_CC
CIPVWOCV	Foreign Key validation for CI_WO_EVT_CC
CIPVWOPI	Insert CI_WO_PROC
CIPVWOPK	Generate CI_WO_PROC keys
CIPVWOPV	Foreign Key validation for CI_WO_PROC
CIPVWOSI	Insert CI_WO_PROC_SA
CIPVWOSV	Foreign Key validation for CI_WO_PROC_SA
CIPVWOVI	Insert CI_WO_EVT
CIPVWOVV	Foreign Key validation for CI_WO_EVT
CIPVWPCI	Insert CI_WF_PROC_CTXT
CIPVWPCV	Foreign Key validation for CI_WF_PROC_CTXT
CIPVWPRI	Insert CI_WF_PROC
CIPVWPRK	Generate CI_WF_PROC keys
CIPVWPRV	Foreign Key validation for CI_WF_PROC
CNV-BCG	Conversion-clear balance control ID
CNV-ADM	Conversion-create ADM triggers
CNV-BAL	Conversion-tidy balances

Validation Batch Control Application Services

The following batch controls use the C1-VALIDATE application service:

Batch Control	Description
VAL-ACCT	Validate account
VAL-BCHG	Validate billable charge
VAL-CEVT	Validate contract option event
VAL-CFG	Validate meter configuration
VAL-COLL	Validate collection process
VAL-COP	Validate contract option
VAL-DCL	Validate declaration
VAL-DTST	Validate device test
VAL-FA	Validate field activity
VAL-FO	Validate field order
VAL-IDS	Validate interval data set
VAL-INPF	Validate interval profile
VAL-IRDS	Validate interval register data set
VAL-ITEM	Validate items
VAL-IVS	Validate interval value set
VAL-LL	Validate landlord
VAL-MTR	Validate meter
VAL-PER	Validate person
VAL-PREM	Validate premise
VAL-SA	Validate service agreement
VAL-SARL	Validate SA relationships
VAL-SCM	Validate service credit membership
VAL-SEVP	Validate severance process
VAL-SP	Validate service point
VAL-TDS	Validate TOU data set
VAL-TMAP	Validate TOU map
VAL-WFP	Validate workflow process
VAL-WOP	Validate write off process

Purge Batch Control Application Services

The following batch controls use the C1-PURGE application service:

Batch Control	Description
BCUP-PRG	Purge billable charge upload
FAUP-PRG	Purge completed FA upload
MRUP-PRG	Purge completed MR upload
PYUP-PRG	Purge completed payment upload
C1-PRGST	Purge Completed Service Tasks
TD-PURGE	Purge Completed ToDo Entries
XMLUP-PR	Purge completed XAI upload staging records
C1-PRGSY	Purge Sync Requests
BCUP-PRG	Purge billable charge upload

ILM Batch Control Application Services

The following batch controls use the C1-ILM application service:

Batch Control	Description
C1-ADCRL	ILM Crawler - Adjustment
C1-BLCRL	ILM Crawler - Bill
C1-BSCRL	ILM Crawler - Bill Segment
C1-BCCRL	ILM Crawler - Billable Charge
C1-CACRL	ILM Crawler - Case
C1-CRCRL	ILM Crawler - Customer Relationship Request
C1-FACRL	ILM Crawler - Field Activity
C1-MECRL	ILM Crawler - Match Event
C1-MRCRL	ILM Crawler - Meter Read
C1-ORCRL	ILM Crawler - Order
C1-PECRL	ILM Crawler - Payment Event
C1-URCRL	ILM Crawler - Usage Request

Appendix G

Upgrades to the Oracle Utilities Application Framework Database

This section describes the database upgrade process for the Oracle Utilities Application Framework database since the last release. It highlights changes made to the administrative tables and how those changes should be applied to the data in order for your current database to work with the Oracle Utilities Application Framework application, and to preserve the business logic implemented in the previous version of the application. The changes that do not require data upgrade are not described in this document. The tasks that need to be performed after running the upgrade scripts are included.

> **Note**: Upgrade scripts do not automatically enable the newly added functionality by default. Please refer to the release notes for more information.

The section provides information on upgrading the Oracle Utilities Application Framework Database including:

Upgrading from Oracle Utilities Application Framework v4.3.0.1.0 to v4.3.0.2.0

New Tables

Table	Type of Table
F1_BUS_FLG	Business Flag
F1_BUS_FLG_CHAR	Business Flag Characteristic
F1_BUS_FLG_K	Business Flag Key Table
F1_BUS_FLG_LOG	Business Flag Log
F1_BUS_FLG_LOG_PARM	Business Flag Log Parameter
F1_BUS_FLG_REL	Business Flag Relationship
F1_BUS_FLG_REL_OBJ	Business Flag Related Object
F1_BUS_FLG_TYPE	Business Flag Type
F1_BUS_FLG_TYPE_ALG	Business Flag Type Algorithm
F1_BUS_FLG_TYPE_BUS_PROC	Business Flag Type / Business Process
F1_BUS_FLG_TYPE_CHAR	Business Flag Type Characteristic
F1_BUS_FLG_TYPE_L	Business Flag Type Language
F1_ETL_MP_CTRL	ETL Mapping Control

New Views

None

Dropped Tables

None

Unsupported Tables

None

Added Columns

Table	Column	Required
CI_BATCH_THD	LOG_FILE_NAME	
CI_MD_TBL	CHAR_ENTITY_FLG	N
F1_EXTSYS_OUTMSG_PROF	JSON_CONVRSN_METH_FLG	N
F1_EXTSYS_OUTMSG_PROF	REQ_SCHEMA_NAME	N
F1_EXTSYS_OUTMSG_PROF	RES_SCHEMA_NAME	N

Table	Column	Required
F1_MIGR_OBJ_ALG	ALG_PROC_TYPE_FLG	N

Dropped Columns

Table	Column
CI_MD_TBL	TBL_USAGE_FLG
CI_MD_TBL_FLD	FLD_USAGE_FLG

Unsupported Table Columns

None

Column Format Change

None

Upgrading from Oracle Utilities Application Framework v4.3.0.2.0 to v4.3.0.3.0

New Tables

Table	Type of Table
F1_LGCY_OBJ	Legacy Object
F1_PERF_TGT	Performance Target
F1_PERF_TGT_CHAR	Performance Target Characteristic
F1_PERF_TGT_L	Performance Target Language
F1_PERF_TGT_LOG	Performance Target Log
F1_PERF_TGT_LOG_PARM	Performance Target Log Parameter
F1_PERF_TGT_REL_OBJ	Performance Target Related Object
F1_PERF_TGT_TYPE	Performance Target Type
F1_PERF_TGT_TYPE_CHAR	Performance Target Type Characteristic
F1_PERF_TGT_TYPE_L	Performance Target Type Language
F1_STATS	Statistics Control
F1_STATS_CHAR	Statistics Control Characteristic
F1_STATS_L	Statistics Control Language
F1_STATS_LOG	Statistics Control Log
F1_STATS_LOG_PARM	Statistics Control Log Parameter
F1_STATS_REL_OBJ	Statistics Control Related Object
F1_STATS_SNPSHT	Statistics Snapshot
F1_STATS_SNPSHT_CHAR	Statistics Snapshot Characteristic
F1_STATS_SNPSHT_LOG	Statistics Snapshot Log
F1_STATS_SNPSHT_LOG_PARM	Statistics Snapshot Log Parameter
F1_STATS_SNPSHT_REL_OBJ	Statistics Snapshot Related Object
F1_SVC_CATALOG	Web Service Catalog

New Views
None

Dropped Tables
None

Unsupported Tables

None

Added Columns

Table	Column	Required
F1_EXTSYS_OUTMSG_PROF	NAMESPACE_FLG	N
F1_EXTSYS_OUTMSG_PROF	WSDL_FILE_NAME	N

Dropped Columns

None

Unsupported Table Columns

None

Column Format Change

None

Primary Key Change

None

Upgrading from Oracle Utilities Application Framework v4.3.0.3.0 to v4.3.0.4.0

New Tables

Table	Type of Table
F1_MIGR_REQ_INCL_REQ	Migration request Grouping

New Views

None

Dropped Tables

None

Unsupported Tables

None

Added Columns

Table	Column	Required
CI_BATCH_CTRL	APP_SVC_ID	Y
CI_XAI_RCVR_CTX	SEQNO	Y
CI_XAI_SNDR_CTX	SEQNO	Y
F1_IWS_SVC_ANN	SEQ_NUM	Y
F1_MIGR_REQ	MIGR_REQ_CAT_XFLG	N
F1_MIGR_REQ	MIGR_REQ_CLASS_FLG	Y
F1_MIGR_REQ_INSTR_ENTITY	COMMENT_LONG	N
F1_MIGR_REQ_INSTR_ENTITY	EXT_REFERENCE_ID	N

Dropped Columns

Table	Column
CI_XAI_RCVR_CTX	CTXT_VAL

Unsupported Table Columns

None

Column Format Change

Table Name	Column Name	From	To
F1_EXT_LOOKUP_VAL_CHAR	F1_EXT_LOOKUP_VALUE	VARCHAR2 (30)	VARCHAR2 (254)

Primary Key Change

Table	Primary Key Columns
CI_XAI_RCVR_CTX	XAI_RCVR_ID, SEQNO
CI_XA_SNDR_CTX	XAI_SENDER_ID, SEQNO

Upgrading from Oracle Utilities Application Framework v4.3.0.4.0 to v4.3.0.5.0

New Tables

Table	Description	Type of Table
F1_DEPLOYMENT	Deployment	Transaction
F1_DEPLOYMENT_ITEM	Deployment Item	Transaction
F1_DEPLOYMENT_ITEM_METADATA	Deployment Item Meta Data	Transaction
F1_DEPLOYMENT_PART	Deployment Part	Master
F1_DEPLOYMENT_PART_L	Deployment Part Language	Master
F1_DEPLOYMENT_TYPE	Deployment Type	Master
F1_DEPLOYMENT_TYPE_L	Deployment Type Language	Master
F1_DEPTYP_DEPPART	Deployment Type / Deployment Part	Transaction
F1_DEPTYP_MDT_TYPE	Deployment Type / MDT Type	Transaction
F1_DEPTYP_MSG_CAT	Deployment Type Message Category	Transaction
F1_DEPTYP_USR_GRP	Deployment Type User Group	Transaction
F1_MDT	Mobile Data Terminal	Transaction
F1_MDT_CHAR	Mobile Data Terminal Characteristics	Transaction
F1_MDT_TYPE	Mobile Data Terminal Type	Master
F1_MDT_TYPE_CHAR	Mobile Data Terminal Type Characteristics	Master
F1_MDT_TYPE_L	Mobile Data Terminal Type Language	Master
F1_MOB_COMP_CHAR	Mobile Component Characteristics	Admin - System
F1_MOB_COMP_CNT	Mobile Component Content	Admin - System
F1_MOBILE_COMPONENT	Mobile Component	Admin - System
F1_MOBILE_COMPONENT_L	Mobile Component Language	Admin - System

F1_REMOTE_MSG	Remote Message	Transaction
F1_REMOTE_MSG_CHAR	Remote Message Characteristics	Transaction
F1_REMOTE_MSG_LOG	Remote Message Log	Transaction
F1_REMOTE_MSG_LOG_PARM	Remote Message Log Parameters	Transaction
F1_WEB_CAT_L	Web Service Category Language	Admin - System
F1_WEB_CAT_INCL_SVC	Web Service Category - Included Services	Admin - System
F1_WEB_CAT	Web Service Category	Admin - System

Note that in addition, the following table was added to 4.3.0.4.0 via a hot fix, but was not included in 4.3.0.5.0 until after the final build and is therefore added as a hot fix. Clients upgrading to 4.3.0.5.0 may see that the table is dropped via the blueprint and then reinstated after applying the bug fixes.

Table	Description	Type of Table
F1_MIGR_OBJ_SQL_PK	Migration Object SQL Primary Key	Transaction

New Views

None

Dropped Tables

Table
F1_IWS_ANN_CHAR
F1_IWS_ANN_TYPE_CHAR

Unsupported Tables

None

Added Columns

Table	Column	Required
CI_MD_SVC	APP_SVC_ID	N
F1_OUTMSG	BO_XML_DATA_AREA	N
F1_OUTMSG_TYPE	OUTMSG_PRIOR_FLG	Y
F1_OUTMSG_TYPE	OWNER_FLG	N
F1_OUTMSG_TYPE	TYPE_BUS_OBJ_CD	N

Table	Column	Required
F1_OUTMSG_TYPE_L	OWNER_FLG	N

Dropped Columns
None

Unsupported Table Columns
None

Column Format Change
None

Primary Key Change
None

Index Changes
Index S1C675S1 for table F1_EXT_LOOKUP_VAL_CHAR has been renamed to F1C675S1.

Upgrading from Oracle Utilities Application Framework v4.3.0.5.0 to v4.3.0.6.0

New Tables

Table	Description	Table Type
F1_ATTACHMENT_K	Attachment Key	Transaction
F1_CRYPTO_KEY	Key Ring Key	Admin
F1_CRYPTO_KEY_RING	Key Ring	Admin
F1_CRYPTO_KEY_RING_L	Key Ring Language	Admin
F1_CRYPTO_KEY_RING_LOG	Key Ring Log	Admin
F1_CRYPTO_KEY_RING_LOG_PARM	Key Ring Log Parameter	Admin
F1_CUBE_TYPE	Cube Type	Admin
F1_CUBE_TYPE_L	Cube Type Language	Admin
F1_CUBE_VIEW	Cube View	Transaction
F1_CUBE_VIEW_L	Cube View Language	Transaction
F1_CUBE_VIEW_LOG	Cube View Log	Transaction
F1_CUBE_VIEW_LOG_PARM	Cube View Log Parameters	Transaction
F1_DEPLOYMENT_K	Deployment Key	Transaction
F1_ERASURE_SCHED	Object Erasure Schedule	Transaction
F1_ERASURE_SCHED_K	Object Erasure Schedule Key	Transaction
F1_ERASURE_SCHED_LOG	Object Erasure Schedule Log	Transaction
F1_ERASURE_SCHED_LOG_PARM	Object Erasure Schedule Log Parameter	Transaction
F1_MDT_K	Mobile Data Terminal Key	Transaction
F1_MIGR_OBJ_SQL_PK	Migration Object SQL Primary Key	Transaction
F1_PROC_DEFN	Process Flow Type	Admin
F1_PROC_DEFN_L	Process Flow Type Language	Admin
F1_PROC_NEXT_PANEL	Next Panel	Admin
F1_PROC_PANEL	Process Panel	Admin
F1_PROC_STORE	Process Flow	Transaction

Table	Description	Table Type
F1_PROC_STORE_DTL_ELEMENTS	Process Flow Detail Elements	Transaction
F1_PROC_STORE_K	Process Flow Key	Transaction
F1_PROC_STORE_LOG	Process Flow Log	Transaction
F1_PROC_STORE_LOG_PARM	Process Flow Log Parameters	Transaction
F1_REMOTE_MSG_K	Mobile Remote Message Key	Transaction
F1_STATS_SNPSHT_K	Statistics Snapshot Key	Transaction

Note that the following tables have system generated keys but do not have a separate key table. Per the new table list, the key tables are provided and these tables are updated accordingly.

Table	Description
F1_ATTACHMENT	Attachment
F1_DEPLOYMENT	Deployment
F1_MDT	Mobile Data Terminal
F1_REMOTE_MSG	Mobile Remote Message
F1_STATS_SNPSHT	Statistics Snapshot

New Views

None

Dropped Tables

Table
F1_IWS_SVC_OPER_L

Unsupported Tables

The table below has been added for future functionality and is not currently in use.

Table
F1_CRYPTO_KEY_RING_LINK

Added Columns

Table	Column	Required
CI_BATCH_RUN	END_DTTM	N

Table	Column	Required
CI_BATCH_RUN	START_DTTM	N
CI_COUNTRY	ADDR1_USG_FLG	Y
CI_COUNTRY	ADDR2_USG_FLG	Y
CI_COUNTRY	ADDR3_USG_FLG	Y
CI_COUNTRY	ADDR4_USG_FLG	Y
CI_COUNTRY	CITY_USG_FLG	Y
CI_COUNTRY	COUNTY_USG_FLG	Y
CI_COUNTRY	GEO_CODE_USG_FLG	Y
CI_COUNTRY	HOUSE_TYPE_USG_FLG	Y
CI_COUNTRY	IN_CITY_LIM_USG_FLG	Y
CI_COUNTRY	NUM1_USG_FLG	Y
CI_COUNTRY	NUM2_USG_FLG	Y
CI_COUNTRY	POSTAL_USG_FLG	Y
CI_COUNTRY	STATE_USG_FLG	Y
F1_ATTACHMENT	ATTACHMENT_EXT_ID	N
F1_ATTACHMENT	COMMENT_LONG	N
F1_IWS_SVC	RESOURCE_CAT_XFLG	N
F1_IWS_SVC	WEB_SVC_CLASS_FLG	Y
F1_IWS_SVC_OPER	RESOURCE_URI	N
F1_IWS_SVC_OPER	REST_HTTP_METHOD_FLG	N
F1_SVC_CATALOG	WEB_SVC_CLASS_FLG	Y

Dropped Columns

None

Column Format Change

None

Primary Key Change

None

Index Changes

None

Upgrading from Oracle Utilities Application Framework v4.3.0.6.0 to v4.4.0.0.0

New Tables
None

New Views
None

Dropped Tables
None

Unsupported Tables
None

Added Columns
None

Dropped Columns
None

Column Format Change
None

Primary Key Change
None

Index Changes
create index XT039S8 on CI_TD_ENTRY(ENTRY_STATUS_FLG,TD_TYPE_CD,MESSAGE_CAT_NBR,MESSAGE_NBR)

Appendix H

Oracle Application Framework System Table Guide

This section lists the system tables owned by the Oracle Utilities Application Framework V4.4.0.0 and explains the data standards of the system tables. The data standards are required for the installation of Oracle Utilities Application Framework, development within the Oracle Utilities Application Framework, and the configuration and customization of Oracle Utilities products. Adhering to the data standards is a prerequisite for seamless upgrade to future releases.

This section includes:

About the Application Framework System Tables

System tables are a subset of the tables that must be populated at the time the product is installed. They include metadata and configuration tables. The data stored in the system tables are the information that Oracle Utilities Application Framework product operations are based on.

As the product adds more functionality, the list of system tables can grow. The complete list of the system tables can be found in the System Table List section.

System Table Standards

System table standards must be observed for the following reasons:

- The product installation and upgrade process and customer modification data extract processes depend on the data prefix and owner flag values to determine the system data owned by each product.
- The standards ensure that there will be no data conflict in the product being developed and the future Oracle Utilities Application Framework release. They also ensure that there will be no data conflict between customer modifications and future Oracle Utilities product releases.
- The data prefix is used to prevent test data from being released to production.

  Developer's Note: All test data added to the system data tables must be prefixed by ZZ (all upper case) in order for the installation and upgrade utility to recognize them as test data.

Guidelines for System Table Updates

This section describes guidelines regarding the updating of the system table properties.

Business Configuration Tables

The majority of data in the tables in this group belongs to the customer. But these tables are shipped with some initial data in order for the customer to login to the system and begin configuring the product. Unless specified otherwise, the initial data is maintained by Oracle Utilities Application Framework and subject to subsequent upgrade.

Application Security and User Profile

These tables define the access rights of a User Group to Application Services and Application Users.

Properties	Description
Tables	SC_ACCESS_CNTL, SC USER, SC_USR_GRP_PROF, SC_USR_GRP_USR, SC_USER_GROUP, SC_USER_GROUP_L
Initial Data	User Group ALL_SERVICES and default system user SYSUSER. Upon installation the system default User Group ALL_SERVICES is given unrestricted accesses to all services defined in Oracle Utilities Application Framework.

Developer's Note: When a new service is added to the system, all actions defined for the service must be made available to the User Group ALL_ SERVICES.

Currency Code

The ISO 4217 three-letter codes are taken as the standard code for the representation of each currency.

Properties	Description
Tables	CI_CURRENCY_CD, CI_CURRENCY_CD_L
Initial Data	United States Dollar (USD)

Display Profile

The Display Profile Code is referenced in the User (SC_USER) table.

Properties	Description
Tables	CI_DISP_PROF, CI_DISP_PROF_L

Properties	Description
Initial Data	North America (NORTHAM) and Europe (EURO) and HIJRI Format (HIJRI) **Configuration Note**: In order to use HIJRI Format display profile, additional configuration is needed to define the mappings between Hijri dates and Gregorian dates. Refer to the Display Profile documentation for more information.

Configuration Note: In order to use HIJRI Format display profile, additional configuration is needed to define the mappings between Hijri dates and Gregorian dates.

Refer to the Display Profile documentation for more information.

Installation Options

Installation Option has only one row that is shipped with the initial installation of the Oracle Utilities Application Framework. The updatable columns in these tables are customer data and will not be overridden by the upgrade process unless a special script is written and included in the upgrade process.

Properties	Description
Tables	F1_INSTALLATION, CI_INSTALL_ALG, CI_INSTALL_MSG, CI_INSTALL_MSG_L, CI_INSTALL_PROD
Initial Data	Option 11111

Developer's Note: The system data owner of an environment is defined in the Installation Option. This Owner Flag value is stamped on all system data that is added to this environment. The installation default value is Customer Modification (CM). This value must be changed in the base product development environments.

Language Code

Language Code must be a valid code defined in ISO 639-2 Alpha-3. Adding a new language code to the table without translating all language dependent objects in the system can cause errors when a user chooses the language.

Properties	Description
Tables	CI_LANGUAGE
Initial Data	English (ENG)

Time Zone

The installation options require a valid time zone. A value for UTC (Coordinated Universl Time) is provided. Implementations should define the appropriate time zone and update the installation option value accordingly.

Properties	Description
Tables	CI_TIME_ZONE, CI_TIME_ZONE_L
Initial Data	UTC

To Do Priority and Role

New To Do Types released will be linked to the default To Do Role and set to the product assigned priority value initially. These initial settings can be overridden by the implementation.

Properties	Description
Tables	CI_ROLE(L), CI_TD_VAL_ROLE
Initial Data	F1_DFLT

Development and Implementation System Tables

This section defines the standards for the system tables that contain data for application development. The data in these tables implement business logic and UI functions shared by various products and product extensions in the same database.

Standards

When adding new data, the owner flag value of the environment must prefix certain fields of these tables. For example, when a developer adds a new algorithm type to an Oracle Utilities Customer Care and Billing environment, C1 should prefix the new Algorithm Type code. The fields that are subject to this rule are listed in Standard Data Fields property.

The data that is already in these tables cannot be modified if the data owner is different than the environment owner. This prevents the developers from accidentally modifying system data that belongs to the Oracle Utilities Application Framework or the base products. However, some fields are exempt from this rule and can be modified by Customer Modification. These fields are listed in the Customer Modification Fields property.

Note that the system supports a system upgrade rule called Override Owner flag. If duplicate data rows (data row with same primary key values) are found at the time of upgrade, the owner flag values will get overridden. The lower level application system data will override the upper level system data. For example, F1 overrides C1, F1&C1 override CM, and so on. This rule will be applied to the following tables: CI_CHAR_ENTITY, CI_MD_MO_ALG, C1_PORTAL_OPT, F1_BUS_OBJ_ALG, F1_BUS_OBJ_STATUS_ALG, CI_MD_MO_OPT, F1_BUS_OBJ_OPT, F1_BUS_OBJ_STATUS_OPT, F1_BUS_OBJ_STATUS, F1_BUS_OBJ_STATUS_L

Algorithm Type

Properties	Description
Tables	CI_ALG_TYPE, CI_ALG_TYPE_L, CI_ALG_TYPE_PRM, CI_ALG_TYPE_PRM_L
Standard Data Fields	Algorithm Type (ALG_TYPE_CD)
Customer Modification	None

Algorithm

Properties	Description
Tables	CI_ALG, CI_ALG_L, CI_ALG_PARM, CI_ALG_VER
Standard Data Fields	Algorithm (ALG_CD)
Customer Modification	None

Application Security

Properties	Description
Tables	SC_APP_SERVICE, SC_APP_SERVICE_L, CI_APP_SVC_ACC
Standard Data Fields	Application Service ID (APP_SVC_ID).
Customer Modification	None

Batch Control

Properties	Description
Tables	CI_BATCH_CTRL, CI_BATCH_CTRL_L, CI_BATCH_CTRL_P, CI_BATCH_CTRL_P_L
Standard Data Fields	Batch Process (BATCH_CD), Program Name (PROGRAM_NAME)

Properties	Description
Customer Modification	Next Batch Number (NEXT_BATCH_NBR), Last Update Instance (LAST_UPDATE_INST), Last Update Date time (LAST_UPDATE_DTTM) and the batch process update these columns. Time Interval (TIMER_INTERVAL), Thread Count (BATCH_THREAD_CNT), Maximum Commit Records (MAX_COMMIT_RECS), User (USER_ID), Language (LANGUAGE_CD), Email Address (EMAILID), Start program debug tracing (TRC_PGM_STRT_SW), End Program Debug trace (TRC_PGM_END_SW), SQL debug tracing (TRC_SQL_SW) and Standard debug tracing (TRC_STD_SW) on CI_BATCH_CTRL Table. Batch Parameter Value (BATCH_PARM_VAL) and Security flag (TEXT_SECURITY_FLG) on Batch Control Parameters Table (CI_BATCH_CTRL_P)

Business Object

Properties	Description
Tables	F1_BUS_OBJ, F1_BUS_OBJ_L, F1_BUS_OBJ_ALG, F1_BUS_OBJ_OPT, F1_BUS_OBJ_STATUS, F1_BUS_OBJ_STATUS_L, F1_BUS_OBJ_STATUS_ALG, F1_BUS_OBJ_STATUS_OPT, F1_BUS_OBJ_STATUS_RSN, F1_BUS_OBJ_STATUS_RSN_L, F1_BUS_OBJ_STATUS_RSN_CHAR, F1_BUS_OBJ_TR_RULE, F1_BUS_OBJ_TR_RULE_L
Standard Data Fields	Business Object (BUS_OBJ_CD), Status Reason (BO_STATUS_REASON_CD)
Customer Modification	Batch Control (BATCH_CD), Alert (BO_ALERT_FLG), Sequence (SORT_SEQ5), Status Reason (STATUS_REASON_FLG) fields on Business Object Status Table (F1_BUS_OBJ_STATUS). Instance Control (INSTANCE_CTRL_FLG), Application Service (APP_SVC_ID) on Business Object Table (F1_BUS_OBJ). Status Reason Selection (STATUS_REASON_SELECT_FLG) on Status Reason Table (F1_BUS_OBJ_STATUS_RSN)

Business Service

Properties	Description
Tables	F1_BUS_SVC, F1_BUS_SVC_L
Standard Data Fields	Business Service (BUS_SVC_CD)
Customer Modification	Application Service (APP_SVC_ID)

Characteristics

Properties	Description
Tables	CI_CHAR_TYPE, CI_CHAR_TYPE_L, CI_CHAR_ENTITY, CI_CHAR_VAL, CI_CHAR_VAL_L
Standard Data Fields	Characteristic Type (CHAR_TYPE_CD) Characteristic Value (CHAR_VAL) on CI_CHAR_VAL If the characteristic type is customizable, Customer Modification can insert new characteristic values. CM must prefix when implementers introduce a new characteristic value.
Customer Modification	Adhoc Characteristic Value Validation Rule (ADHOC_VAL_ALG_CD), Allow Search by Characteristic Value (SEARCH_FLG)

Configuration Migration Assistant

Properties	Description
Tables	F1_MIGR_PLAN,F1_MIGR_PLAN_L,F1_MIGR_PLAN_INSTR,F1_MIGR_PLAN_INSTR_L, F1_MIGR_PLAN_INSTR_ALG, F1_MIGR_REQ, F1_MIGR_REQ_L, F1_MIGR_REQ_INSTR, F1_MIGR_REQ_INST R_L, F1_MIGR_REQ_INSTR_ENTITY, F1_MIGR_REQ_INCL_REQ
Standard Data Fields	Migration Plan Code (MIGR_PLAN_CD), Migration Request Code (MIGR_REQ_CD)
Customer Modification	None

Data Area

Properties	Description
Tables	F1_DATA_AREA, F1_DATA_AREA_L
Standard Data Fields	Data Area Code (DATA_AREA_CD)
Customer Modification	None

Deployment Part

Properties	Description
Tables	F1_DEPLOYMENT_PART, F1_DEPLOYMENT_PART_L, F1_DEPLOYMENT_ITEM
Standard Data Fields	Deployment ID (F1_DEPLOYMENT_ID)
Customer Modification	None

Display Icon

Properties	Description
Tables	CI_DISP_ICON, CI_DISP_ICON_L
Standard Data Fields	Display Icon Code (DISP_ICON_CD)
Customer Modification	None

Extendable Lookup

Properties	Description
Tables	F1_EXT_LOOKUP_VAL, F1_EXT_LOOKUP_VAL_L, F1_EXT_LOOKUP_VAL_CHAR
Standard Data Fields	Business Object (BUS_OBJ_CD), Extendable Lookup Value (F1_EXT_LOOKUP_VALUE)
Customer Modification	Business Object Data Area (BO_DATA_AREA) Override Description (DESCR_OVRD) on Extendable Lookup Field Value Language Table (F1_EXT_LOOKUP_VAL_L) **Note:** When the product releases base owned records in Extendable Lookup, if there are additional elements the business object will map the element to the BO_DATA_AREA if the value is allowed to be modified by an implementation.

Foreign Key Reference

Properties	Description
Tables	CI_FK_REF, CI_FK_REF_L
Standard Data Fields	FK reference code (FK_REF_CD)
Customer Modification	Info Program Name (INFO_PRG), Zone (ZONE_CD)

Inbound Web Service

Properties	Description
Tables	F1_IWS_SVC_L, F1_IWS_SVC, F1_IWS_SVC_OPER_L, F1_IWS_SVC_OPER, F1_IWS_ANN_L, F1_IWS_ANN_PARM, F1_IWS_ANN, F1_IWS_ANN_TYPE_L, F1_IWS_ANN_TYPE, F1_IWS_ANN_TYPE_PARM, F1_IWS_ANN_TYPE_PARM_L
Standard Data Fields	Webservice Name (IN_SVC_NAME), Annotation (ANN_CD), Annotation Type (ANN_TYPE_CD)

Properties	Description
Customer Modification	Debug (DEBUG_SW), Active (ACTIVE_SW), Trace (TRACE_SW), Request XSL (REQUEST_XSL), Response XSL (RESPONSE_XSL)

Unsupported Metadata

Properties	Description
Tables	F1_LGCY_OBJ
Standard Data Fields	Object ID (LGCY_OBJ_ID)
Customer Modification	None

Lookup

Properties	Description
Tables	CI_LOOKUP_FIELD, CI_LOOKUP_VAL, CI_LOOKUP_VAL_L
Standard Data Fields	Field Name (FIELD_NAME) • A lookup field name must have corresponding field metadata. The name of the lookup field column must be assigned to avoid conflicts among different products. If you follow the standards for database field names, a Customer Modification lookup field name will be automatically Customer Modification prefixed. Field Value (FIELD_VALUE) • If a lookup field is customizable, Customer Modification can insert new lookup values. X or Y must prefix when implementers introduce a new lookup value. • Product development may add lookup values to a Oracle Utilities Application Framework owned lookup field's value. When extended new value is added, the Owner Flag is used to prefix the value. For example: When the Oracle Utilities Customer Care and Billing product adds a new value to the algorithm entity flag (ALG_ENTITY_FLG), it is prefixed with C1.
Customer Modification	Override Description (DESCR_OVRD) on Lookup Field Value Language Table (CI_LOOKUP_VAL_L)

Map

Properties	Description
Tables	F1_MAP, F1_MAP_L
Standard Data Fields	UI Map (MAP_CD)
Customer Modification	None

Managed Content

Properties	Description
Tables	F1_MANAG_CONTENT, F1_MANAG_CONTENT_L
Standard Data Fields	Managed Content (MANAG_CONTENT_CD)
Customer Modification	None

Messages

Properties	Description
Tables	CI_MSG_CATEGORY, CI_MSG_CATEGORY_L, CI_MSG, CI_MSG_L

Properties	Description
Standard Data Fields	Message Category (MESSAGE_CAT_NBR) • Messages are grouped in categories and each category has message numbers between 1 and 99999. A range of message categories is assigned to a product. An implementation may only use categories assigned for customization use. • Implementer Message Categories are 80000 and 90000 • Reserved for Tests - 99999 Message Number (MESSAGE_NBR) for message categories • Message numbers below 1000 are reserved for common messages. Implementers must not use message numbers below 1000. Message Number (MESSAGE_NBR) for Java message categories • Subsystem Standard Messages - 00001 thru 02000 • Reserved - 02001 thru 09999 • Published Messages - 10001 thru 11000 • Package Messages - 10001 thru 90000 • Reserved - 90001 thru 99999 • Each package is allocated 100 message numbers, each starting from 101. • Published Messages are messages that are special-interest messages that implementations need to know about and are therefore published in the user docs. Examples of these include messages that are highly likely to be changed for an implementation, or messages that are embedded into other texts/messages and therefore the message number is never shown • Reserved message number ranges are for future use and therefore must not be used by all products.
Customer Modification	Override Description (DESCRLONG_OVRD), Message Text Override (MESSAGE_TEXT_OVRD)

Meta Data - Table and Field

Properties	Description
Tables	CI_MD_TBL, CI_MD_TBL_FLD, CI_MD_TBL_L, CI_MD_TBL_FLD_L, CI_MD_FLD, CI_MD_FLD_L, F1_DB_OBJECTS_REPO
Standard Data Fields	Table Name (TBL_NAME) • Table names must match with the physical table name or view name in the database. • Field Name (FLD_NAME) Field name must match with the physical column name in the database unless the field is a work field. Field name does not have to follow the prefixing standard unless the field is a work field or customer modification field. • F1_DB_OBJECTS_REPO Table stores information about Indexes, Sequences, Triggers and other database objects excluding Tables and Fields (as they are already stored in the other Metadata tables)
Customer Modification	AuditSwitches(AUDIT_INSERT_SW,AUDIT_UPDATE_SW, AUDIT_DELETE_SW), Override label (OVRD_LABEL) on MD Table Field Table (CI_MD_TBL_FLD). Audit Program Name (AUDIT_PGM_NAME), Audit Table Name (AUDIT_TBL_NAME), Audit Program Type (AUDIT_PGM_TYPE_FLG), Key Validation (KEY_VALIDATION_FLG) and Caching strategy (CACHE_FLG) on MD Table (CI_MD_TBL). Override Label (OVRD_LABEL) and Customer Specific Description (DESCRLONG_OVRD) on Field Table.

Meta Data - Constraints

Properties	Description
Tables	CI_MD_CONST, CI_MD_CONST_FLD
Standard Data Fields	Constraint Id (CONST_ID) • Index Name for Primary Constraints • <Index Name>Rnn for Foreign Key Constraints Where • nn: integer, 01 through 99
Customer Modification	None

Meta Data - Menu

Menus can be extended to support multiple products by adding a new menu line to an existing menu. The sequence number on the menu line language table

(CI_MD_MENU_LINE_L) determines the order the menu lines appear. Within the same sequence, alphabetic sorting is used.

Properties	Description
Tables	CI_MD_MENU, CI_MD_MENU_L, CI_MD_MENU_ITEM, CI_MD_MENU_ITEM_L, CI_MD_MENU_LINE, CI_MD_MENU_LINE_L
Standard Data Fields	Menu Name (MENU_NAME), Menu Item Id (MENU_ITEM_ID), Menu Line Id (MENU_LINE_ID)
Customer Modification	Override Label (OVRD_LABEL) on Menu Line Language Table (CI_MD_MENU_LINE_L)

Meta Data - Program, Location and Services

Properties	Description
Tables	CI_MD_PRG_COM, CI_MD_PRG_LOC, CI_MD_SVC, CI_MD_SVC_L, CI_MD_SVC_PRG, CI_MD_PRG_MOD, CI_MD_PRG_EL_AT, CI_MD_PRG_ELEM, CI_MD_PRG_SEC, CI_MD_PRG_SQL, CI_MD_PRG_VAR, CI_MD_PRG_TAB
Standard Data Fields	Program Component Id (PROG_COM_ID), Location Id (LOC_ID), Program Component Name (PROG_COM_NAME), Service Name (SVC_NAME), Navigation Key (NAVIGATION_KEY)
Customer Modification	User Exit Program Name (USER_EXIT_PGM_NAME) on Program Components Table (CI_MD_PRG_COM),

Meta Data - Maintenance Object

Properties	Description
Tables	CI_MD_MO, CI_MD_MO_L, CI_MD_MO_TBL, CI_MD_MO_OPT, CI_MD_MO_ALG
Standard Data Fields	Maintenance Object (MAINT_OBJ_CD)
Customer Modification	None

Meta Data - Work Tables

Properties	Description
Tables	CI_MD_WRK_TBL, CI_MD_WRK_TBL_L, CI_MD_WRK_TBLFLD, CI_MD_MO_WRK
Standard Data Fields	Work Table Name (WRK_TBL_NAME)
Customer Modification	None

Meta Data - Search Object

Properties	Description
Tables	CI_MD_SO, CI_MD_SO_L, CI_MD_SO_RSFLD, CI_MD_SO_RSFLDAT, CI_MD_SOCG, CI_MD_SOCG_FLD, CI_MD_SOCG_FLDAT, CI_MD_SOCG_L, CI_MD_SOCG_SORT
Standard Data Fields	Search Object (SO_CD)
Customer Modification	None

Mobile Component

Properties	Description
Tables	F1_MOBILE_COMPONENT, F1_MOBILE_COMPONENT_L, F1_MOB_COMP_CNT, F1_MOBILE_COMP_CHAR
Standard Data Fields	Mobile Component Code (F1_MOB_COMP_TYPE_CD)
Customer Modification	Expiration Days (F1_EXPIRATION_TIME_DUR)

Navigation Option

Properties	Description
Tables	CI_NAV_OPT, CI_NAV_OPT_L, CI_NAV_OPT_CTXT, CI_NAV_OPT_USG, CI_MD_NAV
Standard Data Fields	Navigation Option Code (NAV_OPT_CD), Navigation Key (NAVIGATION_KEY)
Customer Modification	None

Outbound Message Type

Properties	Description
Tables	F1_OUTMSG_TYPE, F1_OUTMSG_TYPE_L
Standard Data Fields	Outbound Message Type Code (OUTMSG_TYPE_CD)
Customer Modification	Priority (OUTMSG_PRIOR_FLG)

Portal and Zone

Properties	Description
Tables	CI_PORTAL, CI_PORTAL_L, CI_PORTAL_ZONE, CI_PORTAL_OPT, CI_ZONE, CI_ZONE_L, CI_ZONE_PRM, CI_ZONE_HDL, CI_ZONE_HDL_L, CI_ZONE_HDL_PRM, CI_ZONE_HDL_PRM_L, CI_UI_ZONE
Standard Data Fields	Portal Code (PORTAL_CD), Zone Code (ZONE_CD), Zone Type Code (ZONE_HDL_CD) • A new Zone can be added to the Product owned Portal Pages. • The existing Zones cannot be removed from the Product owned Portal Pages.
Customer Modification	Sort Sequence (SORT_SEQ) on Context Sensitive Zone Table (CI_UI_ZONE). Show on Portal Preferences (USER_CONFIG_FLG) on Portal Table (CI_PORTAL). Override Sequence (SORT_SEQ_OVRD) on Portal Zone Table (CI_PORTAL_ZONE). Customer Specific Description (DESCRLONG_OVRD) on Zone Language Table (CI_ZONE_L). Override Parameter Value (ZONE_HDL_PARM_OVRD) on Zone Type Parameters Table (CI_ZONE_HDL_PRM). Override Parameter Value (ZONE_PARM_VAL_OVRD) on Zone Parameters Table (CI_ZONE_PRM).

Process Flow Type

Properties	Description
Tables	F1_PROC_DEFN F1_PROC_DEFN_L F1_PROC_NEXT_PANEL F1_PROC_PANEL
Standard Data Fields	Process Flow Type (PROCESS_CD)
Customer Modification	None

Sequence

Properties	Description
Tables	CI_SEQ
Standard Data Fields	Sequence Name (SEQ_NAME)
Customer Modification	Sequence Number (SEQ_NBR) This field is updated by the application process and must be set to 1 initially.

Schema

Properties	Description
Tables	F1_SCHEMA
Standard Data Fields	Schema Name (SCHEMA_NAME)
Customer Modification	None

Script

Properties	Description
Tables	CI_SCR, CI_SCR_L, CI_SCR_CRT, CI_SCR_CRT_GRP, CI_SCR_CRT_GRP_L, CI_SCR_DA, CI_SCR_FLD_MAP, CI_SCR_PRMPT, CI_SCR_PRMPT_L, CI_SCR_STEP, CI_SCR_STEP_L
Standard Data Fields	Script (SCR_CD)
Customer Modification	None

To Do Type

Properties	Description
Tables	CI_TD_TYPE, CI_TD_TYPE_L, CI_TD_SRTKEY_TY, CI_TD_DRLKEY_TY, CI_TD_SRTKEY_TY_L
Standard Data Fields	To Do Type Code (TD_TYPE_CD)
Customer Modification	Creation Batch Code (CRE_BATCH_CD), Route Batch Code (RTE_BATCH_CD), Priority Flag (TD_PRIORITY_FLG) on To Do Type Table (CI_TD_TYPE)

Web Service Category

Properties	Description
Tables	F1_WEB_CAT, F1_WEB_CAT_L, F1_WEB_CAT_INCL_SVC
Standard Data Fields	Web Service Category code (WEB_SVC_CAT_CD)
Customer Modification	None

XAI Configuration

Properties	Description
Tables	CI_XAI_ADAPTER, CI_XAI_ADAPTER_L, CI_XAI_CLASS, CI_XAI_CLASS_L, CI_XAI_ENV_HNDL, CI_XAI_ENV_HNDL_L, CI_XAI_FORMAT, CI_XAI_FORMAT_L, CI_XAI_RCVR, CI_XAI_RCVR_L, CI_XAI_RCVR_CTX, CI_XAI_RCVR_RSP, CI_XAI_RCVR_RGRP, CI_XAI_SENDER, CI_XAI_SERNDER_L, CI_XAI_SNDR_CTX, CI_XAI_OPTION
Standard Data Fields	Adapter Id (XAI_ADAPTER_ID), Class Id (XAI_CLASS_ID), Envelope Handler Id (XAI_ENV_HNDL_ID), XAI Format Id (XAI_FORMAT_ID), Receiver Id (XAI_RCVR_ID), Sender Id (XAI_SENDER_ID)
Customer Modification	Option Value (OPTION_VALUE) on Message Option Table (CI_XAI_OPTION)

XAI Services

Properties	Description
Tables	CI_XAI_IN_SVC, CI_XAI_IN_SVC_L, CI_XAI_SVC_PARM
Standard Data Fields	XAI Inbound Service Id (XAI_IN_SVC_ID), XAI Inbound Service Name (XAI_IN_SVC_NAME)
Customer Modification	XAI Version (XAI_VERSION_ID), Trace (TRACE_SW), Debug (DEBUG_SW), Request XSL (INPUT_XSL), Response XSL (RESPONSE_XSL), Record XSL (RECORD_XSL and Post Error (POST_ERROR_SW) on XAI Inbound Service Table (CI_XAI_IN_SVC)

System Table List

This section contains names of system tables, upgrade actions, and a brief description of tables. The upgrade actions are explained below.

Keep (KP): The data in the table in the customer's database is kept untouched. No insert or delete is performed to this table by the upgrade process. The initial installation will add necessary data for the system

Merge (MG): The non-base product data in the table in the database is kept untouched. If the data belongs to the base product, any changes pertaining to the new version of the software are performed.

Refresh (RF): The existing data in the table is replaced with the data from the base product table. The product does not support customer specific data in these tables.

> **Note.** New product data is also inserted into tables marked as 'Merge'. If implementers add rows for a customer specific enhancement, it can cause duplication when the system data gets upgraded to the next version. We strongly recommend following the guidelines on how to use designated range of values or prefixes to segregate the implementation data from the base product data.

Table Name	Upgrade Action	Description
CI_ALG	MG	Algorithm
CI_ALG_L	MG	Algorithm Language
CI_ALG_PARM	MG	Algorithm Parameters
CI_ALG_TYPE	MG	Algorithm Type
CI_ALG_TYPE_L	MG	Algorithm Type Language
CI_ALG_TYPE_PRM	MG	Algorithm Type Parameter
CI_ALG_TYPE_PRM_L	MG	Algorithm Type Parameter Language
CI_ALG_VER	MG	Algorithm Version
CI_APP_SVC_ACC	MG	Application Service Access Mode
CI_BATCH_CTRL	MG	Batch Control
CI_BATCH_CTRL_ALG	MG	Batch Control Algorithm
CI_BATCH_CTRL_L	MG	Batch Control Language
CI_BATCH_CTRL_P	MG	Batch Control Parameters
CI_BATCH_CTRL_P_L	MG	Batch Control Parameters Language
CI_CHAR_ENTITY	MG	Characteristic Type Entity
CI_CHAR_TYPE	MG	Characteristic Type
CI_CHAR_TYPE_L	MG	Characteristic Type Language
CI_CHAR_VAL	MG	Characteristic Type Value
CI_CHAR_VAL_L	MG	Characteristic Type Value Language

Table Name	Upgrade Action	Description
CI_DISP_ICON	MG	Display Icon
CI_DISP_ICON_L	MG	Display Icon Language
CI_FK_REF	MG	Foreign Key Reference
CI_FK_REF_L	MG	Foreign Key Reference Language
CI_LANGUAGE	MG	Language Code
CI_LOOKUP_FIELD	MG	Lookup Field
CI_LOOKUP_VAL	MG	Lookup Field Value
CI_LOOKUP_VAL_L	MG	Lookup Field Value Language
CI_MD_CONST	MG	Constraints
CI_MD_CONST_FLD	MG	Constraint Fields
CI_MD_FLD	MG	Field
CI_MD_FLD_L	MG	Field Language
CI_MD_MENU	MG	Menu Information
CI_MD_MENU_IMOD	MG	Menu Item Module Maint
CI_MD_MENU_ITEM	MG	Menu Item
CI_MD_MENU_ITEM_L	MG	Menu Item Language
CI_MD_MENU_L	MG	Menu Language
CI_MD_MENU_LINE	MG	Menu Line
CI_MD_MENU_LINE_L	MG	Menu Line Language
CI_MD_MENU_MOD	MG	Menu Product Components
CI_MD_MO	MG	Maintenance Object
CI_MD_MO_ALG	MG	Maintenance Object Algorithm
CI_MD_MO_L	MG	Maintenance Object Language
CI_MD_MO_OPT	MG	Maintenance Object Option
CI_MD_MO_TBL	MG	Maintenance Object Table
CI_MD_MO_WRK	MG	Maintenance Object Work Tables
CI_MD_NAV	MG	Navigation Key
CI_MD_PRG_COM	MG	Program Components
CI_MD_PRG_ELEM	MG	UI Page Elements
CI_MD_PRG_EL_AT	MG	UI Page Element Attributes
CI_MD_PRG_LOC	MG	Program Location
CI_MD_PRG_MOD	MG	Program Module

Table Name	Upgrade Action	Description
CI_MD_PRG_SEC	MG	UI Page Sections
CI_MD_PRG_SQL	MG	MD SQL Meta Data
CI_MD_PRG_TAB	MG	UI Tab Meta Data
CI_MD_PRG_VAR	MG	Program Variable
CI_MD_SO	MG	Search Object
CI_MD_SOCG	MG	Search Object Criteria Group
CI_MD_SOCG_FLD	MG	Search Object Criteria Group Field
CI_MD_SOCG_FLDAT	MG	Search Criteria Group Field Attribute
CI_MD_SOCG_L	MG	Search Object Criteria Group Language
CI_MD_SOCG_SORT	MG	Search Criteria Group Result Sort Order
CI_MD_SO_L	MG	Search Object Language
CI_MD_SO_RSFLD	MG	Search Object Result Field
CI_MD_SO_RSFLDAT	MG	Search Object Result Field Attribute
CI_MD_SVC	MG	MD Service
CI_MD_SVC_L	MG	MD Service Language
CI_MD_SVC_PRG	MG	MD Service Program
CI_MD_TAB_MOD	MG	UI Tab Module
CI_MD_TBL	MG	MD Table
CI_MD_TBL_FLD	MG	MD Table Field
CI_MD_TBL_FLD_L	MG	MD Table Field Language
CI_MD_TBL_L	MG	MD Table Language
CI_MD_WRK_TBL	MG	Work Table
CI_MD_WRK_TBLFLD	MG	Work Table Field
CI_MD_WRK_TBL_L	MG	Work Table Language
CI_MSG	MG	Message
CI_MSG_CATEGORY	MG	Message Category
CI_MSG_CATEGORY_L	MG	Message Category Language
CI_MSG_L	MG	Message Language
CI_NAV_OPT	MG	Navigation Option
CI_NAV_OPT_CTXT	MG	Navigation Option Context
CI_NAV_OPT_L	MG	Navigation Option Language
CI_NAV_OPT_USG	MG	Navigation Option Usage

Table Name	Upgrade Action	Description
CI_PORTAL	MG	Portal
CI_PORTAL_L	MG	Portal Language
C1_PORTAL_OPT	MG	Portal Option
CI_PORTAL_ZONE	MG	Portal Zone
CI_SCR	MG	Script
CI_SCR_CRT	MG	Script Criteria
CI_SCR_CRT_GRP	MG	Script Criteria Group
CI_SCR_CRT_GRP_L	MG	Script Criteria Group Language
CI_SCR_DA	MG	Script Data Area
CI_SCR_FLD_MAP	MG	Script Field Mapping
CI_SCR_L	MG	Script Language
CI_SCR_PRMPT	MG	Script Prompt
CI_SCR_PRMPT_L	MG	Script Prompt Language
CI_SCR_STEP	MG	Script Step
CI_SCR_STEP_L	MG	Script Step Language
CI_SEQ	MG	Sequence
CI_TD_DRLKEY_TY	MG	To Do Type Drill Key
CI_TD_SRTKEY_TY	MG	To Do Type Sort Key
CI_TD_SRTKEY_TY_L	MG	To Do Type Sort Key Language
CI_TD_TYPE	MG	To Do Type
CI_TD_TYPE_L	MG	To Do Type Language
CI_UI_ZONE	MG	Context Sensitive Zone
CI_USR_NAV_LINK	MG	User Favorite Links
CI_XAI_ADAPTER	MG	XAI Adapter
CI_XAI_ADAPTER_L	MG	XAI Adapter Lang
CI_XAI_CLASS	MG	Message Class
CI_XAI_CLASS_L	MG	Message Class Language
CI_XAI_ENV_HNDL	MG	XAI Envelope Handler
CI_XAI_ENV_HNDL_L	MG	XAI Envelope Handler Language
CI_XAI_IN_SVC	MG	XAI Inbound Service
CI_XAI_IN_SVC_L	MG	XAI Inbound Service Language
CI_XAI_SVC_PARM	MG	XAI Inbound Service Parameters

Table Name	Upgrade Action	Description
CI_ZONE	MG	Zone
CI_ZONE_HDL	MG	Zone Type
CI_ZONE_HDL_L	MG	Zone Type Language
CI_ZONE_HDL_PRM	MG	Zone Type Parameters
CI_ZONE_HDL_PRM_L	MG	Zone Type Parameters Language
CI_ZONE_L	MG	Zone Language
CI_ZONE_PRM	MG	Zone Parameters
F1_BUS_OBJ	MG	Business Object
F1_BUS_OBJ_ALG	MG	Business Object Algorithm
F1_BUS_OBJ_L	MG	Business Object Language
F1_BUS_OBJ_OPT	MG	Business Object Option
F1_BUS_OBJ_STATUS	MG	Business Object Status
F1_BUS_OBJ_STATUS_ALG	MG	Business Object Status Algorithm
F1_BUS_OBJ_STATUS_L	MG	Business Object Status Language
F1_BUS_OBJ_STATUS_OPT	MG	Business Object Status Option
F1_BUS_OBJ_STATUS_RSN	MG	Status Reason
F1_BUS_OBJ_STATUS_RSN_L	MG	Status Reason Language
F1_BUS_OBJ_TR_RULE	MG	Business Object Transition Rule
F1_BUS_OBJ_TR_RULE_L	MG	Business Object Transition Rule Language
F1_BUS_SVC	MG	Business Service
F1_BUS_SVC_L	MG	Business Service Language
F1_DATA_AREA	MG	Data Area
F1_DATA_AREA_L	MG	Data Area Language
F1_DB_OBJECTS_REPO	MG	Database Objects Repository
F1_DEPLOYMENT_ITEM	MG	Deployment Part Item
F1_DEPLOYMENT_PART	MG	Deployment Part
F1_DEPLOYMENT_PART_L	MG	Deployment Part Language
F1_EXT_LOOKUP_VAL	MG	Extendable Lookup
F1_EXT_LOOKUP_VAL_L	MG	Extendable Lookup Language
F1_EXT_LOOKUP_VAL_CHAR	MG	Extendable Lookup Characteristics
F1_IWS_ANN	MG	Web Service Annotation

Table Name	Upgrade Action	Description
F1_IWS_ANN_L	MG	Web Service Annotation Language
F1_IWS_ANN_PARM	MG	Web Service Annotation Parameter
F1_IWS_ANN_TYPE	MG	Web Service Annotation Type
F1_IWS_ANN_TYPE_L	MG	Web Service Annotation Type Language
F1_IWS_ANN_TYPE_PARM	MG	Web Service Annotation Type Parm
F1_IWS_ANN_TYPE_PARM_L	MG	Web Service Annotation Type Parameter Language
F1_IWS_SVC	MG	Inbound Web Service
F1_IWS_SVC_L	MG	Inbound Web Service Language
F1_IWS_SVC_OPER	MG	Inbound Web Service Operations
F1_MANAG_CONTENT	MG	Managed Content
F1_MANAG_CONTENT_L	MG	Managed Content Language
F1_MAP	MG	UI Map
F1_MAP_L	MG	UI Map Language
F1_MIGR_PLAN	MG	Migration Plan
F1_MIGR_PLAN_INSTR	MG	Migration Plan Instruction
F1_MIGR_PLAN_INSTR_ALG	MG	Migration Plan Instruction Algorithm
F1_MIGR_PLAN_INSTR_L	MG	Migration Plan Instruction Language
F1_MIGR_PLAN_L	MG	Migration Plan Language
F1_MIGR_REQ	MG	Migration Request
F1_MIGR_REQ_INCL_REQ	MG	Migration Request Grouping
F1_MIGR_REQ_INSTR	MG	Migration Request Instruction
F1_MIGR_REQ_INSTR_ENTITY	MG	Migration Request Instruction Entity
F1_MIGR_REQ_INSTR_L	MG	Migration Request Instruction Language
F1_MIGR_REQ_L	MG	Migration Request Language
F1_MOBILE_COMPONENT	MG	Mobile Component
F1_MOBILE_COMPONENT_L	MG	Mobile Component Language
F1_MOB_COMP_CHAR	MG	Mobile Component Characteristics
F1_MOB_COMP_CNT	MG	Mobile Component Content
F1_OUTMSG_TYPE	MG	Outbound Message Type
F1_OUTMSG_TYPE_L	MG	Outbound Message Type Language

Table Name	Upgrade Action	Description
F1_PROC_DEFN	MG	Process Flow Type
F1_PROC_DEFN_L	MG	Process Flow Type Language
F1_PROC_NEXT_PANEL	MG	Next Panel
F1_PROC_PANEL	MG	Process Panels
F1_SCHEMA	MG	Schema
F1_WEB_CAT	MG	Web Service Category
F1_WEB_CAT_L	MG	Web Service Category Language
F1_WEB_CAT_INCL_SVC	MG	Web Service Category Included Services
SC_ACCESS_CNTL	MG	User Group Access Control
SC_APP_SERVICE	MG	Application Service
SC_APP_SERVICE_L	MG	Application Service Language
SC_USR_GRP_PROF	MG	User Group Profile
CI_CURRENCY_CD	KP	Currency Code
CI_CURRENCY_CD_L	KP	Currency Code Language
CI_DISP_PROF	KP	Display Profile
CI_DISP_PROF_L	KP	Display Profile Language
CI_TIME_ZONE	KP	Time Zone
CI_TIME_ZONE_L	KP	Time Zone Language
CI_USR_PORTAL	KP	User Portal
CI_XAI_JNDI_SVR	KP	XAI JNDI Server
CI_XAI_JNDI_SVR_L	KP	XAI JNDI Server Language
CI_XAI_OPTION	KP	Message Option
CI_XAI_SENDER	KP	Message Sender
CI_XAI_SENDER_L	KP	Message Sender Language
CI_XAI_SNDR_CTX	KP	Message Sender Context
F1_BUS_OBJ_STATUS_RSN_CHAR	KP	Status Reason Characteristic
F1_INSTALLATION	KP	Installation Option - Framework
SC_USER	KP	User
SC_USER_CHAR	KP	User Characteristic
SC_USER_GROUP	KP	User Group
SC_USER_GROUP_L	KP	User Group Language

Table Name	Upgrade Action	Description
SC_USR_GRP_USR	KP	User Group User
CI_MD_ATT_TY	RF	MD Element Attribute Type
CI_MD_AT_DTL	RF	MD Element Attribute Type Detail
CI_MD_AT_DTL_L	RF	MD Element Attribute Type Detail Language
CI_MD_CTL	RF	Generator Control
CI_MD_CTL_L	RF	Generator Control Language
CI_MD_CTL_TMPL	RF	Generator Control Template
CI_MD_ELTY	RF	MD Element Type
CI_MD_ELTY_AT	RF	Element Type Attributes
CI_MD_ELTY_L	RF	Element Type Language
CI_MD_LOOKUP_F	RF	MD Lookup Field
CI_MD_MSG	RF	MD Message
CI_MD_MSG_L	RF	MD Message Language
CI_MD_PDF	RF	Predefined Fields
CI_MD_PDF_VAL	RF	Predefined Values
CI_MD_SRC_TYPE	RF	Source Type
CI_MD_SRC_TYPE_L	RF	Source Type Language
CI_MD_TMPL	RF	Template
CI_MD_TMPL_ELTY	RF	Template Element Types
CI_MD_TMPL_L	RF	Template Language
CI_MD_TMPL_VAR	RF	Template Variable
CI_MD_TMPL_VAR_L	RF	Template Variable Language
CI_MD_VAR	RF	Variable
CI_MD_VAR_DTL	RF	Variable Detail
CI_MD_VAR_DTL_L	RF	Variable Detail Language
CI_XAI_EXECUTER	RF	XAI Executer
CI_XAI_EXECUTER_L	RF	XAI Executer Language
CI_XAI_FORMAT	RF	XAI Format
CI_XAI_FORMAT_L	RF	XAI Format Language
F1_LGCY_OBJ	RF	Unsupported Metadata