Oracle Utilities Meter Data Management

Database Administrator's Guide

Release 2.1.0.3

E38616-06

May 2015

(Updated October 2017)

ORACLE

Oracle Utilities Meter Data Management Database Administrator's Guide

E38616-06

Contents

Chapter 4

Chapter 5

Appendix A

Preface

This guide provides instructions for installing and maintaining the database for Oracle Utilities Meter Data Management. This preface contains these topics:

Audience

Oracle Utilities Meter Data Management Database Administrator's Guide is intended for database administrators who will be installing and maintaining the database for Oracle Utilities Meter Data Management.

Related Documents

The following documentation is included with this release.

Installation, Configuration, and Release Notes

- *Oracle Utilities Meter Data Management Release Notes*
- *Oracle Utilities Meter Data Management Quick Install Guide*
- *Oracle Utilities Meter Data Management Installation Guide*
- *Oracle Utilities Meter Data Management Database Administrator's Guide*
- *Oracle Utilities Meter Data Management Configuration Guide*

User Guides

- *Oracle Utilities Meter Data Management User's Guide*
- *Oracle Utilities Service and Measurement Data Foundation User's Guide*

Framework Documents

- *Oracle Utilities Application Framework Release Notes*
- *Oracle Utilities Application Framework Business Process Guide*
- *Oracle Utilities Application Framework Administration Guide*

Supplemental Documents

- *Oracle Utilities Meter Data Management Batch Server Administration Guide*

- *Oracle Utilities Meter Data Management Server Administration Guide*
- *Oracle Utilities Meter Data Management 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.
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.

Chapter 1

Database Overview

This section provides an overview of the Oracle Utilities Meter Data Management database, including:

- Supported Database Platforms
- Prerequisite Software for Database Server
- Database Maintenance Rules

Supported Database Platforms

Oracle Utilities Meter Data Management is supported on the following platforms:

Platform	Database Version
AIX 7.1 TL01 (POWER 64-bit)	Oracle Database Server 11.2.0.1+ (64-bit) Oracle Database Server 12.1.0.1+
Oracle Enterprise Linux 5.x/6.x/7.x x86_64 (64-bit) (Based on Red Hat Enterprise Linux (64-bit))*	Oracle Database Server 11.2.0.1+ (64-bit) Oracle Database Server 12.1.0.1+
Oracle Solaris 10 or Oracle Solaris 11 (SPARC 64-bit)	Oracle Database Server 11.2.0.1+ (64-bit) Oracle Database Server 12.1.0.1+
Windows Server 2008 R2 and 2012 R2 (x86_64 64-bit)	Oracle Database Server 11.2.0.1+ (64-bit) Oracle Database Server 12.1.0.1+

* Oracle Utilities Meter Data Management is tested and supported on the versions of Oracle Linux specified. Because Oracle Linux is 100% userspace-compatible with Red Hat Enterprise Linux, Oracle Utilities Meter Data Management also is supported on Red Hat Enterprise Linux for this release.

Note: Windows Server is **not** supported for Production environments. Wherever Windows Server is referenced within this guide, it is supported for Test or Development environments **only**.

The following Oracle Database Server Edition is supported:

- Oracle Database Enterprise Edition

  Note: Oracle Database Enterprise Edition and the Partitioning and Advanced Compression options are strongly recommended in all situations.

Refer to My Oracle Support for additional details.

Prerequisite Software for Database Server

The prerequisite software for the database component of Oracle Utilities Meter Data Management is as follows:

Oracle Database Server 11.2.0.1+ or 12.1.0.1+: This is required for installing the database component of the Oracle Utilities Meter Data Management product. The following version of the database server is supported:

- Oracle Database Enterprise Edition

The follow database feature is required:

- Oracle Locator

  Note: Oracle Spatial is not required.

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 of these components is 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 (that is, 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 in this document.

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 in anyway (altered, removed or added).
- 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 section provides the steps required to install or upgrade the Oracle Utilities Meter Data Management database, including:

- Installation Overview
- Initial Install
- Upgrade Install
- Demo Install

Installation Overview

> **Note:** Refer to the section Supported Database Platforms for the hardware and software versions required for the installation of Oracle Utilities Meter Data Management database components.

The following type of installation is available for Oracle Utilities Meter Data Management:

- **Initial Install** - a database with no demo data.
- **Upgrade Install** - a database upgrade.
- **Demo Install** - a database populated with demo data.

The database installation requires a supported version of the Java Development Kit to be installed on the Windows desktop where the install package is staged and run from. Refer to the Supported Platform section of the *Oracle Utilities Meter Data Management Installation Guide* for the required version of Java.

For an Initial Install or Demo Install you will create an empty database on the Unix or Windows server and then populate the database with data. For a database Upgrade Install you will upgrade your current Oracle Utilities Meter Data Management database.

Review the Storage.xml file prior to an Initial Install or Upgrade Install. Information in this file is used by ORADBI while installing and upgrading the Oracle Utilities Meter Data Management database objects.

For optimum storage allocation, database administrators should create multiple tablespaces with extents sized to store different types of tables/indexes. They can then edit this file before each upgrade and 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, securefile options, Advanced Compression, and parallel information are used only for new objects. Therefore, for initial installs, information for each object should be reviewed. For upgrades, only tablespace information for objects added in the current release needs to be reviewed. Be careful while editing the Storage.xml 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.

Initial Install

This section describes how to install the database components of Oracle Utilities Meter Data Management, including:

- Copying and Decompressing Install Media
- Creating the Database
- Installing the CISADM Schema
- Postinstallation Tasks

Copying and Decompressing Install Media

To copy and decompress the Oracle Utilities Meter Data Management database:

1. Download the Oracle Utilities Meter Data Management V2.1.0.3.0 Oracle database from the Oracle Software Delivery Cloud.
2. Create a temporary directory, such as C:\OUMDM\temp or /OUMDM/temp (referred to below as <TEMPDIR>. This directory must be located outside any current working Oracle Utilities application environment. All the files that are placed in this directory as a part of the installation can be deleted after a successful installation.
3. Copy the MDM-V2.1.0.3.0-Database.zip file from the downloaded package to the <TEMPDIR> directory.
4. Unzip the MDM-V2.1.0.3.0-Database.zip file to a temporary folder. This file contains the database components required to install the Oracle Utilities Meter Data Management database.

Creating the Database

Note: You must have Oracle Database Server 11.2.0.1 or higher version installed on your machine in order to create the database. The database can be created using Database Configuration Assistant (DBCA).

Using DBCA

For creating an Initial Install or production database it is recommended that you use the Database Configuration Assistant (DBCA). Once the database is created the instance configuration can be done according to the environment needs and based on your production recommendations.

The script for creating the product users is located under the relevant database version subdirectory of the DatabaseCreation directory.

1. You must create tablespace CISTS_01 before running the script for creating the product users.
2. Execute the …\MDM\DatabaseCreation\Unix\11g\users.sql after logging into the database as sys user, to create the product users.

Installing the CISADM Schema

You must install the Oracle Utilities Application Framework V4.2.0.3.0 prior to Oracle Utilities Meter Data Management. The files for Oracle Utilities Application Framework installation are located in the FW42030 folder. The installation process will prompt you for the following information:

- A database user that will own the application schema (for example, CISADM).
- A database user 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. (for example, CISUSER).
- A database user with read-only privileges to the objects in the application schema. (for 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. (for example, CIS_USER).
- A database role with read-only privileges to the objects in the application schema. (for example, CIS_READ).
- Location for jar files. (The Jar files are bundled with the database package.)
- Java Home (For example, C:\Java\jdk1.6.0_20)

To install the CISADM schema for Oracle Utilities Meter Data Management, follow these steps:

Install Oracle Utilities Application Framework V4.2.0.3.0

1. Run OraDBI.exe from the ..\FW42030\Install-Upgrade directory. Please run the utility from the command prompt.

   Note: Be sure to run OraDBI.exe from a Window 32-bit desktop that has the Oracle 11.2.0.1 client (or higher) and Java Development Kit Version 6.0 Update 20 or later. The database should already be listed in the local file tnsnames.ora

2. The utility prompts you to enter values for the following parameters:
   - Name of the target database: <DB NAME>
   - Name of the Database User Name: <CISADM>
   - Password of the User Name: <Password for CISADM>
   - Location of Java Home (e.g. C:\Java\jdk1.6.0_20): <Java Home>
   - Location of TUGBU Jar files (e.g. C:\FW42030\JarFiles): <..\FW42030\jarfiles>
   - Oracle User with read-write privileges to the Database Schema: <CISUSER>
   - Oracle User with read-only privileges to the Database Schema: <CISREAD>
   - Oracle database role with read-write privileges to the Database Schema: <CIS_USER>
   - Oracle database role with read-only privileges to the Database Schema: <CIS_READ>
   - Name of the owner of the Database Schema: <CISADM>
   - Enter the password for the CISADM schema (or hit ENTER to quit): <CISADM user's password>
   - Re-enter the password: <CISADM user's password>

After setting up roles and users, the utility continues upgrading schema and system data definitions. If an error occurs while executing an SQL or another utility, it logs and displays the error message and allows you to re-execute the current step.

Install Prerequisite Database Hot Fixes

Before installing Oracle Utilities Meter Data Management, you must install Oracle Utilities Framework Prerequisite database hot fixes as described below. Apply prerequisite Framework DB single fixes. See the *Oracle Utilities Meter data Management Installation Guide* for more details.

Apply the prerequisite Framework database fixes by running the CDXPatch.exe

The utility prompts you for the value of the following parameters:

- The target database type (O/M/D) [O]: O
- The name of the user that owns the database objects: <CISADM>
- The password for the user (in silent mode): <Password for CISADM user>
- The name of the Oracle database: <DB Name>

CDXPatch.exe can be executed by selecting it from Windows explorer, or by using a command line from a DOS window. Use the option "-h" to see the help.

After the patches are processed, if the utility prompts you to create security for new objects. enter "N" because security for new objects is generated in subsequent steps during installation of Oracle Utilities Meter Data Management.

Install Oracle Utilities Service and Measurement Data Foundation Database Component

Before installing Oracle Utilities Meter Data Management, you must install Oracle Utilities Service and Measurement Data Foundation.

The following procedure describes how to install the database component of Oracle Utilities Service and Measurement Data Foundation database.

1. Run OraDBI.exe from the ..\SMDF\Install-Upgrade directory. The utility prompts you to enter values for the following parameters:
    - Name of the target database: <DB NAME>
    - Name of the Database User Name: <CISADM>
    - Password of the User Name: <Password for CISADM>
    - Location of Java Home (e.g. C:\Java\jdk1.6.0_20): <Java Home>
    - Location of TUGBU Jar files (e.g. C:\FW42030\JarFiles): <..\FW42030\jarfiles>
    - Oracle User with read-write privileges to the Database Schema: <CISUSER>
    - Oracle User with read-only privileges to the Database Schema: <CISREAD>
    - Oracle database role with read-write privileges to the Database Schema: <CIS_USER>
    - Oracle database role with read-only privileges to the Database Schema: <CIS_READ>
    - Name of the owner of the Database Schema: <CISADM>
    - Enter the password for the CISADM schema (or hit ENTER to quit): <CISADM user's password>
    - Re-enter the password: <CISADM user's password>

Install the Oracle Utilities Meter Data Management Database Component

To install the Oracle Utilities Meter Data Management Database Component:

1. Run OraDBI.exe from the ..\MDM\Install-Upgrade directory. The utility prompts you to enter values for the following parameters:
    - Name of the target database: <DB NAME>

- Name of the Database User Name: <CISADM>
- Password of the User Name: <Password for CISADM>
- Location of Java Home (e.g. C:\Java\jdk1.6.0_20): <Java Home>
- Location of TUGBU Jar files (e.g. C:\FW42030\JarFiles): <..\FW42030\jarfiles>
- Oracle User with read-write privileges to the Database Schema: <CISUSER>
- Oracle User with read-only privileges to the Database Schema: <CISREAD>
- Oracle database role with read-write privileges to the Database Schema: <CIS_USER>
- Oracle database role with read-only privileges to the Database Schema: <CIS_READ>
- Name of the owner of the Database Schema: <CISADM>
- Enter the password for the CISADM schema (or hit ENTER to quit): <CISADM user's password>
- Re-enter the password: <CISADM user's password>

If you chose to continue, OraDBI first checks for the existence of each of the users specified and prompts for their password, default tablespace, and temporary tablespace.

After setting up roles and users, the utility continues upgrading schema and system data definitions. If an error occurs while executing an SQL or another utility, it logs and displays the error message and allows you to re-execute the current step.

2. **Optional:** This optional step should be executed if you have installed Oracle Utilities Meter Data Analytics 2.5.0.0.2 (2.5 Patch Set 2), or if you plan to install it in the future.
   - Go to ..MDM\Post-Upgrade folder and run Materialized_View_Creation.sql from sql prompt as follows.
   - Connect to Database Owner Schema e.g. <CISADM>/<CISADM>@<SID>
   - Run Materialized_View_Creation.sql as @Materialized_View_Creation.sql from sql prompt.

Go to ..MDM\Security folder and run oragensec.bat after changing the appropriate database role and users details.

OraDBI Performs the Following Tasks

- 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).
- Connects to the database as CISADM account, checks whether the user already has the application schema installed to verify whether this is an initial installation.
- 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 necessarily mean a problem exists.
- Stores the Schema owner and password in the feature configuration table. The password is stored in encrypted format.

Postinstallation Tasks

Populating Language Data

Please note that this database contains data in the ENGLISH language only. If you use any other supported language, you can run the F1-LANG batch program to duplicate the entries for new language records. For more information on running this batch program, refer to the user documentation section "Defining Background Processes."

Database Statistics Generation

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 using the DBMS_STATS package.

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.

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

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

3. Grant permission by running following 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.

Create Activity Statistics Materialized view

To improve the performance of drill down queries, use the following procedure to create the materialized view and then refresh the materialized view.

Go to ..SMDF\Post-Upgrade and run the scripts below.

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

```
@D1_ACTIVITY_STAT_MV.sql
@D1_MV_REFRESH_PROC.sql
```

Configuring Security

The configuration utility and scripts are located in the .\MDM\Security folder. To configure security, follow these steps to execute the OraGenSec.bat utility:

> **Note:** Database vault must be disabled before running this utility.

1. Navigate to the ..\MDM\Security folder.
2. Edit the OraGenSec.bat file and replace the parameter database-name with the name of your database. This file is provided for your convenience and executes the Oragensec.exe utility based on the parameters passed into it.

   > **Note:** Be sure to run OraGenSec.bat from a Windows 32-bit desktop that has the Oracle 11.2.0.1 client (or higher) installed. Your database should already be listed in the local file tnsnames.ora.

   The script will execute as the following:

   ```
   oragensec -d CISADM,CISADM,database-name -r CIS_READ,CIS_USER -a A
   -u CISUSER,CISREAD
   ```

3. Execute the edited OraGenSec.bat file. The utility configures security for the CISADM schema objects

Upgrade Install

This section describes how to upgrade the database components of Oracle Utilities Meter Data Management, including:

- Supported Upgrade Path
- Copying and Decompressing Install Media
- Installing the CISADM Schema
- Postinstallation Tasks

Supported Upgrade Path

Direct upgrade to Oracle Utilities Meter Data Management V2.1.0.3 is supported from the following versions:

- Oracle Utilities Meter Data Management V2.0.1.9
- Oracle Utilities Meter Data Management V2.1.0.1
- Oracle Utilities Meter Data Management V2.1.0.2

The following section assumes:

- an existing Oracle Utilities Meter Data Management V2.0.1.9 installation on top of an Oracle Utilities Application Framework V4.1.0.2.0 installation

  or

- an existing Oracle Utilities Meter Data Management V2.1.0.1 installation on top of an Oracle Utilities Application Framework V4.2.0.1.0 installation

  or

- an existing Oracle Utilities Meter Data Management V2.1.0.2 installation on top of an Oracle Utilities Application Framework V4.2.0.2.0 installation

Copying and Decompressing Install Media

To copy and decompress the Oracle Utilities Meter Data Management database:

1. Download the Oracle Utilities Meter Data Management V2.1.0.3.0 Oracle database from the Oracle Software Delivery Cloud.
2. Create a temporary directory, such as C:\OUMDM\temp or /OUMDM/temp. (Referred to below as <TEMPDIR>) This directory must be located outside any current working Oracle Utilities application environment. All files that are placed in this directory as a part of the installation can be deleted after completing a successful installation.
3. Copy the file MDM-V2.1.0.3.0-Database.zip from the delivered package to the <TEMPDIR>.
4. Unzip the MDM-V2.1.0.3.0-Database.zip file to a temporary folder. This file contains the database components required to install the Oracle Utilities Meter Data Management database.

Installing the CISADM Schema

Please back up your database before carrying out the upgrade process.

You must install Oracle Utilities Application Framework V4.2.0.3 and prerequisite single fixes prior to Oracle Utilities Meter Data Management. The files for Oracle Utilities Application Framework installation are located in the FW42030 folder. The installation process prompts you for the following information:

- A database user that will own the application schema (for example, CISADM).
- A database user 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. (for example, CISUSER).
- A database user with read-only privileges to the objects in the application schema. (for 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. (for example, CIS_USER).
- A database role with read-only privileges to the objects in the application schema (for example, CISREAD)
- Location of the jar files (The jar files are bundled with the database package)
- Java Home (For example, C:\Java\jdk1.6.0_20)

Install Oracle Utilities Application Framework V4.2.0.3.0

1. Run OraDBI.exe from the ..\FW42030\Install-Upgrade directory. Please run the utility from the command prompt.

   Note: Be sure to run OraDBI.exe from a Window 32-bit or 64-bit desktop that has the Oracle Database 11g Release 2 Client (11.2.0.1), 32-bit and Java Development Kit Version 6.0 Update 20 or later installed. The database should already be listed in the local file tnsnames.ora

2. The utility prompts you to enter values for the following parameters:
   - Name of the target database: <DB NAME>
   - Name of the Database User Name: <CISADM>
   - Password of the User Name: <Password for CISADM>
   - Location of Java Home (e.g. C:\Java\jdk1.6.0_20): <Java Home>
   - Location of TUGBU Jar files (e.g. C:\FW42030\JarFiles): <..\FW42030\jarfiles>
   - Oracle User with read-write privileges to the Database Schema: <CISUSER>
   - Oracle User with read-only privileges to the Database Schema: <CISREAD>
   - Oracle database role with read-write privileges to the Database Schema: <CIS_USER>
   - Oracle database role with read-only privileges to the Database Schema: <CIS_READ>
   - Name of the owner of the Database Schema: <CISADM>
   - Enter the password for the CISADM schema (or hit ENTER to quit): <CISADM user's password>
   - Re-enter the password: <CISADM user's password>

   After setting up roles and users, the utility continues upgrading schema and system data definitions. If an error occurs while executing an SQL or another utility, it logs and displays the error message and allows you to re-execute the current step.

Install Prerequisite Database Hot Fixes

Before installing Oracle Utilities Meter Data Management, you must install Oracle Utilities Framework Prerequisite database hot fixes as described below. Apply prerequisite Framework DB single fixes. See the Oracle Utilities Meter Data Management *Installation Guide* for more details.

Apply the prerequisite Framework database fixes by running the CDXPatch.exe

The utility prompts you for the value of the following parameters:

- The target database type (O/M/D) [O]: O
- The name of the user that owns the database objects: <CISADM>
- The password for the user (in silent mode): <Password for CISADM user>
- The name of the Oracle database: <DB Name>

CDXPatch.exe can be executed by selecting it from Windows explorer, or by using a command line from a DOS window. Use the option "-h" to see the help.

After the patches are processed, if the utility prompts you to create security for new objects. enter "N" because security for new objects is generated in subsequent steps during installation of Oracle Utilities Meter Data Management.

Install Oracle Utilities Service and Measurement Data Foundation Database Component

Before installing Oracle Utilities Meter Data Management, you must install Oracle Utilities Service and Measurement Data Foundation.

The following procedure describes how to install the database component of Oracle Utilities Service and Measurement Data Foundation database.

1. Run OraDBI.exe from the ..\SMDF\Install-Upgrade directory. The utility prompts you to enter values for the following parameters:
    - Name of the target database: <DB NAME>
    - Name of the Database User Name: <CISADM>
    - Password of the User Name: <Password for CISADM>
    - Location of Java Home (e.g. C:\Java\jdk1.6.0_20): <Java Home>
    - Location of TUGBU Jar files (e.g. C:\FW42030\JarFiles): <..\FW42030\jarfiles>
    - Oracle User with read-write privileges to the Database Schema: <CISUSER>
    - Oracle User with read-only privileges to the Database Schema: <CISREAD>
    - Oracle database role with read-write privileges to the Database Schema: <CIS_USER>
    - Oracle database role with read-only privileges to the Database Schema: <CIS_READ>
    - Name of the owner of the Database Schema: <CISADM>
    - Enter the password for the CISADM schema (or hit ENTER to quit): <CISADM user's password>
    - Re-enter the password: <CISADM user's password>

Install the Oracle Utilities Meter Data Management Database Component

To install the Oracle Utilities Meter Data Management Database Component:

1. Run OraDBI.exe from the ..\MDM\Install-Upgrade directory. The utility prompts you to enter values for the following parameters:
    - Name of the target database: <DB NAME>
    - Name of the Database User Name: <CISADM>
    - Password of the User Name: <Password for CISADM>
    - Location of Java Home (e.g. C:\Java\jdk1.6.0_20): <Java Home>
    - Location of TUGBU Jar files (e.g. C:\FW42030\JarFiles): <..\FW42030\jarfiles>
    - Oracle User with read-write privileges to the Database Schema: <CISUSER>
    - Oracle User with read-only privileges to the Database Schema: <CISREAD>

- Oracle database role with read-write privileges to the Database Schema: <CIS_USER>
- Oracle database role with read-only privileges to the Database Schema: <CIS_READ>
- Name of the owner of the Database Schema: <CISADM>
- Enter the password for the CISADM schema (or hit ENTER to quit): <CISADM user's password>
- Re-enter the password: <CISADM user's password>

If you chose to continue, OraDBI first checks for the existence of each of the users specified and prompts for their password, default tablespace, and temporary tablespace.

After setting up roles and users, the utility continues upgrading schema and system data definitions. If an error occurs while executing an SQL or another utility, it logs and displays the error message and allows you to re-execute the current step.

2. **Optional:** This optional step should be executed if you have installed Oracle Utilities Meter Data Analytics 2.5.0.0.2 (2.5 Patch Set 2), or plans to install it in the future.
   - Go to ..MDM\Post-Upgrade folder and run Materialized_View_Creation.sql from sql prompt as follows.
   - Connect to Database Owner Schema e.g. <CISADM>/<CISADM>@<SID>
   - Run Materialized_View_Creation.sql as @Materialized_View_Creation.sql from sql prompt.

Go to ..MDM\Security folder and run oragensec.bat after changing the appropriate database role and users details.

OraDBI Performs the Following Tasks

- 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).
- Connects to the database as CISADM account, checks whether the user already has the application schema installed to verify whether this is an initial installation.
- 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 necessarily mean a problem exists.
- Stores the Schema owner and password in the feature configuration table. The password is stored in encrypted format.

Postinstallation Tasks

Populating Language Data

Please note that this database contains data in the ENGLISH language only. If you use any other supported language, you can run the F1-LANG batch program to duplicate the entries for new language records. For more information on running this batch program, refer to the user documentation section "Defining Background Processes."

Database Statistics Generation

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 using the DBMS_STATS package.

Environment Registration

Note: If the target database is registered as a configuration laboratory or archiving database in another database, or another database has been registered as a configuration laboratory or archiving database in this database, it is required that you upgrade the registration at this stage.

The detailed instructions for environment registration can be found in the Oracle Utilities Meter Data Management user documentation. Please refer to this documentation before executing the environment registration utility EnvSetup.exe included in the post-install folder.

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.

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

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

3. Grant permission by running following 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.

Create Activity Statistics Materialized view

To improve the performance of drill down queries, use the following procedure to create the materialized view and then refresh the materialized view.

Go to ..SMDF\Post-Upgrade and run the scripts below.

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

```
@D1_ACTIVITY_STAT_MV.sql
@D1_MV_REFRESH_PROC.sql
```

Configuring Security

The configuration utility and scripts are located in the .\MDM\Security folder. To configure security, follow these steps to execute the OraGenSec.bat utility:

> **Note:** Database vault must be disabled before running this utility.

1. Navigate to the ..\MDM\Security folder.
2. Edit the OraGenSec.bat file and replace the parameter database-name with the name of your database. This file is provided for your convenience and executes the Oragensec.exe utility based on the parameters passed into it.

   > **Note:** Be sure to run OraGenSec.bat from a Windows 32-bit desktop that has the Oracle 11.2.0.1 client (or higher) installed. Your database should already be listed in the local file tnsnames.ora.

   The script will execute as the following:

   ```
   oragensec -d CISADM,CISADM,database-name -r CIS_READ,CIS_USER -a A
   -u CISUSER,CISREAD
   ```

3. Execute the edited OraGenSec.bat file. The utility configures security for the CISADM schema objects

Consideration for Upgrade from Framework Versions Prior to Version 4.2.0 Service Pack 2

Customers upgrading from Framework versions prior to 4.2.0.3 need to run an upgrade script to trim the SRCH_CHAR_VAL column on the char tables. The search char value column is so far being populated with trailing spaces for Java-based objects for char types which are pre-defined values as well as foreign key values with user defined keys. This will result in empty results during "exact string" searches on SRCH_CHAR_VAL column. As a part of patch 16745968, this is fixed in the application but the existing data needs to be cleaned up.

The upgrade script "FW4202_Trim_SRCH_CHAR_VAL.sql" that is available in FW420 SP3 database blueprint upgrade folder can be used to perform this cleanup. This SQL should be executed by a schema owner and it will create a SQL file named "TRIM_SRCH_CHAR_VAL.sql". This SQL file ("TRIM_SRCH_CHAR_VAL.sql") is going to trim SRCH_CHAR_VAL columns of all the characteristics tables and will do this update operation in multiple chunks of key ranges. Only char types with pre-defined values as well as foreign key values with user defined keys will be updated in those char tables. Once sql-"FW4202_Trim_SRCH_CHAR_VAL.sql" is executed it cannot be re-executed.

> **Note:** This is only needed during upgrades from FW versions below 420 SP2. It should NOT be run in an initial install environment.

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 ..\FW42030\Install-Upgrade directory

```
@FW4202_Trim_SRCH_CHAR_VAL.sql
```

3. Run the generated TRIM_SRCH_CHAR_VAL.sql script

   ```
   @TRIM_SRCH_CHAR_VAL.sql
   ```

Demo Install

This section describes how to install the demo database components for Oracle Utilities Meter Data Management, including:

- Copying and Decompressing Install Media
- Creating the Database and Importing Dump File
- Postinstallation Tasks

Copying and Decompressing Install Media

To copy and decompress the Oracle Utilities Meter Data Management database:

1. Download the Oracle Utilities Meter Data Management V2.1.0.3.0 Oracle database from the Oracle Software Delivery Cloud.
2. Unzip the MDM-V2.1.0.3.0-Database.zip file to a temporary folder. This file contains the database components required to install the Oracle Utilities Meter Data Management database.

Creating the Database and Importing Dump File

You can use the database creation tool (cdxdba.plx for UNIX or CDXDBA.exe for Windows) to create the demo database with AL32UTF8 character set.

The UNIX and Windows database creation utilities create an empty database with AL32UTF8 character set and at least one tablespace for storing the application objects before running the installation. The default name of the application tablespace is CISTS_01.

- Creating the Demo Database on Unix
- Creating the Demo Database on Windows

Creating the Demo Database on Unix

The files for creating the database are located in ../MDM/DatabaseCreation/Unix directory. Follow these steps to create a database:

> **NOTE:** For Oracle 12c (12.1.0.1+), use the files under ../MDM/DatabaseCreation/Unix_12c.

1. FTP the contents of the Database Creation folder to a temporary directory on the UNIX server.
2. Set the ORACLE_HOME and ORACLE_BASE variables.
3. Run the utility cdxdba.plx by executing the following command:

   ```
   perl cdxdba.plx
   ```
4. When prompted, provide the following parameter values:
   - Instance name (DEMO):
   - ORACLE_BASE: the directory where the setup files for the database will be created (/orasw/app/oracle):
   - ORACLE_HOME: the folder where the current version of Oracle software is installed (/orasw/app/oracle/product/):
   - ORACLE_DATA: the directory where the data files for the database will be created (/db05/oradata):
   - Character set for the database (AL32UTF8):

Enter the parameter values based on the settings of your database server. You can also accept the default values displayed if they match your database server settings. You will be prompted to confirm the settings and then to select Y to create the database.

- ORACLE_SID: DEMO
- ORACLE_HOME: /orasw/app/oracle/product/
- ORACLE_BASE: /orasw/app/oracle
- ORACLE_DATA: /db05/oradata
- Character Set: AL32UTF8
- Do you want to continue (Y/N)?

5. When the database has been created, you will be prompted with the following questions:
   - Do you want to import a demo database dump into this database (Y/N)?
   - Select Y to import the Demo Install data.

   For the demo installation use the dump file exp_demo.dmp.

   Note: The data_pump_dir must exist in the database created above before continuing with the import. You should also copy the exp_demo.dmp file to the data_pump_dir. Decompress the exp_demo.dmp.gz file first to extract the exp_demo.dmp file. This file is in ..\Demo directory.

   - Do you want to import a demo database dump into this database (Y/N)? Y
   - Enter the name of the dump file (exp_demo.dmp):
   - Enter the name of the dump file directory (data_pump_dir):
   - Enter the name of the log file (exp_demo.log):
6. Update the oratab file for the new database and then check the connectivity to this database from another server and from your desktop after updating local tnsnames.ora file.

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

   a. Set the correct ORACLE_SID and ORACLE_HOME.

   b. Run following command to import demo dump:

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

Creating the Demo Database on Windows

The files for creating the database are located in the ..\MDM\DatabaseCreation\Windows directory.

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.

Follow these steps to create the database:

1. From a command prompt, run the utility CDXDBA.exe, located in the Windows folder.

   The utility displays the following options:

   E - Export a schema from the database

   R - Refresh a schema with a database dump

   C - Create/Recreate a local database

   H - See help for the command line options

   Q - Quit

2. Select option C to create an empty database on your machine.

   Provide the following values:

   - Provide the instance name (DEMO): <DB Name> For example, MDM_DB
   - Enter the character set of the database (AL32UTF8): AL32UTF8
   - Enter ORACLE_BASE: the directory where the setup files for the database will be created (c:\oracle): <Oracle_Base> For example, c:\app\oracle
   - Enter ORACLE_HOME: the folder where the current version of Oracle software is installed (c:\oracle\product\11.1.0.6\Db_1):< Oracle_Home> For example, c:\app\oracle\db_home
   - Enter ORACLE_DATA: the directory where the data files for the database will be created (c:\app\oracle\oradata): <Directory where data files will be created>

3. Once the database has been created, select the R - Refresh a schema option with a database dump file to load the Demo Install data.

   - Select an option: R
   - Enter the instance name (DEMO): <DB name>
   - Is it a LOCAL database (exists on the same machine) (Y/N): <Please provide Y or N>
   - Enter the name of the Oracle account that owns that application schema (cisadm):CISADM
   - Enter password for CISADM (cisadm): CISADM
   - Enter the character set of the database (AL32UTF8): AL32UTF8
   - Enter the name of data pump directory (DATA_PUMP_DIR): DATA_PUMP_DIR
   - Enter the name of the dump file (exp_demo.dmp):exp_demo.dmp
   - Enter the name of the log file (imp_demo.log):exp_demo.log

For the DB user system, the password is manager. Option R causes the utility to drop all the objects from the schema and import the schema from a database dump file.

For the Demo Installation, use the dump file exp_demo.dmp.

> **Note:** The data_pump_dir must exist in the database created above before continuing with the import. You should also copy the exp_demo.dmp file to the data_pump_dir. Decompress the exp_demo.dmp.gz file to extract the exp_demo.dmp file. This file is in the ..\Demo directory. Check the connectivity to this database from another server and from your desktop after updating local tnsnames.ora file

Postinstallation Tasks

Configuring Security

The configuration utility and scripts are located in the .\MDM\Security folder. To configure security, follow these steps to execute the OraGenSec.bat utility:

> **Note:** Database vault must be disabled before running.

1. Navigate to the ..\MDM\Security folder.

2. Edit the OraGenSec.bat file and replace the parameter database-name with the name of your database. This file is provided for your convenience and executes the Oragensec.exe utility based on the parameters passed into it.

   Note: Be sure to run OraGenSec.bat from a Windows 32-bit desktop that has the Oracle 11.2.0.1 client (or higher) installed. Your database should already be listed in the local file tnsnames.ora.

The script will execute as the following:

```
oragensec -d CISADM,CISADM,database-name -r CIS_READ,CIS_USER -a A
-u CISUSER,CISREAD
```

3. Execute the edited OraGenSec.bat file. The utility configures security for the CISADM schema objects

Populating Language Data

Please note that this database contains data in the ENGLISH language only. If you use any other supported language, you can run the F1-LANG batch program to duplicate the entries for new language records. For more information on running this batch program, refer to the user documentation section "Defining Background Processes." You can also install the language specific demo data packages (if available) into the database. Please contact your Oracle representative to receive information on these packages.

Chapter 3

Database Design

This section provides a standard for database objects such as tables, columns, and indexes, for products using the Oracle Utilities Application Framework. This standard helps 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 section includes:

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. See "Indexes" on page 3 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 (LANGUAGE_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. 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. Prior to Version 4.1.0 of the framework the leading character of the base Owner Flag was used. From 4.1.0 on the first two characters of product's owner flag value should be used. For client specific implementation of index, use CM for Owner Flag.
- 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
- 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.

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, Framework its F1. Other examples are D1,D2, etc.
   - 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_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

- OF stands for Owner Flag. For example, the framework is F1. Other examples are D1,D2, etc.
- 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

   - OF stands for Owner Flag. For example, the framework is F1. Other examples are D1,D2, etc.

   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

Oracle Utilities Application Framework 4.1.0 Group Fix 2 and later versions support 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 added to FW410 Group Fix 2 fixes the metadata to make sure that the existing tables will not be affected.

This new feature only supports tables maintained by Java. 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 COBOL.

XML Type Support

Oracle Utilities Application Framework v4.2.0.0 onwards 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 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

The following section outlines the general implementation guidelines for the database components, including:

- Configuration Guidelines

  Note: Refer to My Oracle Support for more information.

- Oracle Database Implementation Guidelines

Configuration Guidelines

This section includes 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. This section includes:

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.

In an Oracle Utilities Application Framework environment, always make sure that the optimization parameters are set as follows:

```
optimizer_index_cost_adj=1
```

This will make sure that the optimizer gives a higher priority to index scans.

Note: If the implementation creates a CM index on table-columns for which 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 the Final Measurement table (D1_MSRMT), keep the current table partition uncompressed. All of the older partitions will be compressed based on QUERY HIGH compression.
- For the Initial Measurement Data table (D1_INIT_MSMRT_DATA), always keep CLOBs in securefile and medium compressed. 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, including ILM enabled MOs shall be compressed.
- 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(s)) indexes will share this tablespace.
  - Do NOT specify standard index compression
- LOBs and CLOBs
  - Securefile medium compression in row

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, please refer to the chapter **Information Lifecycle Management and Data Archiving** in this guide 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 ensure that the "COMPRESS" flag is turned on by setting it to "Y" in Storage.xml.

See "Database Syntax" on page 6 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

optimizer_index_cost_adj=1

optimizer_index_caching=100

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 batch job BATCH C1- TRMDD

```
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
/
```

Chapter 5

Information Lifecycle Management and Data Archiving

Oracle Utilities Meter Data Management 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.
- On-going Maintenance Phase - This phase covers the ongoing maintenance tasks.

Preparation Phase

> Note: In order to successfully implement ILM as described here, the following DB Version and Patch are pre-requisites: database version 12.1.0.2.0 Enterprise Edition 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 Module Specific ILM Implementation Details section outlines the additional steps to be performed on base delivered ILM Enabled Tables to conform to ILM requirements. In addition, Sample SQL for Enabling ILM (Initial Installation)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 Sample SQL for Enabling ILM (Existing Installation) for detailed information using To Do Entry as an example. Also refer to Sample SQL for Enabling ILM with Sub Retention (Existing Installation) for detailed information using D1_INIT_MSRMT_DATA 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'.

   Note: Certain ILM enabled MOs, specifically IMD, Device Event, and Activity, support more than one retention period also known as sub retention periods. For these MOs the table will be sub-partitioned based on the retention period. Furthermore, a more detailed approach will be required to set both the ILM date (ILM_DT) and the retention period (<field name>). If your implementation does not wish to leverage the ability to define multiple retention periods for these MOs, this note can be ignored and the general guidelines for ILM enablement can be followed. If your implementation wishes to leverage the multiple retention period capability then please refer to the Module Specific ILM Implementation Details For Sub Retention section below.
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 Sample SQL for Enabling ILM (Existing Installation) . If sub retention is supported, sample DDL for child tables can be found in Sample SQL for Enabling ILM with Sub Retention (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.
11. If sub-retention is not supported, create the ILM specific indexes from the Module Specific ILM Implementation Details section.

Table Name	Index Name
CI_TD_ENTRY	CM_ILM_XT039S8
D1_ACTIVITY	CM_ILM_D1T319S1
D1_COMM_IN	CM_ILM_D1T386S1
D1_COMM_OUT	CM_ILM_D1T380S1

Table Name	Index Name
D1_COMPL_EVT	CM_ILM_D1T340S1
D1_DVC_EVT	CM_ILM_D1T400S4
D1_INIT_MSRMT_DATA	CM_ILM_D1T304S4
D1_USAGE	CM_ILM_D1T281S2
D1_VEE_EXCP	CM_ILM_D1T308S2
F1_SYNC_REQ	CM_ILM_F1T014S4
F1_SYNC_REQ_IN	CM_ILM_F1T191S3
F1_OUTMSG	CM_ILM_FT010S2
F1_SVC_TASK	CM_ILM_F1C474S2
F1_OBJ_REV	CM_ILM_FT035S6
D1_SNAPSHOT_DL_CTRL	CM_ILM_D1T433S1
D1_SP_SNAP_DL	CM_ILM_D1T434S1
D1_SP_UNR_USG_SNAP_DL	CM_ILM_D1T436S1
D1_SP_VEE_EXCP_SNAP_DL	CM_ILM_D1T440S1

12. If sub-retention is supported, please follow the procedure below:

    a. Install patch 22446142 for Information Lifecycle Management sub-retention which adds the RETENTION_PERIOD column to the D1_INIT_MSRMT_DATA, D1_DVC_EVT, and D1_ACTIVITY tables.

    b. Create the following ILM specific indexes from the Module Specific ILM Implementation Details section:

Table Name	Index Name
CI_TD_ENTRY	CM_ILM_XT039S8
D1_COMM_IN	CM_ILM_D1T386S1
D1_COMM_OUT	CM_ILM_D1T380S1
D1_COMPL_EVT	CM_ILM_D1T340S1
D1_USAGE	CM_ILM_D1T281S2
D1_VEE_EXCP	CM_ILM_D1T308S2
F1_SYNC_REQ	CM_ILM_F1T014S4
F1_SYNC_REQ_IN	CM_ILM_F1T191S3
F1_OUTMSG	CM_ILM_FT010S2
F1_SVC_TASK	CM_ILM_F1C474S2
F1_OBJ_REV	CM_ILM_FT035S6
D1_SNAPSHOT_DL_CTRL	CM_ILM_D1T433S1
D1_SP_SNAP_DL	CM_ILM_D1T434S1
D1_SP_UNR_USG_SNAP_DL	CM_ILM_D1T436S1

Table Name	Index Name
D1_SP_VEE_EXCP_SNAP_DL	CM_ILM_D1T440S1

and the ILM subretention specific indexes from the Module Specific ILM Implementation Details For Sub Retention section:

Table Name	Index Name
D1_ACTIVITY	CM_ILM_D1T319S1
D1_DVC_EVT	CM_ILM_D1T400S4
D1_INIT_MSRMT_DATA	CM_ILM_D1T304S4

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.

This section details the following maintenance objects:

- To Do Entry
- Sync Request (Outbound)
- Inbound Sync Request
- Outbound Message
- Service Task
- Object Revision
- Activity
- Communication In
- Communication Out
- Device Event
- Completion Event
- Initial Measurement Data
- Usage Transaction
- VEE Exception
- Snapshot Tables

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)					CI_TD_ENTRY. CRE_DTTM
		XT039P0	TD_ENTRY_ID	Global Partitioned	RANGE (TD_ENTRY_ID)	
		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	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		
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		

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_MSG_ PARM(Child table of CI_TD_LOG)	Reference Partitioning	XT040P0	TD_ENTRY_ID, SEQ_NUM	Global		
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		
		F1T014S3	MAINT_OBJ_CD, PK_VALUE1, PK_VALUE2, F1_SYNC_REQ_ID	Global		
		CM_ILM_F1T014 S4	ILM_DT, ILM_ARC_SW, F1_SYNC_REQ_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_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		

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
		F1T191S1	BO_STATUS_CD, BUS_OBJ_CD, F1_SYNC_REQ_ IN_ID	Global		
		F1T191S2	MAINT_OBJ_CD, EXT_PK_VALUE1, NT_XID_CD, PK_VALUE1	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_PAR M (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		
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		

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_F1C474S2	ILM_DT, ILM_ARC_SW, F1_SVC_TASK_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_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		
		F1C479S1	MAINT_OBJ_CD, PK_VALUE1, PK_VALUE2, PK_VALUE3	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

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
		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		
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.

Activity

Note: If sub retention periods will be defined for this MO, then please follow the guidelines set forth in Module Specific ILM Implementation Details For Sub Retention section.

This table describes the 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
D1_ACTIVITY (Parent)	RANGE (ILM_DT, D1_ACTIVITY_ID) **Note**: Default is to use sub-retention or use RANGE (ILM_DT, D1_ACTIVITY_ID) if not using sub-retention.					D1_ACTIVITY. CRE_DTTM
		D1T319P0	D1_ACTIVITY_ID	Global Partitioned	RANGE (D1_ACTIVITY_ID)	
		D1T319S0	BUS_OBJ_CD, BO_STATUS_CD, D1_ACTIVITY_ID	Global Partitioned	HASH(BUS_OBJ_CD, BO_STATUS_CD, D1_ACTIVITY_ID)	
		CM_ILM_ D1T319S1	ILM_DT, ILM_ARCH_SW, D1_ACTIVITY_ID	Local		
D1_ACTIVITY_ CHAR	REFERENCE (D1_ACTIVITY_ CHAR_FK)					
		D1T320P0	D1_ACTIVITY_ID, CHAR_TYPE_CD, SEQ_NUM	Global Partitioned	RANGE(D1_ ACTIVITY_ID)	
		D1T320S0	SRCH_CHAR_VAL	Global Partitioned	HASH(SRCH_CHAR _VAL)	
D1_ACTIVITY_ IDENTIFIER	REFERENCE (D1_ACTIVITY_ID ENTIFIER_FK)					
		D1T330P0	D1_ACTIVITY_ID, ACTIVITY_ID_TYPE_ FLG	Global Partitioned	RANGE(D1_ ACTIVITY_ID)	
		D1T330S0	ID_VALUE, ACTIVITY_ID_TYPE_ FLG	Global Partitioned	HASH(ID_VALUE, ACTIVITY_ID_ TYPE_FLG)	
D1_ACTIVITY_ LOG	REFERENCE (D1_ACTIVITY_ LOG_FK)					
		D1T321P0	D1_ACTIVITY_ID, SEQNO	Global Partitioned	RANGE(D1_ ACTIVITY_ID)	
		D1T321S1	CHAR_TYPE_CD, CHAR_VAL_FK1	Global Partitioned	HASH(CHAR_TYPE _CD, CHAR_VAL_FK1)	
		D1T321S2	CHAR_TYPE_CD, CHAR_VAL	Global Partitioned	HASH(CHAR_TYPE _CD, CHAR_VAL)	

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
D1_ACTIVITY_LOG_PARM	REFERENCE (D1_ACTIVITY_LOG_PARM_FK)					
		D1T322P0	D1_ACTIVITY_ID, SEQNO PARM_SEQ	Global Partitioned	RANGE(D1_ACTIVITY_ID)	
D1_ACTIVITY_REL	REFERENCE (D1_ACTIVITY_REL_FK)					
		D1T323P0	D1_ACTIVITY_ID, ACTIVITY_REL_TYPE_FLG	Global Partitioned	RANGE(D1_ACTIVITY_ID)	
		D1T323S0	REL_ACTIVITY_ID	Global Partitioned	HASH(REL_ACTIVITY_ID)	
D1_ACTIVITY_REL_OBJ	REFERENCE (D1_ACTIVITY_REL_OBJ_FK)					
		D1T324P0	D1_ACTIVITY_ID, MAINT_OBJ_CD, ACTIVITY_REL_OBJ_TYPE_FLG	Global Partitioned	RANGE(D1_ACTIVITY_ID)	
		D1T324S0	PK_VALUE1, PK_VALUE2, PK_VALUE3, PK_VALUE4, PK_VALUE5, MAINT_OBJ_CD	Global Partitioned	HASH(PK_VALUE1, PK_VALUE2, PK_VALUE3, PK_VALUE4)	

Communication In

This table describes the Communication In 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
D1_COMM_IN (Parent)	RANGE(ILM_DT, D1_COMM_ID)					D1_COMM_IN.CRE_DTTM
		D1T386P0	D1_COMM_ID	Global Partitioned	RANGE (D1_COMM_ID)	
		CM_ILM_D1T386S1	ILM_DT, ILM_ARCH_SW, D1_COMM_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
D1_COMM_IN_CHAR	REFERENCE (D1_COMM_IN_CHAR_FK)					
		D1T387P0	D1_COMM_ID, CHAR_TYPE_CD, SEQ_NUM	Global Partitioned	RANGE (D1_COMM_ID)	
		D1T387S0	SRCH_CHAR_VAL	Global Partitioned	HASH(SRCH_CHAR_VAL)	
D1_COMM_IN_IDENTIFIER	REFERENCE (D1_COMM_IN_IDENTIFIER_FK)					
		D1T391P0	D1_COMM_ID, COMM_ID_TYPE_FLG	Global Partitioned	RANGE(D1_COMM_ID)	
		D1T391S0	ID_VALUE, COMM_ID_TYPE_FLG	Global Partitioned	HASH(ID_VALUE, COMM_ID_TYPE_FLG)	
D1_COMM_IN_LOG	REFERENCE (D1_COMM_IN_LOG_FK)					
		D1T388P0	D1_COMM_ID, SEQNO	Global Partitioned	RANGE(D1_COMM_ID)	
		D1T388S1	CHAR_TYPE_CD, CHAR_VAL_FK1	Global Partitioned	HASH(CHAR_TYPE_CD, CHAR_VAL_FK1)	
		D1T388S2	CHAR_TYPE_CD, CHAR_VAL	Global Partitioned	HASH(CHAR_TYPE_CD, CHAR_VAL)	
D1_COMM_IN_LOG_PARM	REFERENCE (D1_COMM_IN_LOG_PARM_FK)					
		D1T389P0	D1_COMM_ID, SEQNO PARM_SEQ	Global Partitioned	RANGE(D1_COMM_ID)	
D1_COMM_IN_REL_OBJ	REFERENCE (D1_COMM_IN_REL_OBJ_FK)					
		D1T390P0	D1_COMM_ID, MAINT_OBJ_CD, COMM_REL_OBJ_TYPE_FLG	Global Partitioned	RANGE(D1_COMM_ID)	
		D1T390S0	PK_VALUE1, PK_VALUE2, PK_VALUE3, PK_VALUE4, PK_VALUE5, MAINT_OBJ_CD	Global Partitioned	HASH(PK_VALUE1, PK_VALUE2, PK_VALUE3, PK_VALUE4)	

Communication Out

This table describes the Communication Out 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
D1_COMM_OUT (Parent)	RANGE(ILM_DT, D1_COMM_ID)					D1_COMM_ OUT. CRE_DTTM
		D1T380P0	D1_COMM_ID	Global Partitioned	RANGE (D1_COMM_ID)	
		CM_ILM_ D1T380S1	ILM_DT, ILM_ARCH_SW, D1_COMM_ID	Local		
D1_COMM_OUT _CHAR	REFERENCE (D1_COMM_OUT _CHAR_FK)					
		D1T381P0	D1_COMM_ID, CHAR_TYPE_CD, SEQ_NUM	Global Partitioned	RANGE (D1_COMM_ID)	
		D1T381S0	SRCH_CHAR_VAL	Global Partitioned	HASH(SRCH_CHAR _VAL)	
D1_COMM_OUT _IDENTIFIER	REFERENCE (D1_COMM_OUT _IDENTIFIER_ FK)					
		D1T385P0	D1_COMM_ID, COMM_ID_TYPE_FLG	Global Partitioned	RANGE(D1_COMM _ID)	
		D1T385S0	ID_VALUE, COMM_ID_TYPE_FLG	Global Partitioned	HASH(ID_VALUE, COMM_ID_TYPE_ FLG)	
D1_COMM_ OUT_LOG	REFERENCE (D1_COMM_OUT _LOG_FK)					
		D1T382P0	D1_COMM_ID, SEQNO	Global Partitioned	RANGE(D1_COMM _ID)	
		D1T382S1	CHAR_TYPE_CD, CHAR_VAL_FK1	Global Partitioned	HASH(CHAR_ TYPE_CD, CHAR_VAL_FK1)	
		D1T382S2	CHAR_TYPE_CD, CHAR_VAL	Global Partitioned	HASH(CHAR_ TYPE_CD, CHAR_VAL)	
D1_COMM_OUT _LOG_PARM	REFERENCE (D1_COMM_OUT _LOG_PARM_FK)					
		D1T383P0	D1_COMM_ID, SEQNO PARM_SEQ	Global Partitioned	RANGE(D1_COMM _ID)	

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
D1_COMM_OUT_REL_OBJ	REFERENCE (D1_COMM_OUT_REL_OBJ_FK)					
		D1T384P0	D1_COMM_ID, MAINT_OBJ_CD, COMM_REL_OBJ_TYPE_FLG	Global Partitioned	RANGE(D1_COMM_ID)	
		D1T384S0	PK_VALUE1, PK_VALUE2, PK_VALUE3, PK_VALUE4, PK_VALUE5, MAINT_OBJ_CD	Global Partitioned	HASH(PK_VALUE1, PK_VALUE2, PK_VALUE3, PK_VALUE4)	

Device Event

Note: If sub retention periods will be defined for this MO, then please follow the guidelines set forth in the Module Specific ILM Implementation Details For Sub Retention section.

This table describes the Device 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
D1_DVC_EVT (Parent)	RANGE(ILM_DT, DVC_EVT_ID) **Note:** Default is to use sub-retention or use RANGE (ILM_DT,DVC_EVT_ID) if not using sub-retention.					D1_DVC_EVT.CRE_DTTM
		D1T400P0	DVC_EVT_ID	Global Partitioned	RANGE (DVC_EVT_ID)	
		D1T400S1	BUS_OBJ_CD, BO_STATUS_CD, DVC_EVT_ID	Global Partitioned	HASH(BUS_OBJ_CD,BO_STATUS_CD, DVC_EVT_ID)	
		D1T400S2	D1_DEVICE_ID, DVC_EVT_DTTM	Global Partitioned	HASH(D1_DEVICE_ID, DVC_EVT_DTTM)	
		D1T400S3	BUS_OBJ_CD, BO_STATUS_CD, D1_DEVICE_ID, DVC_EVT_ID	Global Partitioned	HASH(BUS_OBJ_CD,BO_STATUS_CD, D1_DEVICE_ID, DVC_EVT_ID)	

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_D1T400S4	ILM_DT, ILM_ARCH_SW, DVC_EVT_ID	Local		
D1_DVC_EVT_CHAR	REFERENCE (D1_DVC_EVT_CHAR_FK)					
		D1T401P0	DVC_EVT_ID, CHAR_TYPE_CD, SEQ_NUM	Global Partitioned	RANGE(DVC_EVT_ID)	
		D1T401S0	SRCH_CHAR_VAL	Global Partitioned	HASH(SRCH_CHAR_VAL)	
D1_DVC_EVT_IDENTIFIER	REFERENCE (D1_DVC_EVT_IDENTIFIER_FK)					
		D1T405P0	DVC_EVT_ID, DVC_EVT_ID_TYPE_FLG	Global Partitioned	RANGE(DVC_EVT_ID)	
		D1T405S0	ID_VALUE, DVC_EVT_ID_TYPE_FLG	Global Partitioned	HASH(ID_VALUE, DVC_EVT_ID_TYPE_FLG)	
D1_DVC_EVT_LOG	REFERENCE (D1_DVC_EVT_LOG_FK)					
		D1T402P0	DVC_EVT_ID, SEQNO	Global Partitioned	RANGE(DVC_EVT_ID)	
		D1T402S1	CHAR_TYPE_CD, CHAR_VAL_FK1	Global Partitioned	HASH(CHAR_TYPE_CD, CHAR_VAL_FK1)	
		D1T402S2	CHAR_TYPE_CD, CHAR_VAL	Global Partitioned	HASH(CHAR_TYPE_CD, CHAR_VAL)	
D1_DVC_EVT_LOG_PARM	REFERENCE (D1_DVC_EVT_LOG_PARM_FK)					
		D1T403P0	DVC_EVT_ID, SEQNO PARM_SEQ	Global Partitioned	RANGE(DVC_EVT_ID)	
D1_DVC_EVT_REL_OBJ	REFERENCE (D1_DVC_EVT_REL_OBJ_FK)					
		D1T404P0	DVC_EVT_ID, MAINT_OBJ_CD, DVC_EVT_REL_OBJ_TYPE_FLG	Global Partitioned	RANGE(DVC_EVT_ID)	

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
		D1T404S0	PK_VALUE1, PK_VALUE2, PK_VALUE3, PK_VALUE4, PK_VALUE5, MAINT_OBJ_CD	Global Partitioned	HASH(PK_VALUE1 , PK_VALUE2, PK_VALUE3, PK_VALUE4)	

Completion Event

This table describes the Completion 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
D1_COMPL_EVT (Parent)	RANGE(ILM_DT, COMPL_EVT_ID)					D1_COMPL_ EVT. CRE_DTTM
		D1T340P0	COMPL_EVT_ID	Global Partitioned	RANGE (COMPL_EVT_ID)	
		D1T340S0	D1_ACTIVITY_ID	Global Partitioned	HASH(D1_ ACTIVITY_ID)	
		CM_ILM_ D1T340S1	ILM_DT, ILM_ARCH_SW, DVC_EVT_ID	Local		
D1_COMPL_EVT _CHAR	REFERENCE (D1_COMPL_ EVT_CHAR_FK)					
		D1T341P0	COMPL_EVT_ID, CHAR_TYPE_CD, SEQ_NUM	Global Partitioned	RANGE(COMPL_ EVT_ID)	
		D1T341S1	SRCH_CHAR_VAL	Global Partitioned	HASH(SRCH_CHAR _VAL)	
D1_COMPL_EVT _LOG	REFERENCE (D1_COMPL_EVT _LOG_FK)					
		D1T342P0	COMPL_EVT_ID, SEQNO	Global Partitioned	RANGE(COMPL_ EVT_ID)	
		D1T342S1	CHAR_TYPE_CD, CHAR_VAL_FK1	Global Partitioned	HASH(CHAR_ TYPE_CD, CHAR_VAL_FK1)	
		D1T342S2	CHAR_TYPE_CD, CHAR_VAL	Global Partitioned	HASH(CHAR_ TYPE_CD, CHAR_VAL)	

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
D1_COMPL_EVT_LOG_PARM	REFERENCE (D1_COMPL_EVT_LOG_PARM_FK)					
		D1T343P0	COMPL_EVT_ID, SEQNO PARM_SEQ	Global Partitioned	RANGE(COMPL_EVT_ID)	
D1_COMPL_EVT_REL_OBJ	REFERENCE (D1_COMPL_EVT_REL_OBJ_FK)					
		D1T344P0	COMPL_EVT_ID, MAINT_OBJ_CD, COMPL_EVT_REL_OBJ_TYP_FLG	Global Partitioned	RANGE(COMPL_EVT_ID)	
		D1T344S0	PK_VALUE1, PK_VALUE2, PK_VALUE3, PK_VALUE4, PK_VALUE5, MAINT_OBJ_CD	Global Partitioned	HASH(PK_VALUE1, PK_VALUE2, PK_VALUE3, PK_VALUE4)	

Initial Measurement Data

Note: If sub retention periods will be defined for this MO, then please follow the guidelines set forth in the Module Specific ILM Implementation Details For Sub Retention section.

This table describes the Initial Measurement Data 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
D1_INIT_MSRMT_DATA (Parent)	RANGE (ILM_DT,MEASR_COMP_ID) Note: Default is to use sub-retention or use RANGE (ILM_DT,MEASR_COMP_ID) if not using sub-retention.					D1_INIT_MSRMT_DATA. CRE_DTTM
		D1T304P0	INIT_MSRMT_DATA_ID	Global Partitioned	RANGE (INIT_MSRMT_DATA_ID)	

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
		D1T304S1	MEASR_COMP_ID, BO_STATUS_CD, BUS_OBJ_CD, D1_TO_DTTM, D1_FROM_DTTM	Global Partitioned	RANGE (MEASR_COMP_ID)	
		D1T304S2	BUS_OBJ_CD, BO_STATUS_CD, INIT_MSRMT_DATA_ID	Global Partitioned	HASH (BUS_OBJ_CD, BO_STATUS_CD, INIT_MSRMT_DAT A_ID)	
		D1T304S3	IMD_EXT_ID, INIT_MSRMT_DATA_ID	Global Partitioned	HASH(IMD_EXT_ ID)	
		CM_ILM_ D1T304S4	ILM_DT, ILM_ARCH_SW, INIT_MSRMT_DATA_ID	Local		
D1_INIT_MSRM T_DATA_CHAR	REFERENCE (D1_INIT_MSRMT _DATA_CHAR_FK)					
		D1T305P0	INIT_MSRMT_DATA_ID, CHAR_TYPE_CD, SEQ_NUM	Global Partitioned	RANGE(INIT_MSR MT_DATA_ID)	
		D1T305S1	SRCH_CHAR_VAL	Global Partitioned	HASH(SRCH_CHAR _VAL)	
D1_INIT_MSRM T_DATA_LOG	REFERENCE (D1_INIT_MSRMT _DATA_LOG_FK)					
		D1T306P0	INIT_MSRMT_DATA_ID, SEQNO	Global Partitioned	RANGE (INIT_MSRMT_DA TA_ID)	
D1_INIT_MSRM T_DATA_LOG_ PARM	REFERENCE (D1_INIT_MSRMT _DATA_LOG_PAR M_FK)					
		D1T307P0	INIT_MSRMT_DATA_ID, SEQNO PARM_SEQ	Global Partitioned	RANGE (INIT_MSRMT_DA TA_ID)	

Usage Transaction

This table describes the Usage Transaction 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
D1_USAGE (Parent)	RANGE(ILM_DT, D1_USAGE_ID)					D1_USAGE. CRE_DTTM
		D1T281P0	D1_USAGE_ID	Global Partitioned	RANGE (D1_USAGE_ID)	
		D1T281S0	US_ID, START_DTTM	Global Partitioned	RANGE (US_ID)	
		D1T281S1	BUS_OBJ_CD, BO_STATUS_CD, D1_USAGE_ID	Global Partitioned	HASH(BUS_OBJ_ CD, BO_STATUS_CD, D1_USAGE_ID)	
		CM_ILM_ D1T281S2	ILM_DT, ILM_ARCH_SW, D1_USAGE_ID	Local		
		D1T419S1	USG_EXT_ID, D1_USAGE_ID	Global Partitioned	RANGE (USG_EXT_ID)	
D1_USAGE_ CHAR	REFERENCE (D1_USAGE_ CHAR_FK)					
		D1T285P0	D1_USAGE_ID, CHAR_TYPE_CD, SEQ_NUM	Global Partitioned	RANGE(D1_ USAGE_ID)	
		D1T285S1	SRCH_CHAR_VAL	Global Partitioned	HASH(SRCH_CHAR _VAL)	
D1_USAGE_LOG	REFERENCE (D1_USAGE_LOG _FK)					
		D1T286P0	D1_USAGE_ID, SEQNO	Global Partitioned	RANGE(D1_USAG E_ID)	
		D1T286S1	CHAR_TYPE_CD, CHAR_VAL_FK1	Global Partitioned	HASH(CHAR_ TYPE_CD, CHAR_VAL_FK1)	
		D1T286S2	CHAR_TYPE_CD, CHAR_VAL	Global Partitioned	HASH(CHAR_ TYPE_CD, CHAR_VAL)	
D1_USAGE_LOG _PARM	REFERENCE(D1_ USAGE_LOG_ PARM_FK)					
		D1T287P0	D1_USAGE_ID, SEQNO PARM_SEQ	Global Partitioned	RANGE (D1_USAGE_ID)	
D1_USAGE_ PERIOD	REFERENCE(D1_ USAGE_PERIOD_ FK)					

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
		D1T283P0	D1_USAGE_ID, PERIOD_SEQ_NUM	Global Partitioned	RANGE(D1_ USAGE_ID)	
D1_USAGE_ PERIOD_ITEM_ DET	REFERENCE(D1_ USAGE_PERIOD_ ITEM_DET_FK)					
		D1T431P0	D1_USAGE_ID, PERIOD_SEQ_NUM, ITEM_SEQ_NUM	Global Partitioned	RANGE(D1_USAG E_ID)	
D1_USAGE_ PERIOD_SQ	REFERENCE(D1_ USAGE_PERIOD_ SQ_FK)					
		D1T284P0	D1_USAGE_ID, PERIOD_SEQ_NUM, SQ_SEQ_NUM	Global Partitioned	RANGE(D1_ USAGE_ID)	
D1_USAGE_REL	REFERENCE (D1_USAGE_REL_ FK)					
		D1T316P0	D1_USAGE_ID, USAGE_REL_TYPE_FLG	Global Partitioned	RANGE(D1_ USAGE_ID)	
		D1T316S0	REL_USAGE_ID, USAGE_REL_TYPE_FLG, D1_USAGE_ID	Global Partitioned	HASH(REL_USAGE _ID, USAGE_REL_TYPE _FLG, D1_USAGE_ID)	
D1_USAGE_ SCALAR_DTL	REFERENCE(D1_ USAGE_SCALAR_ DTL_FK)					
		D1T282P0	D1_USAGE_ID, D1_SP_ID, SEQ_NUM	Global Partitioned	RANGE(D1_ USAGE_ID)	

VEE Exception

This table describes the VEE Exception 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
D1_VEE_EXCP (Parent)	RANGE(ILM_DT, VEE_EXCP_ID)					D1_VEE_EXCP. CRE_DTTM
		D1T308P0	VEE_EXCP_ID	Global Partitioned	RANGE (VEE_EXCP_ID)	

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
		D1T308S1	INIT_MSRMT_DATA_ID	Global Partitioned	HASH(INIT_ MSRMT_DATA_ID)	
		CM_ILM_ D1T308S2	ILM_DT, ILM_ARCH_SW, VEE_EXCP_ID	Local		
D1_VEE_EXCP_ CHAR	REFERENCE (D1_VEE_EXCP_ CHAR_FK)					
		D1T310P0	VEE_EXCP_ID, CHAR_TYPE_CD, SEQ_NUM	Global Partitioned	RANGE(VEE_ EXCP_ID)	
		D1T310S1	SRCH_CHAR_VAL	Global Partitioned	HASH(SRCH_CHAR _VAL)	
D1_VEE_EXCP_ PARM	REFERENCE (D1_VEE_EXCP_ PARM_FK)					
		D1T309P0	VEE_EXCP_ID, PARM_SEQ	Global Partitioned	RANGE(VEE_ EXCP_ID))	

Snapshot Tables

This table below describes the snapshot tables.

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
D1_SNAPSHOT_ DL_CTRL	RANGE(ILM_DT)					D1_SNAPSHOT _DL_CTRL. SNAPSHOT_ DTTM
		D1T433P0	SNAPSHOT_FACT_NAME _CD, SNAPSHOT_DTTM	Global Partitioned		
		CM_ILM_ D1T433S1	ILM_DT, ILM_ARCH_SW, SNAPSHOT_FACT_ NAME_CD, SNAPSHOT_DTTM	Local		
D1_SP_SNAP_DL	RANGE(ILM_DT, SP_SNAP_ID)					D1_SP_SNAP_D L.SNAPSHOT_ DTTM
		D1T434P0	SP_SNAP_ID	Global Partitioned	RANGE(SP_SNAP_ ID)	

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
		D1T434S0	D1_SP_ID, SNAPSHOT_DTTM, SNAPSHOT_TYPE_FLG	Global Partitioned		
		CM_ILM_ D1T434S1	ILM_DT, ILM_ARCH_SW, SP_SNAP_ID	Local		
D1_SP_UNR_ USG_SNAP_DL	RANGE(ILM_DT, SP_UNR_USG_ SNAP_ID)					D1_SP_UNR_US G_SNAP_DL.SN APSHOT_DTTM
		D1T438P0	SP_UNR_USG_SNAP_ID	Global Partitioned	RANGE(SP_USG_S NAP_ID)	
		D1T436S0	D1_SP_ID, SNAPSHOT_DTTM, MEASR_COMP_ID, USG_SNAPSHOT_TYPE_ FLG, D1_TOU_CD, MSRMT_COND_FLG, SNAPSHOT_TYPE_FLG	Global Partitioned		
		CM_ILM_ D1T436S1	ILM_DT, ILM_ARCH_SW, SP_USG_SNAP_ID	Local		
D1_SP_VEE_ EXCP_SNAP_DL	RANGE(ILM_DT, SP_VEE_EXCP_ SNAP_ID)					D1_SP_VEE_ EXCP_SNAP_ DL.SNAPSHOT_ DTTM
		D1T440P0	SP_VEE_EXCP_SNAP_ID	Global Partitioned	RANGE(SP_VEE_ EXCP_SNAP_ID)	
		D1T440S0	D1_SP_ID, SNAPSHOT_DTTM, MEASR_COMP_ID, EXCP_TYPE_CD, D1_IMD_TYPE_FLG, EXCP_SEVERITY_FLG, VEE_GRP_CD, VEE_RULE_CD, SNAPSHOT_TYPE_FLG			
		CM_ILM_ D1T440S1	ILM_DT, ILM_ARCH_SW, SP_VEE_EXCP_SNAP_ID	Local		

Module Specific ILM Implementation Details For Sub Retention

This section outlines each maintenance object that has been configured to support ILM as well as sub retention periods. This differs from the standard ILM enabled tables in that the partitioning strategy is inclusive of an additional column that defines the retention period for each record. In each case, the recommendation of the initial load of the ILM_DT and the <field name for retention period> for existing records is noted. The CTAS operation for these tables includes an

extra step of generating a temporary mapping table that will allow the select for the ILM_DT to also identify the appropriate <retention period field name> for each record.

This section details the following maintenance objects that support ILM as well as sub retention periods:

- Activity
- Device Event
- Initial Measurement Data

Activity

Note: If sub retention periods will not be defined for this MO, then please follow the guidelines set forth in the Module Specific ILM Implementation Details section.

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
D1_ACTIVITY (Parent)	RANGE (ILM_DT, RETENTION_ PERIOD)					D1_ACTIVITY. CRE_DTTM
		D1T319P0	D1_ACTIVITY_ID	Global Partitioned	RANGE (D1_ACTIVITY_ID)	
		D1T319S0	BUS_OBJ_CD, BO_STATUS_CD, D1_ACTIVITY_ID	Global Partitioned	HASH(BUS_OBJ_CD, BO_STATUS_CD, D1_ACTIVITY_ID)	
		CM_ILM_ D1T319S1	ILM_DT, RETENTION_PERIOD, ILM_ARCH_SW, D1_ACTIVITY_ID	Local		
D1_ACTIVITY_ CHAR	REFERENCE (D1_ACTIVITY_ CHAR_FK)					
		D1T320P0	D1_ACTIVITY_ID, CHAR_TYPE_CD, SEQ_NUM	Global Partitioned	RANGE(D1_ ACTIVITY_ID)	
		D1T320S0	SRCH_CHAR_VAL	Global Partitioned	HASH(SRCH_CHAR _VAL)	
D1_ACTIVITY_ IDENTIFIER	REFERENCE (D1_ACTIVITY_ID ENTIFIER_FK)					
		D1T330P0	D1_ACTIVITY_ID, ACTIVITY_ID_TYPE_ FLG	Global Partitioned	RANGE(D1_ ACTIVITY_ID)	
		D1T330S0	ID_VALUE, ACTIVITY_ID_TYPE_ FLG	Global Partitioned	HASH(ID_VALUE, ACTIVITY_ID_ TYPE_FLG)	
D1_ACTIVITY_ LOG	REFERENCE (D1_ACTIVITY_ LOG_FK)					

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
		D1T321P0	D1_ACTIVITY_ID, SEQNO	Global Partitioned	RANGE(D1_ ACTIVITY_ID)	
		D1T321S1	CHAR_TYPE_CD, CHAR_VAL_FK1	Global Partitioned	HASH(CHAR_TYPE _CD, CHAR_VAL_FK1)	
		D1T321S2	CHAR_TYPE_CD, CHAR_VAL	Global Partitioned	HASH(CHAR_TYPE _CD, CHAR_VAL)	
D1_ACTIVITY_ LOG_PARM	REFERENCE (D1_ACTIVITY_ LOG_PARM_FK)					
		D1T322P0	D1_ACTIVITY_ID, SEQNO PARM_SEQ	Global Partitioned	RANGE(D1_ ACTIVITY_ID)	
D1_ACTIVITY_ REL	REFERENCE (D1_ACTIVITY_ REL_FK)					
		D1T323P0	D1_ACTIVITY_ID, ACTIVITY_REL_TYPE_ FLG	Global Partitioned	RANGE(D1_ ACTIVITY_ID)	
		D1T323S0	REL_ACTIVITY_ID	Global Partitioned	HASH(REL_ ACTIVITY_ID)	
D1_ACTIVITY_ REL_OBJ	REFERENCE (D1_ACTIVITY_ REL_OBJ_FK)					
		D1T324P0	D1_ACTIVITY_ID, MAINT_OBJ_CD, ACTIVITY_REL_OBJ_ TYPE_FLG	Global Partitioned	RANGE(D1_ ACTIVITY_ID)	
		D1T324S0	PK_VALUE1, PK_VALUE2, PK_VALUE3, PK_VALUE4, PK_VALUE5, MAINT_OBJ_CD	Global Partitioned	HASH(PK_VALUE1, PK_VALUE2, PK_VALUE3, PK_VALUE4)	

Query for Setting the Retention Period

The following query should be used to create a temporary table to create a mapping table that will identify the retention period for each measuring component type. This table will then be used during in the CTAS operation for Activity to identify the retention period for each record.

Please refer to Sample SQL for Enabling ILM with Sub Retention (Existing Installation) for detailed information using Initial Measurement Data as an example.

> **Note**: A pre-requisite to executing this query is configuring the appropriate retention periods in the ILM master configuration in the Oracle Utilities Meter Data Management application.

```
/*****ACTIVITY*****/
CREATE TABLE ILM_ACTIVITY_RETENTION_TMP
AS
select acty.activity_type_cd
/*retrieve the retention period for Activity Types in this order of
precedence:
1. The category based retention period from the MDM master
configuration
2. The MO level retention period from the MO options
3. The installation level retention period from the FW master
configuration
*/
, CAST(coalesce(catMap.retPeriod --Category level
, (select maint_obj_opt_val
from ci_md_mo_opt mmo
where maint_obj_cd = 'D1-ACTIVITY'
and maint_obj_opt_flg ='FLRP'
and seq_num =
(select max(seq_num)
from ci_md_mo_opt mmo
where maint_obj_cd = 'D1-ACTIVITY'
and maint_obj_opt_flg ='FLRP')) --MO level
, extractvalue( xmlparse(content fw_mcfg.mst_config_data)
,'generalMasterConfiguration/defaultRetentionPeriod') --Install level
) as NUMBER(5)) retPeriod
from d1_activity_type acty
, (select extractvalue(value(p),
'activityTypeCategoryRetentionPeriodList/activityTypeCategory'
)ACTIVITY_TYPE_CAT_FLG
, extractvalue(value(p),
'activityTypeCategoryRetentionPeriodList/retentionPeriod'
)retPeriod
from f1_mst_config mdm_mcfg ,
table(xmlsequence(extract(xmlparse(content mdm_mcfg.mst_config_data),
'activityRetentionPeriod/activityTypeCategoryRetentionPeriods/
activityTypeCategoryRetentionPeriodList'
))) p
where mdm_mcfg.bus_obj_cd = 'D1-ILMMSConfig')catMap
, f1_mst_config fw_mcfg
where fw_mcfg.bus_obj_cd = 'F1-ILMMSConfig'
and acty.ACTIVITY_TYPE_CAT_FLG = catMap.ACTIVITY_TYPE_CAT_FLG (+)
order by 1;
```

Device Event

Note: If sub retention periods will not be defined for this MO, then please follow the guidelines set forth in the Module Specific ILM Implementation Details section.

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
D1_DVC_EVT (Parent)	RANGE(ILM_DT, RETENTION_ PERIOD)					D1_DVC_EVT. CRE_DTTM
		D1T400P0	DVC_EVT_ID	Global Partitioned	RANGE (DVC_EVT_ID)	
		D1T400S1	BUS_OBJ_CD, BO_STATUS_CD, DVC_EVT_ID	Global Partitioned	HASH(BUS_OBJ_C D,BO_STATUS_CD, DVC_EVT_ID)	
		D1T400S2	D1_DEVICE_ID, DVC_EVT_DTTM	Global Partitioned	HASH(D1_DEVICE _ID, DVC_EVT_DTTM)	
		D1T400S3	BUS_OBJ_CD, BO_STATUS_CD, D1_DEVICE_ID, DVC_EVT_ID	Global Partitioned	HASH(BUS_OBJ_C D,BO_STATUS_CD, D1_DEVICE_ID, DVC_EVT_ID)	
		CM_ILM_ D1T400S4	ILM_DT, RETENTION_PERIOD, ILM_ARCH_SW, DVC_EVT_ID	Local		
D1_DVC_EVT_ CHAR	REFERENCE (D1_DVC_EVT_ CHAR_FK)					
		D1T401P0	DVC_EVT_ID, CHAR_TYPE_CD, SEQ_NUM	Global Partitioned	RANGE(DVC_EVT _ID)	
		D1T401S0	SRCH_CHAR_VAL	Global Partitioned	HASH(SRCH_CHAR _VAL)	
D1_DVC_EVT_ IDENTIFIER	REFERENCE (D1_DVC_EVT_ IDENTIFIER_FK)					
		D1T405P0	DVC_EVT_ID, DVC_EVT_ID_TYPE_FLG	Global Partitioned	RANGE(DVC_EVT _ID)	
		D1T405S0	ID_VALUE, DVC_EVT_ID_TYPE_FLG	Global Partitioned	HASH(ID_VALUE, DVC_EVT_ID_ TYPE_FLG)	
D1_DVC_EVT_ LOG	REFERENCE (D1_DVC_EVT_ LOG_FK)					
		D1T402P0	DVC_EVT_ID, SEQNO	Global Partitioned	RANGE(DVC_EVT _ID)	

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
		D1T402S1	CHAR_TYPE_CD, CHAR_VAL_FK1	Global Partitioned	HASH(CHAR_ TYPE_CD, CHAR_VAL_FK1)	
		D1T402S2	CHAR_TYPE_CD, CHAR_VAL	Global Partitioned	HASH(CHAR_ TYPE_CD, CHAR_VAL)	
D1_DVC_EVT_ LOG_PARM	REFERENCE (D1_DVC_EVT_ LOG_PARM_FK)					
		D1T403P0	DVC_EVT_ID, SEQNO PARM_SEQ	Global Partitioned	RANGE(DVC_EVT _ID)	
D1_DVC_EVT_ REL_OBJ	REFERENCE (D1_DVC_EVT_ REL_OBJ_FK)					
		D1T404P0	DVC_EVT_ID, MAINT_OBJ_CD, DVC_EVT_REL_OBJ_TYP E_FLG	Global Partitioned	RANGE(DVC_EVT _ID)	
		D1T404S0	PK_VALUE1, PK_VALUE2, PK_VALUE3, PK_VALUE4, PK_VALUE5, MAINT_OBJ_CD	Global Partitioned	HASH(PK_VALUE1 , PK_VALUE2, PK_VALUE3, PK_VALUE4)	

Query for Setting the Retention Period

The following query should be used to create a temporary table to create a mapping table that will identify the retention period for each measuring component type. This table will then be used during in the CTAS operation for Device Event to identify the retention period for each record.

Please refer to Sample SQL for Enabling ILM with Sub Retention (Existing Installation) for detailed information using Initial Measurement Data as an example.

> **Note**: A pre-requisite to executing this query is configuring the appropriate retention periods in the ILM master configuration in the Oracle Utilities Meter Data Management application.

```
CREATE TABLE ILM_DVC_EVT_RETENTION_TMP
AS
select det.dvc_evt_type_cd
/*retrieve the retention period for Device Event Types in this order of
precedence:
1. The category based retention period from the MDM master
configuration
2. The MO level retention period from the MO options
3. The installation level retention period from the FW master
configuration
*/
, CAST(coalesce(catMap.retPeriod --Category level
```

```
, (select maint_obj_opt_val
from ci_md_mo_opt mmo
where maint_obj_cd = 'D1-DVCEVENT'
and maint_obj_opt_flg ='FLRP'
and seq_num = (select max(seq_num)
from ci_md_mo_opt mmo
where maint_obj_cd = 'D1-DVCEVENT'
and maint_obj_opt_flg ='FLRP')) --MO level
, extractvalue( xmlparse(content
fw_mcfg.mst_config_data),
'generalMasterConfiguration/defaultRetentionPeriod') --Install level
) as NUMBER(5)) retPeriod
from d1_dvc_evt_type det
, (select extractvalue(value(p),
'deviceEventCategoryRetentionPeriodList/deviceEventCategory')
dvc_evt_cat_flg
, extractvalue(value(p),
'deviceEventCategoryRetentionPeriodList/retentionPeriod') retPeriod
from f1_mst_config mdm_mcfg ,
table(xmlsequence(extract(xmlparse(content
mdm_mcfg.mst_config_data),
'deviceEventRetentionPeriod/deviceEventCategoryRetentionPeriods/
deviceEventCategoryRetentionPeriodList'
))) p
where mdm_mcfg.bus_obj_cd = 'D1-ILMMSConfig')catMap
, f1_mst_config fw_mcfg
where fw_mcfg.bus_obj_cd = 'F1-ILMMSConfig'
and det.dvc_evt_cat_flg = catMap.dvc_evt_cat_flg (+)
order by 1;
```

Initial Measurement Data

Note: If sub retention periods will not be defined for this MO, then please follow the guidelines set forth in Module Specific ILM Implementation Details section.

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
D1_INIT_MSRMT_DATA (Parent)	RANGE (ILM_DT, RETENTION_ PERIOD)					D1_INIT_MSRMT_DATA. CRE_DTTM
		D1T304P0	INIT_MSRMT_DATA_ID	Global Partitioned	RANGE (INIT_MSRMT_DATA_ID)	
		D1T304S1	MEASR_COMP_ID, BO_STATUS_CD, BUS_OBJ_CD, D1_TO_DTTM, D1_FROM_DTTM	Global Partitioned	RANGE (MEASR_COMP_ID)	
		D1T304S2	BUS_OBJ_CD, BO_STATUS_CD, INIT_MSRMT_DATA_ID	Global Partitioned	HASH (BUS_OBJ_CD, BO_STATUS_CD, INIT_MSRMT_DATA_ID)	

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
		D1T304S3	IMD_EXT_ID, INIT_MSRMT_DATA_ID	Global Partitioned	HASH(IMD_EXT_ID)	
		CM_ILM_D1T304S4	ILM_DT, RETENTION_PERIOD, ILM_ARCH_SW, INIT_MSRMT_DATA_ID	Local		
D1_INIT_MSRMT_DATA_CHAR	REFERENCE (D1_INIT_MSRMT_DATA_CHAR_FK)					
		D1T305P0	INIT_MSRMT_DATA_ID, CHAR_TYPE_CD, SEQ_NUM	Global Partitioned	RANGE(INIT_MSRMT_DATA_ID)	
		D1T305S1	SRCH_CHAR_VAL	Global Partitioned	HASH(SRCH_CHAR_VAL)	
D1_INIT_MSRMT_DATA_LOG	REFERENCE (D1_INIT_MSRMT_DATA_LOG_FK)					
		D1T306P0	INIT_MSRMT_DATA_ID, SEQNO	Global Partitioned	RANGE (INIT_MSRMT_DATA_ID)	
D1_INIT_MSRMT_DATA_LOG_PARM	REFERENCE (D1_INIT_MSRMT_DATA_LOG_PARM_FK)					
		D1T307P0	INIT_MSRMT_DATA_ID, SEQNO PARM_SEQ	Global Partitioned	RANGE (INIT_MSRMT_DATA_ID)	

Query for Setting the Retention Period

The following query should be used to create a temporary table to create a mapping table that will identify the retention period for each measuring component type. This table will then be used during in the CTAS operation for Initial Measurement Data to identify the retention period for each record.

Please refer to Sample SQL for Enabling ILM with Sub Retention (Existing Installation) for detailed information using Initial Measurement Data as an example.

> **Note**: A pre-requisite to executing this query is configuring the appropriate retention periods in the ILM master configuration in the Oracle Utilities Meter Data Management application.

```
CREATE TABLE ILM_IMD_RETENTION_TMP
AS
select mct.measr_comp_type_cd
/*retrieve the retention period for MC Types in this order of precedence:
1. The UOM based retention period from the MDM master configuration
2. The interval IMD retention period from the MDM master configuration
3. The MO level retention period from the MO options
```

```
4. The installation level retention period from the FW master configuration
*/
, CAST(coalesce( (select retPeriod
from (select 'D1IN' interval_scalar_flg
, extractvalue(value(p),'uomRetentionPeriodList/uom') D1_UOM_CD
, extractvalue(value(p),'uomRetentionPeriodList/retentionPeriod') retPeriod
from f1_mst_config mdm_mcfg
, table(xmlsequence(extract(xmlparse(content
mdm_mcfg.mst_config_data),
'imdRetentionPeriod/intervalImdRetentionPeriods/uomRetentionPeriods/
uomRetentionPeriodList'))) p
where mdm_mcfg.bus_obj_cd = 'D1-ILMMSConfig'
union
select 'D1SC' INTERVAL_SCALAR_FLG
, extractvalue(value(p),'uomRetentionPeriodList/uom') D1_UOM_CD
, extractvalue(value(p),'uomRetentionPeriodList/retentionPeriod') retPeriod
from f1_mst_config mdm_mcfg
, table(xmlsequence(extract(xmlparse(content
mdm_mcfg.mst_config_data),
'imdRetentionPeriod/scalarImdRetentionPeriods/uomRetentionPeriods/
uomRetentionPeriodList'))) p
where mdm_mcfg.bus_obj_cd = 'D1-ILMMSConfig') uomMap
where uomMap.interval_scalar_flg = mct.interval_scalar_flg
and trim(mctvi.d1_uom_cd) = trim(uomMap.d1_uom_cd))--UOM
, DECODE(mct.interval_scalar_flg
,'D1IN'
,extractvalue( xmlparse(content mdm_mcfg.mst_config_data),
'imdRetentionPeriod/intervalImdRetentionPeriods/intervalRetentionPeriod') --
interval IMD
,extractvalue( xmlparse(content mdm_mcfg.mst_config_data),
'imdRetentionPeriod/scalarImdRetentionPeriods/scalarRetentionPeriod') --scalar
IMD
)
, (select maint_obj_opt_val
from ci_md_mo_opt mmo
where maint_obj_cd = 'D1-IMD'
and maint_obj_opt_flg ='FLRP'
and seq_num = (select max(seq_num)
from ci_md_mo_opt mmo
where maint_obj_cd = 'D1-IMD'
and maint_obj_opt_flg ='FLRP')) --IMD
, extractvalue( xmlparse(content fw_mcfg.mst_config_data),
'generalMasterConfiguration/defaultRetentionPeriod') --Install
) as NUMBER(5)) retPeriod
from d1_measr_comp_type mct
, d1_mc_type_value_identifier mctvi
, f1_mst_config fw_mcfg
, f1_mst_config mdm_mcfg
where mct.measr_comp_type_cd = mctvi.measr_comp_type_cd
and mctvi.value_id_type_flg = 'D1MS'
and fw_mcfg.bus_obj_cd = 'F1-ILMMSConfig'
and mdm_mcfg.bus_obj_cd = 'D1-ILMMSConfig'
order by 1;
```

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 Sample SQL for Periodic Maintenance for detailed information using To Do Entry(Without LOB), F1_SYNC_REC_IN(With LOB-Tablespace per Partition), Initial Measurement Data (With LOB-Tablespace per Subpartition), and the D1_MSRMT table (Partition Compression) 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/subpartition:
    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, then proceed for 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/subpartition 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/subpartition.
    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. Compress D1_MSRMT table Partition:
    a. Create new compressed tablespace.

b. Create a table using CTAS for each subpartition of the partition being compressed in the new compressed tablespace.

c. Create a unique primary index for each subpartition of the partition being compressed in the new compressed tablespace. Then, alter table to create the primary key.

d. Create a unique secondary index for each subpartition of the partition being compressed in the new compressed tablespace.

e. Exchange the subpartition of the D1_MSRMT table with the newly created table for each subpartition.

f. Drop the original uncompressed tablespace.

g. Alter the partition level metadata to reflect the new compressed tablespace.

h. Rename the new compressed tablespace to the original tablespace name.

5. 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. Remember to assign transportable tablespaces for the archive/dormant data stage to allow for safe removal of the data.
- 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:

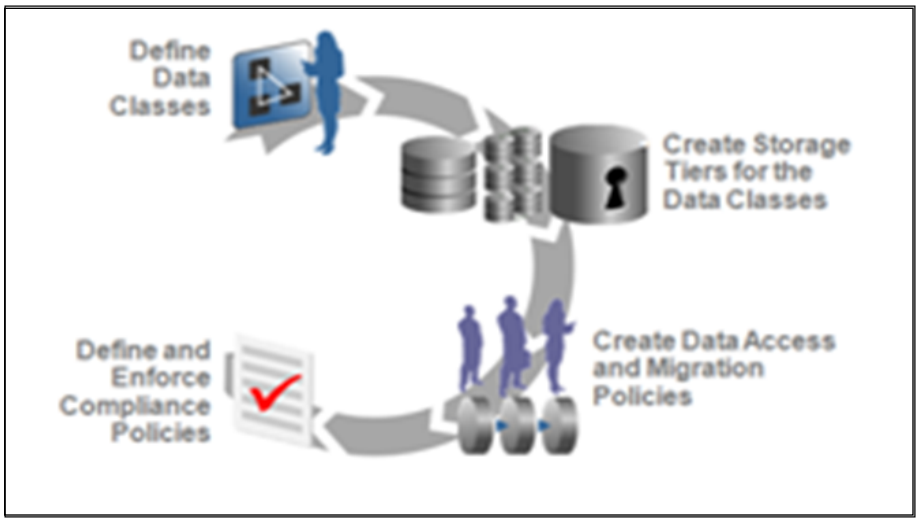
  https://docs.oracle.com/database/121/VLDBG/title.htm

ILM Assistant

The ILM Assistant can provide the following:

- Setup ILM Lifecycle definition - Here you can define different lifecycle definitions for different MOs and configure when the data is ready to be moved to a slower disk.
- Setup ILM Lifecycle tables - Here you define the tables you want to manage and assign it to a Lifecycle definition defined above. You can setup policies so that when data is moved from one partition to another it will be automatically compressed to a desired degree.
- Lifecycle Management - There is a tab called Lifecycle Management where the system admin will be alerted when partitions are eligible for archiving.

ILM Assistant can then be used to ensure the records that have ILM_ARCH_SW = 'Y' can be archived or purged, as deemed appropriate by the business.



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 more details on the convention, refer to the table below:

Convention	Description
OWNERFLAG	Owner flag for the relevant application for example “D1” for MDM
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. P2011JAN • PMAX if it is the Max Value partition

Convention	Description
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
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_PARTITION NAME. For example: CM_D1T304_PMAX, CM_D1T304_P2011JAN • SubPartition Tablespace Name: It is formed by OWNERFLAG_TABLEIDENTIFIER_PARTITION NAME_SUBPARTITIONNAME For example: CM_D1T304_PMAX_SMAX , CM_D1T304_P2011JAN_SMAX, CM_D1T304_PMAX_S001, CM_D1T304_P2011JAN_S181 • Archive Staging Table and its Tablespace Name (When archiving partition): It is formed by OWNERFLAG_TABLEIDENTIFIER_PARTITION NAME_ ARCHIVEFLAG For example: CM_D1T304_P2011JAN_ARC • Archive Staging Table and its Tablespace Name (while archiving subpartition): It is formed by OWNERFLAG_TABLEIDENTIFIER_PARTITION NAME_SUBPARTITIONNAME_ ARCHIVEFLAG For example: CM_D1T304_P2011JAN_S181_ARC • Compressed Tablespace Name (while compressing the partition) For example: CM_D1T304_P2011JAN_ C

Appendix A

Sample SQL for Enabling ILM (Initial Installation)

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

The first is To Do Entry, which does not include a LOB field. The second is Sync Request, which does include a LOB field and has one tablespace per partition. The third is Initial Measurement Data, which includes LOB fields and has one tablespace per subpartition (shown using subretention). Other maintenance object's implementations can follow the appropriate pattern based on whether there is a LOB field or not.

The following DDL(s):

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

Maintenance Object: TO DO ENTRY

Parent Table: CI_TD_ENTRY

```
CREATE BIGFILE TABLESPACE CM_XT039_P2011JAN DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_XT039_P2011FEB DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_XT039_P2011MAR DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_XT039_P2011APR DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_XT039_P2011MAY DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_XT039_P2011JUN DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_XT039_P2011JUL DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_XT039_P2011AUG DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_XT039_P2011SEP DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_XT039_P2011OCT DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_XT039_P2011NOV DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
```

```
CREATE BIGFILE TABLESPACE CM_XT039_P2011DEC 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 DEFAULT SYSDATE,
ILM_ARCH_SW CHAR(1) DEFAULT 'N' NOT NULL ENABLE
)
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 "P2011JAN" VALUES LESS THAN (TO_DATE('2011-02-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
TABLESPACE CM_XT039_P2011JAN,
PARTITION "P2011FEB" VALUES LESS THAN (TO_DATE('2011-03-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
TABLESPACE CM_XT039_P2011FEB,
PARTITION "P2011MAR" VALUES LESS THAN (TO_DATE('2011-04-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
TABLESPACE CM_XT039_P2011MAR,
PARTITION "P2011APR" VALUES LESS THAN (TO_DATE('2011-05-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
TABLESPACE CM_XT039_P2011APR,
PARTITION "P2011MAY" VALUES LESS THAN (TO_DATE('2011-06-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
TABLESPACE CM_XT039_P2011MAY,
PARTITION "P2011JUN" VALUES LESS THAN (TO_DATE('2011-07-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
TABLESPACE CM_XT039_P2011JUN,
PARTITION "P2011JUL" VALUES LESS THAN (TO_DATE('2011-08-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
TABLESPACE CM_XT039_P2011JUL,
PARTITION "P2011AUG" VALUES LESS THAN (TO_DATE('2011-09-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
TABLESPACE CM_XT039_P2011AUG,
PARTITION "P2011SEP" VALUES LESS THAN (TO_DATE('2011-10-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
TABLESPACE CM_XT039_P2011SEP,
PARTITION "P2011OCT" VALUES LESS THAN (TO_DATE('2011-11-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
TABLESPACE CM_XT039_P2011OCT,
PARTITION "P2011NOV" VALUES LESS THAN (TO_DATE('2011-12-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
TABLESPACE CM_XT039_P2011NOV,
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 ) 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 CM_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
)
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
)
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
)
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
)
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
)
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_P2011JAN DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_F1T191_P2011FEB DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_F1T191_P2011MAR DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_F1T191_P2011APR DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_F1T191_P2011MAY DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_F1T191_P2011JUN DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_F1T191_P2011JUL DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_F1T191_P2011AUG DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_F1T191_P2011SEP DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_F1T191_P2011OCT DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_F1T191_P2011NOV DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_F1T191_P2011DEC 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(14) 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 "P2011JAN" VALUES LESS THAN (TO_DATE('2011-02-01 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_P2011JAN )
LOB(PRE_TRN_INIT_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM
CACHE TABLESPACE CM_F1T191_P2011JAN )
LOB(PRE_TRN_FIN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM
CACHE TABLESPACE CM_F1T191_P2011JAN )
LOB(POST_TRN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM CACHE
TABLESPACE CM_F1T191_P2011JAN )
TABLESPACE CM_F1T191_P2011JAN,
PARTITION "P2011FEB" VALUES LESS THAN (TO_DATE('2011-03-01 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_P2011FEB )
LOB(PRE_TRN_INIT_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM
CACHE TABLESPACE CM_F1T191_P2011FEB )
LOB(PRE_TRN_FIN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM
CACHE TABLESPACE CM_F1T191_P2011FEB )
LOB(POST_TRN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM CACHE
TABLESPACE CM_F1T191_P2011FEB )
TABLESPACE CM_F1T191_P2011FEB,
PARTITION "P2011MAR" VALUES LESS THAN (TO_DATE('2011-04-01 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_P2011MAR )
LOB(PRE_TRN_INIT_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM
CACHE TABLESPACE CM_F1T191_P2011MAR )
LOB(PRE_TRN_FIN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM
CACHE TABLESPACE CM_F1T191_P2011MAR )
LOB(POST_TRN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM CACHE
TABLESPACE CM_F1T191_P2011MAR )
TABLESPACE CM_F1T191_P2011MAR,
PARTITION "P2011APR" VALUES LESS THAN (TO_DATE('2011-05-01 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_P2011APR )
LOB(PRE_TRN_INIT_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM
CACHE TABLESPACE CM_F1T191_P2011APR )
LOB(PRE_TRN_FIN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM
CACHE TABLESPACE CM_F1T191_P2011APR )
LOB(POST_TRN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM CACHE
TABLESPACE CM_F1T191_P2011APR )
TABLESPACE CM_F1T191_P2011APR,
PARTITION "P2011MAY" VALUES LESS THAN (TO_DATE('2011-06-01 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_P2011MAY )
LOB(PRE_TRN_INIT_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM
CACHE TABLESPACE CM_F1T191_P2011MAY )
LOB(PRE_TRN_FIN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM
CACHE TABLESPACE CM_F1T191_P2011MAY )
LOB(POST_TRN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM CACHE
TABLESPACE CM_F1T191_P2011MAY )
TABLESPACE CM_F1T191_P2011MAY,
PARTITION "P2011JUN" VALUES LESS THAN (TO_DATE('2011-07-01 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_P2011JUN )
LOB(PRE_TRN_INIT_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM
CACHE TABLESPACE CM_F1T191_P2011JUN )
LOB(PRE_TRN_FIN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM
CACHE TABLESPACE CM_F1T191_P2011JUN )
LOB(POST_TRN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM CACHE
TABLESPACE CM_F1T191_P2011JUN )
TABLESPACE CM_F1T191_P2011JUN,
PARTITION "P2011JUL" VALUES LESS THAN (TO_DATE('2011-08-01 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_P2011JUL )
LOB(PRE_TRN_INIT_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM
CACHE TABLESPACE CM_F1T191_P2011JUL )
LOB(PRE_TRN_FIN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM
CACHE TABLESPACE CM_F1T191_P2011JUL )
LOB(POST_TRN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM CACHE
TABLESPACE CM_F1T191_P2011JUL )
TABLESPACE CM_F1T191_P2011JUL,
PARTITION "P2011AUG" VALUES LESS THAN (TO_DATE('2011-09-01 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_P2011AUG )
LOB(PRE_TRN_INIT_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM
CACHE TABLESPACE CM_F1T191_P2011AUG )
LOB(PRE_TRN_FIN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM
CACHE TABLESPACE CM_F1T191_P2011AUG )
LOB(POST_TRN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM CACHE
TABLESPACE CM_F1T191_P2011AUG )
```

```
TABLESPACE CM_F1T191_P2011AUG,
PARTITION "P2011SEP" VALUES LESS THAN (TO_DATE('2011-10-01 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_P2011SEP )
LOB(PRE_TRN_INIT_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM
CACHE TABLESPACE CM_F1T191_P2011SEP )
LOB(PRE_TRN_FIN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM
CACHE TABLESPACE CM_F1T191_P2011SEP )
LOB(POST_TRN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM CACHE
TABLESPACE CM_F1T191_P2011SEP )
TABLESPACE CM_F1T191_P2011SEP,
PARTITION "P2011OCT" VALUES LESS THAN (TO_DATE('2011-11-01 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_P2011OCT )
LOB(PRE_TRN_INIT_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM
CACHE TABLESPACE CM_F1T191_P2011OCT )
LOB(PRE_TRN_FIN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM
CACHE TABLESPACE CM_F1T191_P2011OCT )
LOB(POST_TRN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM CACHE
TABLESPACE CM_F1T191_P2011OCT )
TABLESPACE CM_F1T191_P2011OCT,
PARTITION "P2011NOV" VALUES LESS THAN (TO_DATE('2011-12-01 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_P2011NOV )
LOB(PRE_TRN_INIT_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM
CACHE TABLESPACE CM_F1T191_P2011NOV )
LOB(PRE_TRN_FIN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM
CACHE TABLESPACE CM_F1T191_P2011NOV )
LOB(POST_TRN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM CACHE
TABLESPACE CM_F1T191_P2011NOV )
TABLESPACE CM_F1T191_P2011NOV,
PARTITION "P2011DEC" VALUES LESS THAN (TO_DATE('2012-01-01 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_P2011DEC )
LOB(PRE_TRN_INIT_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM
CACHE TABLESPACE CM_F1T191_P2011DEC )
LOB(PRE_TRN_FIN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM
CACHE TABLESPACE CM_F1T191_P2011DEC )
LOB(POST_TRN_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM CACHE
TABLESPACE CM_F1T191_P2011DEC )
TABLESPACE CM_F1T191_P2011DEC,
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
)
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
)
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
)
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
)
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
)
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
)
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;
```

Maintenance Object: D1-IMD

Parent Table: D1_INIT_MSRMT_DATA

```
CREATE BIGFILE TABLESPACE CM_D1T304_P2011JAN_S181 DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011JAN_SMAX DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011FEB_S181 DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011FEB_SMAX DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011MAR_S181 DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011MAR_SMAX DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011APR_S181 DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011APR_SMAX DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011MAY_S181 DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011MAY_SMAX DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011JUN_S181 DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011JUN_SMAX DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011JUL_S181 DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011JUL_SMAX DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011AUG_S181 DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011AUG_SMAX DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011SEP_S181 DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011SEP_SMAX DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011OCT_S181 DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011OCT_SMAX DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011NOV_S181 DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011NOV_SMAX DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011DEC_S181 DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011DEC_SMAX DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_PMAX_S181 DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_PMAX_SMAX DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;

CREATE TABLE D1_INIT_MSRMT_DATA
  (
   INIT_MSRMT_DATA_ID CHAR(14) NOT NULL ENABLE,
    MEASR_COMP_ID      CHAR(12) DEFAULT ' ' NOT NULL ENABLE,
    D1_FROM_DTTM DATE,
    D1_TO_DTTM DATE,
    DATA_SRC_FLG        CHAR(4) DEFAULT ' ' NOT NULL ENABLE,
    TIME_ZONE_CD        CHAR(10) DEFAULT ' ' NOT NULL ENABLE,
    BUS_OBJ_CD          CHAR(30) DEFAULT ' ' NOT NULL ENABLE,
    BO_STATUS_CD        CHAR(12) DEFAULT ' ' NOT NULL ENABLE,
    BO_STATUS_REASON_CD VARCHAR2(30) DEFAULT ' ' NOT NULL ENABLE,
    IMD_BO_DATA_AREA CLOB,
    STATUS_UPD_DTTM DATE NOT NULL ENABLE,
    CRE_DTTM DATE NOT NULL ENABLE,
    VERSION    NUMBER(5,0) DEFAULT 1 NOT NULL ENABLE,
    IMD_EXT_ID VARCHAR2(120),
    PREVEE_BO_DATA_AREA CLOB,
    POSTVEE_BO_DATA_AREA CLOB,
    TRACE_BO_DATA_AREA CLOB,
    RAW_BO_DATA_AREA CLOB,
    LAST_UPDATE_DTTM DATE,
    ILM_DT DATE,
    ILM_ARCH_SW CHAR(1),
    RETENTION_PERIOD NUMBER(5,0) DEFAULT 99999 NOT NULL ENABLE
```

```
    )
ENABLE ROW MOVEMENT
LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM CACHE)
LOB ( POSTVEE_BO_DATA_AREA ) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM
CACHE)
LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM CACHE)
LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM CACHE)
LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM CACHE)
PARTITION BY RANGE (ILM_DT)
SUBPARTITION BY range (RETENTION_PERIOD)
(
PARTITION "P2011JAN" VALUES LESS THAN (TO_DATE('2011-02-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))(
SUBPARTITION P2011JAN_S181 VALUES LESS THAN (181) TABLESPACE CM_D1T304_P2011JAN_S181
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JAN_S181)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JAN_S181)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JAN_S181)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JAN_S181)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JAN_S181)
,
SUBPARTITION P2011JAN_SMAX VALUES LESS THAN (MAXVALUE) TABLESPACE CM_D1T304_P2011JAN_SMAX
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JAN_SMAX)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JAN_SMAX)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JAN_SMAX)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JAN_SMAX)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JAN_SMAX)
),
PARTITION "P2011FEB" VALUES LESS THAN (TO_DATE('2011-03-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))(
SUBPARTITION P2011FEB_S181 VALUES LESS THAN (181) TABLESPACE CM_D1T304_P2011FEB_S181
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011FEB_S181)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011FEB_S181)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011FEB_S181)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011FEB_S181)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011FEB_S181)
,
SUBPARTITION P2011FEB_SMAX VALUES LESS THAN (MAXVALUE) TABLESPACE CM_D1T304_P2011FEB_SMAX
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011FEB_SMAX)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011FEB_SMAX)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011FEB_SMAX)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011FEB_SMAX)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011FEB_SMAX)
),
PARTITION "P2011MAR" VALUES LESS THAN (TO_DATE('2011-04-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))(
SUBPARTITION P2011MAR_S181 VALUES LESS THAN (181) TABLESPACE CM_D1T304_P2011MAR_S181
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAR_S181)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAR_S181)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAR_S181)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAR_S181)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAR_S181)
,
SUBPARTITION P2011MAR_SMAX VALUES LESS THAN (MAXVALUE) TABLESPACE CM_D1T304_P2011MAR_SMAX
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAR_SMAX)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAR_SMAX)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAR_SMAX)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAR_SMAX)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAR_SMAX)
),
PARTITION "P2011APR" VALUES LESS THAN (TO_DATE('2011-05-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))(
SUBPARTITION P2011APR_S181 VALUES LESS THAN (181) TABLESPACE CM_D1T304_P2011APR_S181
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011APR_S181)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011APR_S181)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011APR_S181)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011APR_S181)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011APR_S181)
,
SUBPARTITION P2011APR_SMAX VALUES LESS THAN (MAXVALUE) TABLESPACE CM_D1T304_P2011APR_SMAX
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011APR_SMAX)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011APR_SMAX)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011APR_SMAX)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011APR_SMAX)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011APR_SMAX)
),
PARTITION "P2011MAY" VALUES LESS THAN (TO_DATE('2011-06-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))(
SUBPARTITION P2011MAY_S181 VALUES LESS THAN (181) TABLESPACE CM_D1T304_P2011MAY_S181
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAY_S181)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAY_S181)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAY_S181)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAY_S181)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAY_S181)
,
SUBPARTITION P2011MAY_SMAX VALUES LESS THAN (MAXVALUE) TABLESPACE CM_D1T304_P2011MAY_SMAX
```

```
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAY_SMAX)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAY_SMAX)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAY_SMAX)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAY_SMAX)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAY_SMAX)
),
PARTITION "P2011JUN" VALUES LESS THAN (TO_DATE('2011-07-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))(
SUBPARTITION P2011JUN_S181 VALUES LESS THAN (181) TABLESPACE CM_D1T304_P2011JUN_S181
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUN_S181)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUN_S181)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUN_S181)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUN_S181)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUN_S181)
,
SUBPARTITION P2011JUN_SMAX VALUES LESS THAN (MAXVALUE) TABLESPACE CM_D1T304_P2011JUN_SMAX
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUN_SMAX)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUN_SMAX)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUN_SMAX)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUN_SMAX)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUN_SMAX)
),
PARTITION "P2011JUL" VALUES LESS THAN (TO_DATE('2011-08-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))(
SUBPARTITION P2011JUL_S181 VALUES LESS THAN (181) TABLESPACE CM_D1T304_P2011JUL_S181
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUL_S181)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUL_S181)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUL_S181)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUL_S181)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUL_S181)
,
SUBPARTITION P2011JUL_SMAX VALUES LESS THAN (MAXVALUE) TABLESPACE CM_D1T304_P2011JUL_SMAX
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUL_SMAX)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUL_SMAX)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUL_SMAX)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUL_SMAX)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUL_SMAX)
),
PARTITION "P2011AUG" VALUES LESS THAN (TO_DATE('2011-09-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))(
SUBPARTITION P2011AUG_S181 VALUES LESS THAN (181) TABLESPACE CM_D1T304_P2011AUG_S181
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011AUG_S181)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011AUG_S181)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011AUG_S181)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011AUG_S181)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011AUG_S181)
,
SUBPARTITION P2011AUG_SMAX VALUES LESS THAN (MAXVALUE) TABLESPACE CM_D1T304_P2011AUG_SMAX
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011AUG_SMAX)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011AUG_SMAX)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011AUG_SMAX)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011AUG_SMAX)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011AUG_SMAX)
),
PARTITION "P2011SEP" VALUES LESS THAN (TO_DATE('2011-10-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))(
SUBPARTITION P2011SEP_S181 VALUES LESS THAN (181) TABLESPACE CM_D1T304_P2011SEP_S181
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011SEP_S181)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011SEP_S181)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011SEP_S181)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011SEP_S181)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011SEP_S181)
,
SUBPARTITION P2011SEP_SMAX VALUES LESS THAN (MAXVALUE) TABLESPACE CM_D1T304_P2011SEP_SMAX
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011SEP_SMAX)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011SEP_SMAX)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011SEP_SMAX)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011SEP_SMAX)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011SEP_SMAX)
),
PARTITION "P2011OCT" VALUES LESS THAN (TO_DATE('2011-11-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))(
SUBPARTITION P2011OCT_S181 VALUES LESS THAN (181) TABLESPACE CM_D1T304_P2011OCT_S181
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011OCT_S181)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011OCT_S181)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011OCT_S181)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011OCT_S181)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011OCT_S181)
,
SUBPARTITION P2011OCT_SMAX VALUES LESS THAN (MAXVALUE) TABLESPACE CM_D1T304_P2011OCT_SMAX
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011OCT_SMAX)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011OCT_SMAX)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011OCT_SMAX)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011OCT_SMAX)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011OCT_SMAX)
```

```
),
PARTITION "P2011NOV" VALUES LESS THAN (TO_DATE('2011-12-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))(
SUBPARTITION P2011NOV_S181 VALUES LESS THAN (181) TABLESPACE CM_D1T304_P2011NOV_S181
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011NOV_S181)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011NOV_S181)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011NOV_S181)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011NOV_S181)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011NOV_S181)
,
SUBPARTITION P2011NOV_SMAX VALUES LESS THAN (MAXVALUE) TABLESPACE CM_D1T304_P2011NOV_SMAX
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011NOV_SMAX)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011NOV_SMAX)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011NOV_SMAX)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011NOV_SMAX)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011NOV_SMAX)
),
PARTITION "P2011DEC" VALUES LESS THAN (TO_DATE('2012-01-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))(
SUBPARTITION P2011DEC_S181 VALUES LESS THAN (181) TABLESPACE CM_D1T304_P2011DEC_S181
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011DEC_S181)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011DEC_S181)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011DEC_S181)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011DEC_S181)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011DEC_S181)
,
SUBPARTITION P2011DEC_SMAX VALUES LESS THAN (MAXVALUE) TABLESPACE CM_D1T304_P2011DEC_SMAX
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011DEC_SMAX)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011DEC_SMAX)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011DEC_SMAX)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011DEC_SMAX)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011DEC_SMAX)
),
PARTITION "PMAX" VALUES LESS THAN (MAXVALUE)(
SUBPARTITION PMAX_S181 VALUES LESS THAN (181) TABLESPACE CM_D1T304_PMAX_S181
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_PMAX_S181)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_PMAX_S181)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_PMAX_S181)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_PMAX_S181)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_PMAX_S181)
,
SUBPARTITION PMAX_SMAX VALUES LESS THAN (MAXVALUE) TABLESPACE CM_D1T304_PMAX_SMAX
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_PMAX_SMAX)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_PMAX_SMAX)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_PMAX_SMAX)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_PMAX_SMAX)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_PMAX_SMAX)
)
);
```

INDEX

```
CREATE BIGFILE TABLESPACE CM_D1T304_IND DATAFILE '+DATA' SIZE 50M AUTOEXTEND ON MAXSIZE
UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;

CREATE UNIQUE INDEX D1T304P0 ON D1_INIT_MSRMT_DATA(INIT_MSRMT_DATA_ID) TABLESPACE
CM_D1T304_IND
GLOBAL PARTITION BY RANGE (INIT_MSRMT_DATA_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 D1_INIT_MSRMT_DATA ADD CONSTRAINT D1T304P0 PRIMARY KEY(INIT_MSRMT_DATA_ID)
USING INDEX;

CREATE INDEX D1T304S1 ON D1_INIT_MSRMT_DATA (MEASR_COMP_ID, BO_STATUS_CD, BUS_OBJ_CD,
D1_TO_DTTM, D1_FROM_DTTM) TABLESPACE CM_D1T304_IND
GLOBAL PARTITION BY RANGE (MEASR_COMP_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;


CREATE UNIQUE INDEX D1T304S2 ON D1_INIT_MSRMT_DATA(BUS_OBJ_CD, BO_STATUS_CD,
INIT_MSRMT_DATA_ID)
GLOBAL PARTITION BY HASH (BUS_OBJ_CD, BO_STATUS_CD,  INIT_MSRMT_DATA_ID)
(
 PARTITION P1 TABLESPACE CM_D1T304_IND,
 PARTITION P2 TABLESPACE CM_D1T304_IND,
 PARTITION P3 TABLESPACE CM_D1T304_IND,
 PARTITION P4 TABLESPACE CM_D1T304_IND,
 PARTITION P5 TABLESPACE CM_D1T304_IND,
 PARTITION P6 TABLESPACE CM_D1T304_IND,
 PARTITION P7 TABLESPACE CM_D1T304_IND,
 PARTITION P8 TABLESPACE CM_D1T304_IND
)
COMPRESS ADVANCED LOW;

CREATE UNIQUE INDEX D1T304S3 ON D1_INIT_MSRMT_DATA(IMD_EXT_ID,INIT_MSRMT_DATA_ID)
GLOBAL PARTITION BY HASH(IMD_EXT_ID)
(
 PARTITION P1 TABLESPACE CM_D1T304_IND,
 PARTITION P2 TABLESPACE CM_D1T304_IND,
 PARTITION P3 TABLESPACE CM_D1T304_IND,
 PARTITION P4 TABLESPACE CM_D1T304_IND,
 PARTITION P5 TABLESPACE CM_D1T304_IND,
 PARTITION P6 TABLESPACE CM_D1T304_IND,
 PARTITION P7 TABLESPACE CM_D1T304_IND,
 PARTITION P8 TABLESPACE CM_D1T304_IND
 )
COMPRESS ADVANCED LOW;

CREATE UNIQUE INDEX CM_ILM_D1T304S4 ON D1_INIT_MSRMT_DATA (ILM_DT, RETENTION_PERIOD,
ILM_ARCH_SW, INIT_MSRMT_DATA_ID) LOCAL COMPRESS ADVANCED LOW;
```

Child Table: D1_INIT_MSRMT_DATA_CHAR

```
CREATE TABLE D1_INIT_MSRMT_DATA_CHAR
  (
    INIT_MSRMT_DATA_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,
    LAST_UPDATE_DTTM DATE,
    CONSTRAINT D1_INIT_MSRMT_DATA_CHAR_FK FOREIGN KEY(INIT_MSRMT_DATA_ID) REFERENCES
D1_INIT_MSRMT_DATA
    )
PARTITION BY REFERENCE (D1_INIT_MSRMT_DATA_CHAR_FK)
ENABLE ROW MOVEMENT;
```

INDEX

```
CREATE UNIQUE INDEX D1T305P0 ON D1_INIT_MSRMT_DATA_CHAR(INIT_MSRMT_DATA_ID, CHAR_TYPE_CD,
SEQ_NUM) TABLESPACE CM_D1T304_IND
GLOBAL PARTITION BY RANGE(INIT_MSRMT_DATA_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 D1_INIT_MSRMT_DATA_CHAR ADD CONSTRAINT D1T305P0 PRIMARY KEY
(INIT_MSRMT_DATA_ID, CHAR_TYPE_CD, SEQ_NUM) USING INDEX;


CREATE INDEX D1T305S1 ON D1_INIT_MSRMT_DATA_CHAR(SRCH_CHAR_VAL)
```

```
GLOBAL PARTITION BY HASH(SRCH_CHAR_VAL)
 (
 PARTITION P1 TABLESPACE CM_D1T304_IND,
 PARTITION P2 TABLESPACE CM_D1T304_IND,
 PARTITION P3 TABLESPACE CM_D1T304_IND,
 PARTITION P4 TABLESPACE CM_D1T304_IND,
 PARTITION P5 TABLESPACE CM_D1T304_IND,
 PARTITION P6 TABLESPACE CM_D1T304_IND,
 PARTITION P7 TABLESPACE CM_D1T304_IND,
 PARTITION P8 TABLESPACE CM_D1T304_IND
 );
```

Child Table: D1_INIT_MSRMT_DATA_LOG

```
CREATE TABLE D1_INIT_MSRMT_DATA_LOG
  (
    INIT_MSRMT_DATA_ID  CHAR(14) NOT NULL ENABLE,
    SEQNO               NUMBER(5,0) NOT NULL ENABLE,
    BO_STATUS_CD        CHAR(12) DEFAULT ' ' NOT NULL ENABLE,
    BO_STATUS_REASON_CD VARCHAR2(30) DEFAULT ' ' 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,
    LOG_DTTM            DATE NOT NULL ENABLE,
    LOG_ENTRY_TYPE_FLG  CHAR(4) 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,
    USER_ID             CHAR(8) DEFAULT ' ' NOT NULL ENABLE,
    VERSION             NUMBER(5,0) DEFAULT 1 NOT NULL ENABLE,
    LAST_UPDATE_DTTM DATE,
    CONSTRAINT D1_INIT_MSRMT_DATA_LOG_FK FOREIGN KEY(INIT_MSRMT_DATA_ID) REFERENCES
D1_INIT_MSRMT_DATA
    )
PARTITION BY REFERENCE (D1_INIT_MSRMT_DATA_LOG_FK)
ENABLE ROW MOVEMENT;
```

INDEX

```
CREATE UNIQUE INDEX D1T306P0 ON D1_INIT_MSRMT_DATA_LOG (INIT_MSRMT_DATA_ID, SEQNO)
TABLESPACE CM_D1T304_IND
GLOBAL PARTITION BY RANGE(INIT_MSRMT_DATA_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 D1_INIT_MSRMT_DATA_LOG ADD CONSTRAINT D1T306P0 PRIMARY KEY
(INIT_MSRMT_DATA_ID, SEQNO) USING INDEX;
```

Child Table: D1_INIT_MSRMT_DATA_LOG_PARM

```
CREATE TABLE D1_INIT_MSRMT_DATA_LOG_PARM
  (
    INIT_MSRMT_DATA_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,
    LAST_UPDATE_DTTM DATE,
    CONSTRAINT D1_INIT_MSRMT_DATA_LOG_PARM_FK FOREIGN KEY(INIT_MSRMT_DATA_ID) REFERENCES
D1_INIT_MSRMT_DATA
```

```
    )
PARTITION BY REFERENCE (D1_INIT_MSRMT_DATA_LOG_PARM_FK)
ENABLE ROW MOVEMENT;
```

INDEX

```
CREATE UNIQUE INDEX D1T307P0 ON D1_INIT_MSRMT_DATA_LOG_PARM(INIT_MSRMT_DATA_ID, SEQNO,
PARM_SEQ) TABLESPACE CM_D1T304_IND
GLOBAL PARTITION BY RANGE(INIT_MSRMT_DATA_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 D1_INIT_MSRMT_DATA_LOG_PARM ADD CONSTRAINT D1T307P0 PRIMARY KEY
(INIT_MSRMT_DATA_ID, SEQNO, PARM_SEQ) USING INDEX;
```

Child Table: D1_INIT_MSRMT_DATA_K

```
CREATE BIGFILE TABLESPACE CM_D1T314_IND DATAFILE '+DATA' SIZE 50M AUTOEXTEND ON MAXSIZE
UNLIMITED;


CREATE TABLE D1_INIT_MSRMT_DATA_K
  (
    INIT_MSRMT_DATA_ID CHAR(14) NOT NULL ENABLE,
    ENV_ID             NUMBER(6,0) NOT NULL ENABLE,
    CONSTRAINT D1T314P0 PRIMARY KEY (INIT_MSRMT_DATA_ID, ENV_ID) ENABLE
  )
  ORGANIZATION INDEX
  Partition by range(INIT_MSRMT_DATA_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_D1T314_IND;
```

Appendix B

Sample SQL for Enabling ILM (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     NOT NULL ENABLE,
BATCH_CD        DEFAULT ' ' NOT NULL ENABLE,
BATCH_NBR       DEFAULT 0 NOT NULL ENABLE,
MESSAGE_CAT_NBR DEFAULT 0 NOT NULL ENABLE,
MESSAGE_NBR     DEFAULT 0 NOT NULL ENABLE,
ASSIGNED_TO     DEFAULT ' ' NOT NULL ENABLE,
TD_TYPE_CD      DEFAULT ' ' NOT NULL ENABLE,
```

```
    ROLE_ID         DEFAULT ' ' NOT NULL ENABLE,
    ENTRY_STATUS_FLG DEFAULT ' ' NOT NULL ENABLE,
    VERSION         DEFAULT 1 NOT NULL ENABLE,
    CRE_DTTM,
    ASSIGNED_DTTM,
    COMPLETE_DTTM,
    COMPLETE_USER_ID DEFAULT ' ' NOT NULL ENABLE,
    COMMENTS       DEFAULT ' ' NOT NULL ENABLE,
    ASSIGNED_USER_ID DEFAULT ' ' NOT NULL ENABLE,
    TD_PRIORITY_FLG DEFAULT ' ' NOT NULL ENABLE,
    ILM_DT DEFAULT SYSDATE,
    ILM_ARCH_SW DEFAULT 'N' NOT NULL ENABLE
    ) 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 "P2011JAN" VALUES LESS THAN (TO_DATE('2011-02-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
    TABLESPACE CM_XT039_P2011JAN,
    PARTITION "P2011FEB" VALUES LESS THAN (TO_DATE('2011-03-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
    TABLESPACE CM_XT039_P2011FEB,
    PARTITION "P2011MAR" VALUES LESS THAN (TO_DATE('2011-04-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
    TABLESPACE CM_XT039_P2011MAR,
    PARTITION "P2011APR" VALUES LESS THAN (TO_DATE('2011-05-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
    TABLESPACE CM_XT039_P2011APR,
    PARTITION "P2011MAY" VALUES LESS THAN (TO_DATE('2011-06-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
    TABLESPACE CM_XT039_P2011MAY,
    PARTITION "P2011JUN" VALUES LESS THAN (TO_DATE('2011-07-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
    TABLESPACE CM_XT039_P2011JUN,
    PARTITION "P2011JUL" VALUES LESS THAN (TO_DATE('2011-08-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
    TABLESPACE CM_XT039_P2011JUL,
    PARTITION "P2011AUG" VALUES LESS THAN (TO_DATE('2011-09-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
    TABLESPACE CM_XT039_P2011AUG,
    PARTITION "P2011SEP" VALUES LESS THAN (TO_DATE('2011-10-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
    TABLESPACE CM_XT039_P2011SEP,
    PARTITION "P2011OCT" VALUES LESS THAN (TO_DATE('2011-11-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
    TABLESPACE CM_XT039_P2011OCT,
    PARTITION "P2011NOV" VALUES LESS THAN (TO_DATE('2011-12-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
    TABLESPACE CM_XT039_P2011NOV,
    PARTITION "P2011DEC" VALUES LESS THAN (TO_DATE('2012-01-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
    TABLESPACE CM_XT039_P2011DEC,
    PARTITION "PMAX" VALUES LESS THAN (MAXVALUE)
    TABLESPACE CM_XT039_PMAX
    )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_ENRY.

```
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 (TD_ENTRY_ID)
NOLOGGING PARALLEL 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 ) 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
)
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 ( '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 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 Enabling ILM with Sub Retention (Existing Installation)

This section provides additional details including the sample syntax for each step using the Initial Measurement Data maintenance object as an example. Other maintenance object's implementations can follow a similar pattern.

1. Rename existing D1_INIT_MSRMT_DATA tables and primary key indexes and constraints as a backup. It is suggested to use an ILM_ prefix. The following are sample statements:

```
ALTER TABLE D1_INIT_MSRMT_DATA RENAME TO ILM_D1_INIT_MSRMT_DATA;

ALTER TABLE D1_INIT_MSRMT_DATA RENAME CONSTRAINT D1T304P0 TO ILM_D1T304P0;

ALTER INDEX D1T304P0 RENAME TO ILM_D1T304P0;

ALTER TABLE D1_INIT_MSRMT_DATA_CHAR RENAME TO ILM_D1_INIT_MSRMT_DATA_CHAR;

ALTER TABLE D1_INIT_MSRMT_DATA_CHAR RENAME CONSTRAINT D1T305P0 TO ILM_D1T305P0;

ALTER INDEX D1T305P0 RENAME TO ILM_D1T305P0;

ALTER TABLE D1_INIT_MSRMT_DATA_LOG RENAME TO ILM_D1_INIT_MSRMT_DATA_LOG;

ALTER TABLE D1_INIT_MSRMT_DATA_LOG RENAME CONSTRAINT D1T306P0 TO ILM_D1T306P0;

ALTER INDEX D1T306P0 RENAME TO ILM_D1T306P0;

ALTER TABLE D1_INIT_MSRMT_DATA_LOG_PARM RENAME TO ILM_D1_INIT_MSRMT_DATA_LOG_PARM;

ALTER TABLE D1_INIT_MSRMT_DATA_LOG_PARM RENAME CONSTRAINT D1T307P0 TO ILM_D1T307P0;

ALTER INDEX D1T307P0 RENAME TO ILM_D1T307P0;

 ALTER TABLE D1_INIT_MSRMT_DATA_K RENAME TO ILM_D1_INIT_MSRMT_DATA_K;

ALTER TABLE D1_INIT_MSRMT_DATA_K RENAME CONSTRAINT D1T314P0 TO ILM_D1T314P0;

ALTER INDEX D1T314P0 RENAME TO ILM_D1T314P0;
```

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','D1T304S1','CISADM') from dual;
select dbms_metadata.get_ddl('INDEX','D1T304S3','CISADM') from dual;
spool off;
```

3. Drop secondary indexes.

```
DROP INDEX CISADM.D1T304S1;
DROP INDEX CISADM.D1T304S3;
```

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. Use the CTAS queries listed in Chapter 5 to create temporary tables for ACTIVITY, DEVICE EVENT, and INITIAL MEASUREMENT DATA and use the following statements to create the partitioned tables.

Activity:

```
CREATE TABLE D1_ACTIVITY (
D1_ACTIVITY_ID      NOT NULL,
BUS_OBJ_CD          NOT NULL,
BO_STATUS_CD        NOT NULL,
ACTIVITY_TYPE_CD    NOT NULL,
START_DTTM          NOT NULL,
END_DTTM,
CRE_DTTM            NOT NULL,
STATUS_UPD_DTTM     NOT NULL,
BO_STATUS_REASON_CD NOT NULL,
VERSION             NOT NULL,
EFF_DTTM,
BO_DATA_AREA,
FIELD_TASK_TYPE,
CANCEL_REASON,
ILM_DT,
ILM_ARCH_SW,
RETENTION_PERIOD    NOT NULL
)
AS
SELECT
A.D1_ACTIVITY_ID,
A.BUS_OBJ_CD,
A.BO_STATUS_CD,
A.ACTIVITY_TYPE_CD,
A.START_DTTM,
A.END_DTTM,
A.CRE_DTTM,
A.STATUS_UPD_DTTM,
A.BO_STATUS_REASON_CD,
A.VERSION,
A.EFF_DTTM,
A.BO_DATA_AREA,
A.FIELD_TASK_TYPE,
A.CANCEL_REASON,
A.CRE_DTTM as ILM_DT,
'N' as ILM_ARCH_SW,
CAST(COALESCE((SELECT B.RETPERIOD
FROM ILM_ACTIVITY_RETENTION_TMP B
WHERE  B.ACTIVITY_TYPE_CD = A.ACTIVITY_TYPE_CD)
,CAST((select maint_obj_opt_val
from ci_md_mo_opt mmouni
where maint_obj_cd = 'D1-ACTIVITY'
and maint_obj_opt_flg ='FLRP'
and seq_num =
(select max(seq_num)
from ci_md_mo_opt mmo
where maint_obj_cd = 'D1-ACTIVITY'
and maint_obj_opt_flg ='FLRP')) as NUMBER(5))
,CAST((select extractvalue( xmlparse(content fw_mcfg.mst_config_data)
,'generalMasterConfiguration/defaultRetentionPeriod')
from f1_mst_config fw_mcfg
where fw_mcfg.bus_obj_cd = 'F1-ILMMSConfig') as NUMBER(5))
, 99999) as NUMBER(5)) as RETENTION_PERIOD
FROM ILM_D1_ACTIVITY A
/
```

Device Event:

```
CREATE TABLE D1_DVC_EVT(
DVC_EVT_ID          NOT NULL,
DVC_EVT_TYPE_CD,
BUS_OBJ_CD          NOT NULL,
EXT_EVT_NAME_FLG,
D1_SPR_CD,
BO_STATUS_CD        NOT NULL,
STATUS_UPD_DTTM     NOT NULL,
BO_STATUS_REASON_CD NOT NULL,
DVC_EVT_DTTM        NOT NULL,
```

```
CRE_DTTM            NOT NULL,
VERSION             NOT NULL,
DVC_EVT_END_DTTM,
BO_DATA_AREA,
D1_DEVICE_ID,
ILM_DT              NOT NULL,
ILM_ARCH_SW,
RETENTION_PERIOD    NOT NULL)
AS
SELECT
A.DVC_EVT_ID,
A.DVC_EVT_TYPE_CD,
A.BUS_OBJ_CD,
A.EXT_EVT_NAME_FLG,
A.D1_SPR_CD,
A.BO_STATUS_CD,
A.STATUS_UPD_DTTM,
A.BO_STATUS_REASON_CD,
A.DVC_EVT_DTTM,
A.CRE_DTTM,
A.VERSION,
A.DVC_EVT_END_DTTM,
A.BO_DATA_AREA,
A.D1_DEVICE_ID,
A.CRE_DTTM as ILM_DT,
'N' as ILM_ARCH_SW,
CAST(COALESCE((SELECT B.RETPERIOD
FROM ILM_DVC_EVT_RETENTION_TMP B
WHERE  B.DVC_EVT_TYPE_CD = A.DVC_EVT_TYPE_CD)
,CAST((select maint_obj_opt_val
from ci_md_mo_opt mmouni
where maint_obj_cd = 'D1-DVCEVENT'
and maint_obj_opt_flg ='FLRP'
and seq_num =
(select max(seq_num)
from ci_md_mo_opt mmo
where maint_obj_cd = 'D1-DVCEVENT'
and maint_obj_opt_flg ='FLRP')) as NUMBER(5))
,CAST((select extractvalue( xmlparse(content fw_mcfg.mst_config_data)
,'generalMasterConfiguration/defaultRetentionPeriod')
from f1_mst_config fw_mcfg
where fw_mcfg.bus_obj_cd = 'F1-ILMMSConfig') as NUMBER(5))
, 99999) as NUMBER(5)) as RETENTION_PERIOD
FROM ILM_D1_DVC_EVT A
/
```

Initial Measurement Data

```
CREATE TABLE ILM_IMD_RETENTION_TMP
AS
select mct.measr_comp_type_cd
/*retrieve the retention period for MC Types in this order of precedence:
1. The UOM based retention period from the MDM master configuration
2. The interval IMD retention period from the MDM master configuration
3. The MO level retention period from the MO options
4. The installation level retention period from the FW master configuration
*/
, CAST(coalesce( (select retPeriod
from (select 'D1IN' interval_scalar_flg
, extractvalue(value(p),'uomRetentionPeriodList/uom') D1_UOM_CD
, extractvalue(value(p),'uomRetentionPeriodList/retentionPeriod') retPeriod
from f1_mst_config mdm_mcfg
, table(xmlsequence(extract(xmlparse(content
mdm_mcfg.mst_config_data),
'imdRetentionPeriod/intervalImdRetentionPeriods/uomRetentionPeriods/
uomRetentionPeriodList'))) p
where mdm_mcfg.bus_obj_cd = 'D1-ILMMSConfig'
union
select 'D1SC' INTERVAL_SCALAR_FLG
, extractvalue(value(p),'uomRetentionPeriodList/uom') D1_UOM_CD
, extractvalue(value(p),'uomRetentionPeriodList/retentionPeriod') retPeriod
from f1_mst_config mdm_mcfg
, table(xmlsequence(extract(xmlparse(content
mdm_mcfg.mst_config_data),
'imdRetentionPeriod/scalarImdRetentionPeriods/uomRetentionPeriods/
uomRetentionPeriodList'))) p
where mdm_mcfg.bus_obj_cd = 'D1-ILMMSConfig') uomMap
where uomMap.interval_scalar_flg = mct.interval_scalar_flg
and trim(mctvi.d1_uom_cd) = trim(uomMap.d1_uom_cd))--UOM
, DECODE(mct.interval_scalar_flg
,'D1IN'
,extractvalue( xmlparse(content mdm_mcfg.mst_config_data),
'imdRetentionPeriod/intervalImdRetentionPeriods/intervalRetentionPeriod') --interval IMD
,extractvalue( xmlparse(content mdm_mcfg.mst_config_data),
```

```
'imdRetentionPeriod/scalarImdRetentionPeriods/scalarRetentionPeriod') --scalar IMD
)
, (select maint_obj_opt_val
from ci_md_mo_opt mmo
where maint_obj_cd = 'D1-IMD'
and maint_obj_opt_flg ='FLRP'
and seq_num = (select max(seq_num)
from ci_md_mo_opt mmo
where maint_obj_cd = 'D1-IMD'
and maint_obj_opt_flg ='FLRP')) --IMD
, extractvalue( xmlparse(content fw_mcfg.mst_config_data),
'generalMasterConfiguration/defaultRetentionPeriod') --Install
) as NUMBER(5)) retPeriod
from d1_measr_comp_type mct
, d1_mc_type_value_identifier mctvi
, f1_mst_config fw_mcfg
, f1_mst_config mdm_mcfg
where mct.measr_comp_type_cd = mctvi.measr_comp_type_cd
and mctvi.value_id_type_flg = 'D1MS'
and fw_mcfg.bus_obj_cd = 'F1-ILMMSConfig'
and mdm_mcfg.bus_obj_cd = 'D1-ILMMSConfig'
order by 1;


CREATE BIGFILE TABLESPACE CM_D1T304_P2011JAN_S181 DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011JAN_SMAX DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011FEB_S181 DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011FEB_SMAX DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011MAR_S181 DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011MAR_SMAX DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011APR_S181 DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011APR_SMAX DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011MAY_S181 DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011MAY_SMAX DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011JUN_S181 DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011JUN_SMAX DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011JUL_S181 DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011JUL_SMAX DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011AUG_S181 DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011AUG_SMAX DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011SEP_S181 DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011SEP_SMAX DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011OCT_S181 DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011OCT_SMAX DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011NOV_S181 DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011NOV_SMAX DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011DEC_S181 DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011DEC_SMAX DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_PMAX_S181 DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
CREATE BIGFILE TABLESPACE CM_D1T304_PMAX_SMAX DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;


CREATE TABLE D1_INIT_MSRMT_DATA
(
INIT_MSRMT_DATA_ID   NOT NULL,
MEASR_COMP_ID        NOT NULL,
D1_FROM_DTTM,
D1_TO_DTTM,
DATA_SRC_FLG         NOT NULL,
```

```
TIME_ZONE_CD         NOT NULL,
BUS_OBJ_CD           NOT NULL,
BO_STATUS_CD         NOT NULL,
BO_STATUS_REASON_CD  NOT NULL,
IMD_BO_DATA_AREA,
STATUS_UPD_DTTM      NOT NULL,
CRE_DTTM             NOT NULL,
VERSION              NOT NULL,
IMD_EXT_ID,
PREVEE_BO_DATA_AREA,
POSTVEE_BO_DATA_AREA,
TRACE_BO_DATA_AREA,
RAW_BO_DATA_AREA,
LAST_UPDATE_DTTM,
ILM_DT,
ILM_ARCH_SW,
RETENTION_PERIOD     NOT NULL
)
nologging  parallel (degree 10)
LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM CACHE)
LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM
CACHE)
LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM CACHE)
LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM CACHE)
LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM CACHE)
PARTITION BY RANGE (ILM_DT) SUBPARTITION BY RANGE (RETENTION_PERIOD)
(
PARTITION "P2011JAN" VALUES LESS THAN (TO_DATE('2011-02-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))(
SUBPARTITION P2011JAN_S181 VALUES LESS THAN (181) TABLESPACE CM_D1T304_P2011JAN_S181
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JAN_S181)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JAN_S181)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JAN_S181)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JAN_S181)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JAN_S181)
,
SUBPARTITION P2011JAN_SMAX VALUES LESS THAN (MAXVALUE) TABLESPACE CM_D1T304_P2011JAN_SMAX
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JAN_SMAX)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JAN_SMAX)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JAN_SMAX)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JAN_SMAX)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JAN_SMAX)
),
PARTITION "P2011FEB" VALUES LESS THAN (TO_DATE('2011-03-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))(
SUBPARTITION P2011FEB_S181 VALUES LESS THAN (181) TABLESPACE CM_D1T304_P2011FEB_S181
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011FEB_S181)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011FEB_S181)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011FEB_S181)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011FEB_S181)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011FEB_S181)
,
SUBPARTITION P2011FEB_SMAX VALUES LESS THAN (MAXVALUE) TABLESPACE CM_D1T304_P2011FEB_SMAX
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011FEB_SMAX)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011FEB_SMAX)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011FEB_SMAX)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011FEB_SMAX)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011FEB_SMAX)
),
PARTITION "P2011MAR" VALUES LESS THAN (TO_DATE('2011-04-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))(
SUBPARTITION P2011MAR_S181 VALUES LESS THAN (181) TABLESPACE CM_D1T304_P2011MAR_S181
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAR_S181)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAR_S181)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAR_S181)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAR_S181)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAR_S181)
,
SUBPARTITION P2011MAR_SMAX VALUES LESS THAN (MAXVALUE) TABLESPACE CM_D1T304_P2011MAR_SMAX
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAR_SMAX)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAR_SMAX)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAR_SMAX)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAR_SMAX)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAR_SMAX)
),
PARTITION "P2011APR" VALUES LESS THAN (TO_DATE('2011-05-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))(
SUBPARTITION P2011APR_S181 VALUES LESS THAN (181) TABLESPACE CM_D1T304_P2011APR_S181
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011APR_S181)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011APR_S181)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011APR_S181)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011APR_S181)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011APR_S181)
,
SUBPARTITION P2011APR_SMAX VALUES LESS THAN (MAXVALUE) TABLESPACE CM_D1T304_P2011APR_SMAX
```

```
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011APR_SMAX)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011APR_SMAX)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011APR_SMAX)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011APR_SMAX)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011APR_SMAX)
),
PARTITION "P2011MAY" VALUES LESS THAN (TO_DATE('2011-06-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))(
SUBPARTITION P2011MAY_S181 VALUES LESS THAN (181) TABLESPACE CM_D1T304_P2011MAY_S181
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAY_S181)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAY_S181)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAY_S181)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAY_S181)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAY_S181)
,
SUBPARTITION P2011MAY_SMAX VALUES LESS THAN (MAXVALUE) TABLESPACE CM_D1T304_P2011MAY_SMAX
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAY_SMAX)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAY_SMAX)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAY_SMAX)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAY_SMAX)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAY_SMAX)
),
PARTITION "P2011JUN" VALUES LESS THAN (TO_DATE('2011-07-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))(
SUBPARTITION P2011JUN_S181 VALUES LESS THAN (181) TABLESPACE CM_D1T304_P2011JUN_S181
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUN_S181)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUN_S181)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUN_S181)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUN_S181)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUN_S181)
,
SUBPARTITION P2011JUN_SMAX VALUES LESS THAN (MAXVALUE) TABLESPACE CM_D1T304_P2011JUN_SMAX
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUN_SMAX)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUN_SMAX)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUN_SMAX)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUN_SMAX)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUN_SMAX)
),
PARTITION "P2011JUL" VALUES LESS THAN (TO_DATE('2011-08-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))(
SUBPARTITION P2011JUL_S181 VALUES LESS THAN (181) TABLESPACE CM_D1T304_P2011JUL_S181
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUL_S181)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUL_S181)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUL_S181)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUL_S181)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUL_S181)
,
SUBPARTITION P2011JUL_SMAX VALUES LESS THAN (MAXVALUE) TABLESPACE CM_D1T304_P2011JUL_SMAX
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUL_SMAX)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUL_SMAX)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUL_SMAX)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUL_SMAX)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUL_SMAX)
),
PARTITION "P2011AUG" VALUES LESS THAN (TO_DATE('2011-09-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))(
SUBPARTITION P2011AUG_S181 VALUES LESS THAN (181) TABLESPACE CM_D1T304_P2011AUG_S181
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011AUG_S181)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011AUG_S181)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011AUG_S181)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011AUG_S181)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011AUG_S181)
,
SUBPARTITION P2011AUG_SMAX VALUES LESS THAN (MAXVALUE) TABLESPACE CM_D1T304_P2011AUG_SMAX
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011AUG_SMAX)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011AUG_SMAX)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011AUG_SMAX)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011AUG_SMAX)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011AUG_SMAX)
),
PARTITION "P2011SEP" VALUES LESS THAN (TO_DATE('2011-10-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))(
SUBPARTITION P2011SEP_S181 VALUES LESS THAN (181) TABLESPACE CM_D1T304_P2011SEP_S181
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011SEP_S181)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011SEP_S181)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011SEP_S181)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011SEP_S181)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011SEP_S181)
,
SUBPARTITION P2011SEP_SMAX VALUES LESS THAN (MAXVALUE) TABLESPACE CM_D1T304_P2011SEP_SMAX
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011SEP_SMAX)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011SEP_SMAX)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011SEP_SMAX)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011SEP_SMAX)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011SEP_SMAX)
```

```
),
PARTITION "P2011OCT" VALUES LESS THAN (TO_DATE('2011-11-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))(
SUBPARTITION P2011OCT_S181 VALUES LESS THAN (181) TABLESPACE CM_D1T304_P2011OCT_S181
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011OCT_S181)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011OCT_S181)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011OCT_S181)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011OCT_S181)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011OCT_S181)
,
SUBPARTITION P2011OCT_SMAX VALUES LESS THAN (MAXVALUE) TABLESPACE CM_D1T304_P2011OCT_SMAX
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011OCT_SMAX)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011OCT_SMAX)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011OCT_SMAX)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011OCT_SMAX)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011OCT_SMAX)
),
PARTITION "P2011NOV" VALUES LESS THAN (TO_DATE('2011-12-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))(
SUBPARTITION P2011NOV_S181 VALUES LESS THAN (181) TABLESPACE CM_D1T304_P2011NOV_S181
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011NOV_S181)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011NOV_S181)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011NOV_S181)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011NOV_S181)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011NOV_S181)
,
SUBPARTITION P2011NOV_SMAX VALUES LESS THAN (MAXVALUE) TABLESPACE CM_D1T304_P2011NOV_SMAX
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011NOV_SMAX)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011NOV_SMAX)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011NOV_SMAX)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011NOV_SMAX)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011NOV_SMAX)
),
PARTITION "P2011DEC" VALUES LESS THAN (TO_DATE('2012-01-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))(
SUBPARTITION P2011DEC_S181 VALUES LESS THAN (181) TABLESPACE CM_D1T304_P2011DEC_S181
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011DEC_S181)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011DEC_S181)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011DEC_S181)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011DEC_S181)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011DEC_S181)
,
SUBPARTITION P2011DEC_SMAX VALUES LESS THAN (MAXVALUE) TABLESPACE CM_D1T304_P2011DEC_SMAX
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011DEC_SMAX)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011DEC_SMAX)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011DEC_SMAX)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011DEC_SMAX)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011DEC_SMAX)
),
PARTITION "PMAX" VALUES LESS THAN (MAXVALUE)(
SUBPARTITION PMAX_S181 VALUES LESS THAN (181) TABLESPACE CM_D1T304_PMAX_S181
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_PMAX_S181)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_PMAX_S181)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_PMAX_S181)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_PMAX_S181)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_PMAX_S181)
,
SUBPARTITION PMAX_SMAX VALUES LESS THAN (MAXVALUE) TABLESPACE CM_D1T304_PMAX_SMAX
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_PMAX_SMAX)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_PMAX_SMAX)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_PMAX_SMAX)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_PMAX_SMAX)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_PMAX_SMAX)
)) ENABLE ROW MOVEMENT AS
SELECT
A.INIT_MSRMT_DATA_ID,
A.MEASR_COMP_ID,
A.D1_FROM_DTTM,
A.D1_TO_DTTM,
A.DATA_SRC_FLG,
A.TIME_ZONE_CD,
A.BUS_OBJ_CD,
A.BO_STATUS_CD,
A.BO_STATUS_REASON_CD,
A.IMD_BO_DATA_AREA,
A.STATUS_UPD_DTTM,
A.CRE_DTTM,
A.VERSION,
A.IMD_EXT_ID,
A.PREVEE_BO_DATA_AREA,
A.POSTVEE_BO_DATA_AREA,
A.TRACE_BO_DATA_AREA,
A.RAW_BO_DATA_AREA,
A.LAST_UPDATE_DTTM,
A.CRE_DTTM as ILM_DT,
```

```
'N' as ILM_ARCH_SW,
CAST(COALESCE((SELECT C.RETPERIOD
FROM D1_MEASR_COMP B, ILM_IMD_RETENTION_TMP C
WHERE B.MEASR_COMP_ID = A.MEASR_COMP_ID
AND C.MEASR_COMP_TYPE_CD = B.MEASR_COMP_TYPE_CD)
,CAST((select maint_obj_opt_val
from ci_md_mo_opt mmo
where maint_obj_cd = 'D1-IMD'
and maint_obj_opt_flg ='FLRP'
and seq_num =
(select max(seq_num)
from ci_md_mo_opt mmo
where maint_obj_cd = 'D1-IMD'
and maint_obj_opt_flg ='FLRP')) as NUMBER(5))
,CAST((select extractvalue( xmlparse(content fw_mcfg.mst_config_data)
,'generalMasterConfiguration/defaultRetentionPeriod')
from f1_mst_config fw_mcfg
where fw_mcfg.bus_obj_cd = 'F1-ILMMSConfig') as NUMBER(5))
, 99999) as NUMBER(5)) as RETENTION_PERIOD
FROM ILM_D1_INIT_MSRMT_DATA A
/
```

5. Enable logging option for table D1_INIT_MSRMT_DATA.

```
ALTER TABLE D1_INIT_MSRMT_DATA NOPARALLEL LOGGING;
```

6. Create Primary Index for Parent table D1_INIT_MSRMT_DATA.

```
CREATE BIGFILE TABLESPACE CM_D1T304_IND DATAFILE '+DATA' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;

CREATE UNIQUE INDEX D1T304P0 ON D1_INIT_MSRMT_DATA NOLOGGING PARALLEL
(INIT_MSRMT_DATA_ID)
GLOBAL PARTITION BY RANGE (INIT_MSRMT_DATA_ID) (
PARTITION P1 VALUES LESS THAN ('124999999999999'),
 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 INDEX D1T304P0 LOGGING NOPARALLEL;
```

7. Add Primary Key for Parent table D1_INIT_MSRMT_DATA

```
ALTER TABLE D1_INIT_MSRMT_DATA ADD CONSTRAINT D1T304P0 PRIMARY KEY(INIT_MSRMT_DATA_ID)
USING INDEX
/
```

8. Create Secondary Indexes for Parent table D1_INIT_MSRMT_DATA

```
CREATE INDEX D1T304S1 ON D1_INIT_MSRMT_DATA (MEASR_COMP_ID, BO_STATUS_CD, BUS_OBJ_CD,
D1_TO_DTTM, D1_FROM_DTTM)
GLOBAL PARTITION BY RANGE (MEASR_COMP_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
/

CREATE UNIQUE INDEX D1T304S3 ON D1_INIT_MSRMT_DATA(IMD_EXT_ID,INIT_MSRMT_DATA_ID)
GLOBAL PARTITION BY HASH(IMD_EXT_ID) (
PARTITION P1 TABLESPACE CM_D1T304_IND,
PARTITION P2 TABLESPACE CM_D1T304_IND,
PARTITION P3 TABLESPACE CM_D1T304_IND,
PARTITION P4 TABLESPACE CM_D1T304_IND,
PARTITION P5 TABLESPACE CM_D1T304_IND,
PARTITION P6 TABLESPACE CM_D1T304_IND,
PARTITION P7 TABLESPACE CM_D1T304_IND,
PARTITION P8 TABLESPACE CM_D1T304_IND
) COMPRESS ADVANCED LOW
/
```

```
CREATE UNIQUE INDEX CM_ILM_D1T304S4 ON D1_INIT_MSRMT_DATA (ILM_DT, RETENTION_PERIOD,
ILM_ARCH_SW, INIT_MSRMT_DATA_ID) LOCAL COMPRESS ADVANCED LOW
/
```

9. Create Child Tables, Primary Key, Primary Indexes and Secondary Indexes as shown below.

Create Child Table D1_INIT_MSRMT_DATA_CHAR

```
CREATE TABLE D1_INIT_MSRMT_DATA_CHAR
(
INIT_MSRMT_DATA_ID  NOT NULL ENABLE,
CHAR_TYPE_CD  NOT NULL ENABLE,
SEQ_NUM  NOT NULL ENABLE,
CHAR_VAL  DEFAULT ' ' NOT NULL ENABLE,
ADHOC_CHAR_VAL  DEFAULT ' ' NOT NULL ENABLE,
CHAR_VAL_FK1  DEFAULT ' ' NOT NULL ENABLE,
CHAR_VAL_FK2  DEFAULT ' ' NOT NULL ENABLE,
CHAR_VAL_FK3  DEFAULT ' ' NOT NULL ENABLE,
CHAR_VAL_FK4  DEFAULT ' ' NOT NULL ENABLE,
CHAR_VAL_FK5  DEFAULT ' ' NOT NULL ENABLE,
SRCH_CHAR_VAL DEFAULT ' ' NOT NULL ENABLE,
VERSION  DEFAULT 1 NOT NULL ENABLE,
LAST_UPDATE_DTTM ,
CONSTRAINT D1_INIT_MSRMT_DATA_CHAR_FK FOREIGN KEY(INIT_MSRMT_DATA_ID) REFERENCES
D1_INIT_MSRMT_DATA
)
PARTITION BY REFERENCE (D1_INIT_MSRMT_DATA_CHAR_FK) ENABLE ROW MOVEMENT NOLOGGING
PARALLEL
AS SELECT /*+ PARALLEL */ * FROM ILM_D1_INIT_MSRMT_DATA_CHAR
/

ALTER TABLE D1_INIT_MSRMT_DATA_CHAR LOGGING NOPARALLEL
/
```

Create Primary Index for Child Table D1_INIT_MSRMT_DATA_CHAR

```
CREATE UNIQUE INDEX D1T305P0 ON D1_INIT_MSRMT_DATA_CHAR(INIT_MSRMT_DATA_ID,
CHAR_TYPE_CD, SEQ_NUM)
TABLESPACE CM_D1T304_IND NOLOGGING PARALLEL
GLOBAL PARTITION BY RANGE(INIT_MSRMT_DATA_ID) (
PARTITION P1 VALUES LESS THAN ('124999999999999'),
PARTITION P2 VALUES LESS THAN ('249999999999999'),
PARTITION P3 VALUES LESS THAN ('374999999999999'),
PARTITION P4 VALUES LESS THAN ('499999999999999'),
PARTITION P5 VALUES LESS THAN ('624999999999999'),
PARTITION P6 VALUES LESS THAN ('749999999999999'),
PARTITION P7 VALUES LESS THAN ('874999999999999'),
PARTITION P8 VALUES LESS THAN (MAXVALUE)
) COMPRESS ADVANCED LOW
/

ALTER INDEX D1T305P0 LOGGING NOPARALLEL
/
```

Create Primary Key for Child Table D1_INIT_MSRMT_DATA_CHAR

```
ALTER TABLE D1_INIT_MSRMT_DATA_CHAR ADD CONSTRAINT D1T305P0 PRIMARY KEY
(INIT_MSRMT_DATA_ID, CHAR_TYPE_CD, SEQ_NUM) USING INDEX
/
```

Create Secondary Indexes for Child Table D1_INIT_MSRMT_DATA_CHAR

```
CREATE INDEX D1T305S1 ON D1_INIT_MSRMT_DATA_CHAR(SRCH_CHAR_VAL) GLOBAL PARTITION BY
HASH(SRCH_CHAR_VAL)
(
PARTITION P1 TABLESPACE CM_D1T304_IND,
 PARTITION P2 TABLESPACE CM_D1T304_IND,
 PARTITION P3 TABLESPACE CM_D1T304_IND,
 PARTITION P4 TABLESPACE CM_D1T304_IND,
 PARTITION P5 TABLESPACE CM_D1T304_IND,
 PARTITION P6 TABLESPACE CM_D1T304_IND,
 PARTITION P7 TABLESPACE CM_D1T304_IND,
 PARTITION P8 TABLESPACE CM_D1T304_IND
)
/
```

Create Child Table D1_INIT_MSRMT_DATA_LOG

```
CREATE TABLE D1_INIT_MSRMT_DATA_LOG (

INIT_MSRMT_DATA_ID   NOT NULL ENABLE,
    SEQNO                NOT NULL ENABLE,
    BO_STATUS_CD         DEFAULT ' ' NOT NULL ENABLE,
    BO_STATUS_REASON_CD  DEFAULT ' ' NOT NULL ENABLE,
    CHAR_TYPE_CD         DEFAULT ' ' NOT NULL ENABLE,
    CHAR_VAL             DEFAULT ' ' NOT NULL ENABLE,
    ADHOC_CHAR_VAL       DEFAULT ' ' NOT NULL ENABLE,
    CHAR_VAL_FK1         DEFAULT ' ' NOT NULL ENABLE,
```

```
    CHAR_VAL_FK2         DEFAULT ' ' NOT NULL ENABLE,
    CHAR_VAL_FK3         DEFAULT ' ' NOT NULL ENABLE,
    CHAR_VAL_FK4         DEFAULT ' ' NOT NULL ENABLE,
    CHAR_VAL_FK5         DEFAULT ' ' NOT NULL ENABLE,
    DESCRLONG            DEFAULT ' ' NOT NULL ENABLE,
    LOG_DTTM             NOT NULL ENABLE,
    LOG_ENTRY_TYPE_FLG   DEFAULT ' ' NOT NULL ENABLE,
    MESSAGE_CAT_NBR      DEFAULT 0 NOT NULL ENABLE,
    MESSAGE_NBR          DEFAULT 0 NOT NULL ENABLE,
    USER_ID              DEFAULT ' ' NOT NULL ENABLE,
    VERSION              DEFAULT 1 NOT NULL ENABLE,
    LAST_UPDATE_DTTM,
CONSTRAINT D1_INIT_MSRMT_DATA_LOG_FK FOREIGN KEY(INIT_MSRMT_DATA_ID) REFERENCES
D1_INIT_MSRMT_DATA
)
PARTITION BY REFERENCE (D1_INIT_MSRMT_DATA_LOG_FK) ENABLE ROW MOVEMENT NOLOGGING
PARALLEL
AS SELECT /*+ PARALLEL */ * FROM ILM_D1_INIT_MSRMT_DATA_LOG
/

ALTER TABLE D1_INIT_MSRMT_DATA_LOG LOGGING NOPARALLEL
/
```

Create Primary Index for Child Table D1_INIT_MSRMT_DATA_LOG

```
CREATE UNIQUE INDEX D1T306P0 ON D1_INIT_MSRMT_DATA_LOG(INIT_MSRMT_DATA_ID, SEQNO)
TABLESPACE CM_D1T304_IND NOLOGGING PARALLEL
GLOBAL PARTITION BY RANGE(INIT_MSRMT_DATA_ID) (
PARTITION P1 VALUES LESS THAN ('124999999999999'),
PARTITION P2 VALUES LESS THAN ('249999999999999'),
PARTITION P3 VALUES LESS THAN ('374999999999999'),
PARTITION P4 VALUES LESS THAN ('499999999999999'),
PARTITION P5 VALUES LESS THAN ('624999999999999'),
PARTITION P6 VALUES LESS THAN ('749999999999999'),
PARTITION P7 VALUES LESS THAN ('874999999999999'),
PARTITION P8 VALUES LESS THAN (MAXVALUE)
) COMPRESS ADVANCED LOW
/

ALTER INDEX D1T306P0 LOGGING NOPARALLEL
/
```

Create Primary Key for Child Table D1_INIT_MSRMT_DATA_LOG

```
ALTER TABLE D1_INIT_MSRMT_DATA_LOG ADD CONSTRAINT D1T306P0 PRIMARY KEY
(INIT_MSRMT_DATA_ID, SEQNO) USING INDEX
/
```

Create Child Table D1_INIT_MSRMT_DATA_LOG_PARM

```
CREATE TABLE D1_INIT_MSRMT_DATA_LOG_PARM (
INIT_MSRMT_DATA_ID  NOT NULL ENABLE,
SEQNO  NOT NULL ENABLE,
PARM_SEQ  NOT NULL ENABLE,
MSG_PARM_VAL DEFAULT ' ' NOT NULL ENABLE,
MSG_PARM_TYP_FLG  DEFAULT ' ' NOT NULL ENABLE,
VERSION  DEFAULT 1 NOT NULL ENABLE,
LAST_UPDATE_DTTM ,
CONSTRAINT D1_INIT_MSRMT_DATA_LOG_PARM_FK FOREIGN KEY(INIT_MSRMT_DATA_ID) REFERENCE
D1_INIT_MSRMT_DATA
)
PARTITION BY REFERENCE (D1_INIT_MSRMT_DATA_LOG_PARM_FK) ENABLE ROW MOVEMENT NOLOGGING
PARALLEL
AS SELECT /*+ PARALLEL */ * FROM ILM_D1_INIT_MSRMT_DATA_LOG_PARM
/

ALTER TABLE D1_INIT_MSRMT_DATA_LOG_PARM LOGGING NOPARALLEL
/
```

Create Primary Index for Child Table D1_INIT_MSRMT_DATA_LOG_PARM

```
CREATE UNIQUE INDEX D1T307P0 ON D1_INIT_MSRMT_DATA_LOG_PARM(INIT_MSRMT_DATA_ID, SEQNO,
PARM_SEQ)
TABLESPACE CM_D1T304_IND NOLOGGING PARALLEL GLOBAL PARTITION BY
RANGE(INIT_MSRMT_DATA_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 INDEX D1T306P0 LOGGING NOPARALLEL
/
```

Create Primary Key for Child Table D1_INIT_MSRMT_DATA_LOG_PARM

```
ALTER TABLE D1_INIT_MSRMT_DATA_LOG ADD CONSTRAINT D1T307P0 PRIMARY KEY
(INIT_MSRMT_DATA_ID, SEQNO, PARM_SEQ) USING INDEX
/
```

Create Child Table D1_INIT_MSRMT_DATA_K

```
CREATE BIGFILE TABLESPACE CM_D1T314_IND DATAFILE '+DATA' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED;

CREATE TABLE D1_INIT_MSRMT_DATA_K (
INIT_MSRMT_DATA_ID NOT NULL ENABLE,
ENV_ID             NOT NULL ENABLE,
CONSTRAINT D1T314P0 PRIMARY KEY (INIT_MSRMT_DATA_ID, ENV_ID) ENABLE
)
ORGANIZATION INDEX
Partition by range(INIT_MSRMT_DATA_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_D1T314_IND
AS SELECT /*+ PARALLEL */ * FROM ILM_D1_INIT_MSRMT_DATA_K
/

ALTER TABLE D1_INIT_MSRMT_DATA_K LOGGING NOPARALLEL
/
```

10. After verification of the ILM based tables, the user can drop the backup “ILM” renamed tables.

Appendix D

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, Inbound Sync Request and Initial Measurement Data maintenance objects are used as examples. This section contains the following steps:

- Add Partition
- Archive Partition
- Archive Subpartition
- Restore Partition
- Restore SubPartition
- Compress Partition (D1_MSRMT table only)

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_P2011JAN READ ONLY;
   ```

2. Check the feasibility of archive using ILM_ARCH_SW = 'N'.

   ```
   Select count(1) from CISADM.CI_TD_ENTRY PARTITION P2011JAN where ILM_ARCH_SW = 'N';
   ```

   - IF the above query has a count of greater than ZERO records - Change the tablespace back to read and write mode. Archive cannot be done. Do not execute further steps. Stop archiving partition.

     ```
     ALTER TABLESPACE CM_XT039_P2011JAN READ WRITE;
     ```

   - IF above query has ZERO records - Archive can be performed. Continue executing the remainder of the procedure.

3. Create separate archive tablespace for the partition that needs to be archived.

   ```
   CREATE BIGFILE TABLESPACE CM_XT039_P2011JAN_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_P2011JAN_ARC PARALLEL NOLOGGING TABLESPACE
   CM_XT039_P2011JAN_ARC
   AS
   (
   SELECT /*+ PARALLEL */ * FROM CISADM.CI_TD_ENTRY_CHA PARTITION (P2011JAN_S01)
   UNION ALL
   SELECT /*+ PARALLEL */ * FROM CI_TD_ENTRY_CHA PARTITION (P2011JAN_S02)
   UNION ALL
   .
   .
   .
   .
   UNION ALL
   SELECT /*+ PARALLEL */ * FROM CI_TD_ENTRY_CHA PARTITION (P2011JAN_S08)
   );
   ALTER TABLE CM_XT701_P2011JAN_ARC NOPARALLEL LOGGING;
   ```

   b. **CI_TD_MSG_PARM**

   ```
   CREATE TABLE CM_XT04_P2011JAN_ARC PARALLEL NOLOGGING TABLESPACE
   CM_XT039_P2011JAN_ARC
   AS
   (
   SELECT /*+ PARALLEL */ * FROM CISADM. CI_TD_MSG_PARM PARTITION (P2011JAN_S01)
   UNION ALL
   SELECT /*+ PARALLEL */ * FROM CI_TD_MSG_PARM  PARTITION (P2011JAN_S02)
   UNION ALL
   .
   .
   .
   .
   UNION ALL
   SELECT /*+ PARALLEL */ * FROM CI_TD_MSG_PARM  PARTITION (P2011JAN_S08)
   );
   ALTER TABLE CM_XT04_P2011JAN_ARC NOPARALLEL LOGGING;
   ```

   c. CI_TD_LOG

   ```
   CREATE TABLE CM_XT721_P2011JAN_ARC PARALLEL NOLOGGING TABLESPACE
   CM_XT039_P2011JAN_ARC
   AS
   (
   SELECT /*+ PARALLEL */ * FROM CISADM.CI_TD_LOG PARTITION (P2011JAN_S01)
   ```

```
UNION ALL
SELECT /*+ PARALLEL */ * FROM CI_TD_LOG PARTITION (P2011JAN_S02)
UNION ALL
.
.
.
.
UNION ALL
SELECT /*+ PARALLEL */ * FROM CI_TD_LOG PARTITION (P2011JAN_S08)
);
ALTER TABLE CM_XT721_P2011JAN_ARC NOPARALLEL LOGGING;
```

d. CI_TD_SRTKEY

```
CREATE TABLE CM_XT041_P2011JAN_ARC PARALLEL NOLOGGING TABLESPACE
CM_XT039_P2011JAN_ARC
AS
(
SELECT /*+ PARALLEL */ * FROM CISADM.CI_TD_SRTKEY PARTITION (P2011JAN_S01)
UNION ALL
SELECT /*+ PARALLEL */ * FROM CI_TD_SRTKEY PARTITION (P2011JAN_S02)
UNION ALL
.
.
.
.
UNION ALL
SELECT /*+ PARALLEL */ * FROM CI_TD_SRTKEY PARTITION (P2011JAN_S08)
);
ALTER TABLE CM_XT041_P2011JAN_ARC NOPARALLEL LOGGING;
```

e. CI_TD_DRLKEY

```
CREATE TABLE CM_XT037_P2011JAN_ARC PARALLEL NOLOGGING TABLESPACE
CM_XT039_P2011JAN_ARC
AS
(
SELECT /*+ PARALLEL */ * FROM CISADM.CI_TD_DRLKEY PARTITION (P2011JAN_S01)
UNION ALL
SELECT /*+ PARALLEL */ * FROM CISADM.CI_TD_DRLKEY PARTITION (P2011JAN_S02)
UNION ALL
.
.
.
.
UNION ALL
SELECT /*+ PARALLEL */ * FROM CISADM.CI_TD_DRLKEY PARTITION (P2011JAN_S08)
);
ALTER TABLE CM_XT037_P2011JAN_ARC NOPARALLEL LOGGING;
```

5. Create staging table and load data for parent table.

```
CREATE TABLE CM_XT039_P2011JAN_ARC NOLOGGING PARALLEL TABLESPACE CM_XT039_P2011JAN_ARC
AS
SELECT /*+ PARALLEL */ * FROM CISADM.CI_TD_ENTRY PARTITION (P2011JAN);

ALTER TABLE CM_XT039_P2011JAN_ARC NOPARALLEL LOGGING;
```

6. Export tablespace using TRANSPORT_TABLESPACES method.

```
ALTER TABLESPACE CM_XT039_P2011JAN_ARC READ ONLY;

expdp system/manager DIRECTORY=DUMP_DIR DUMPFILE= CM_XT039_P2011JAN_ARC.DMP
TRANSPORT_TABLESPACES = CM_XT039_P2011JAN_ARC LOGFILE=EXP_ CM_XT039_P2011JAN_ARC.LOG
TRANSPORT_FULL_CHECK=Y
```

- Make sure 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_p201101_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 P2011JAN UPDATE INDEXES;
DROP TABLESPACE CM_XT039_P2011JAN INCLUDING CONTENTS AND DATAFILES;
DROP TABLESPACE CM_XT039_P2011JAN_ARC INCLUDING CONTENTS AND DATAFILES;
```

Archive Subpartition

1. Make the tablespace to be archived READ ONLY.

```
ALTER TABLESPACE CM_D1T304_P2011JAN_S181 READ ONLY;
```

2. Check the feasibility of archive using ILM_ARCH_SW = 'N'.

```
Select count(1) from cisadm.D1_INIT_MSRMT_DATA SUBPARTITION P2011JAN_S181 where
ILM_ARCH_SW = 'N';
```

   - IF the above query has a count of greater than ZERO records - Change the tablespace back to read and write mode. Archive cannot be done. Do not execute further steps. Stop archiving partition.

     ```
     ALTER TABLESPACE CM_D1T304_P2011JAN_S181 READ WRITE;
     ```

   - IF above query has ZERO records - Archive can be performed. Continue executing the remainder of the procedure.

3. Create separate archive tablespace for partition that needs to be archived.

```
CREATE BIGFILE TABLESPACE CM_D1T304_P2011JAN_S181_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.

```
CREATE TABLE CM_D1T305_P2011JAN_S181_ARC PARALLEL NOLOGGING TABLESPACE
CM_D1T304_P2011JAN_S181_ARC
AS
(
SELECT /*+ PARALLEL */ * FROM CISADM.D1_INIT_MSRMT_DATA_CHAR PARTITION (P2011JAN_S181)
);

CREATE TABLE CM_D1T306_P2011JAN_S181_ARC PARALLEL NOLOGGING TABLESPACE
CM_D1T304_P2011JAN_S181_ARC
AS
(
SELECT /*+ PARALLEL */ * FROM CISADM.D1_INIT_MSRMT_DATA_LOG PARTITION (P2011JAN_S181)
);

CREATE TABLE CM_D1T307_P2011JAN_S181_ARC PARALLEL NOLOGGING TABLESPACE
CM_D1T304_P2011JAN_S181_ARC
AS
(
SELECT /*+ PARALLEL */ * FROM CISADM.D1_INIT_MSRMT_DATA_LOG_PARM PARTITION
(P2011JAN_S181)
);

ALTER TABLE CM_D1T305_P2011JAN_S181_ARC NOPARALLEL LOGGING;

ALTER TABLE CM_D1T306_P2011JAN_S181_ARC NOPARALLEL LOGGING;

ALTER TABLE CM_D1T307_P2011JAN_S181_ARC NOPARALLEL LOGGING;
```

5. Create staging table and load data for parent table

```
CREATE TABLE ALTER TABLE CM_D1T304_P2011JAN_S181_ARC NOPARALLEL LOGGING; NOLOGGING
PARALLEL TABLESPACE CM_D1T304_P2011JAN_S181_ARC
AS
SELECT /*+ PARALLEL */ * FROM D1_INIT_MSRMT_DATA SUBPARTITION (P2011JAN_S181);

ALTER TABLE CM_D1T304_P2011JAN_S181_ARC NOPARALLEL LOGGING;
```

6. Export tablespace using TRANSPORT_TABLESPACES method.

```
ALTER TABLESPACE CM_D1T304_P2011JAN_S181_ARC READ ONLY;
expdp system/manager DIRECTORY=DUMP_DIR DUMPFILE=CM_D1T304_P2011JAN_S181_ARC.DMP
TRANSPORT_TABLESPACES=CM_D1T304_P2011JAN_S181_ARC
LOGFILE=EXP_CM_D1T304_P2011JAN_S181_ARC.LOG TRANSPORT_FULL_CHECK=Y

i.Make sure the tablespace datafile required for future import should be preserved.
<<Transport THE DATAFILE to the LOCAL DB DIRECTORY DUMP_DIR.  For example if connected
to asmcmd copy the file
cp cm_d1t304_p2011jan_tbs_ar.553.913864937  /tugbu_perf_02/BACKUPS/test_verification/
>>
```

7. Drop the partition, partition tablespace and archive tablespace (since they have been exported).

```
ALTER TABLE D1_INIT_MSRMT_DATA DROP SUBPARTITION P2011JAN_S181 UPDATE INDEXES;
DROP TABLESPACE CM_D1T304_P2011JAN_S181 INCLUDING CONTENTS AND DATAFILES;
DROP TABLESPACE CM_D1T304_P2011JAN_S181_ARC INCLUDING CONTENTS AND DATAFILES;
```

Restore Partition

1. Create separate tablespace to restore the partition.

```
CREATE BIGFILE TABLESPACE CM_XT039_P2011JAN 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 P2011FEB AT (TO_DATE('2011-02-01
00:00:01','SYYYY-MM-DD HH24:MI:SS'))
 INTO
 (
 PARTITION P2011JAN TABLESPACE CM_XT039_P2011JAN , PARTITION P2011FEB
 )
 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 P2011FEB AT (TO_DATE('2011-02-01
00:00:01','SYYYY-MM-DD HH24:MI:SS'))
 INTO
 (
 PARTITION P2011JAN TABLESPACE CM_F1T191_P2011JAN
LOB(BO_DATA_AREA,PRE_TRN_INIT_BO_DATA_AREA,PRE_TRN_FIN_BO_DATA_AREA,POST_TRN_BO_D
ATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM CACHE
TABLESPACE        CM_F1T191_P2011JAN )
, PARTITION P2011FEB
 )
 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_D1T304_P2011JAN_S181_ARC.DMP
PARTITION_OPTIONS=DEPARTITION LOGFILE=IMP_CM_D1T304_P2011JAN_S181_ARC.LOG
TRANSPORT_DATAFILES=/tugbu_perf_02/BACKUPS/test_verification/
cm_d1t304_p2011jan_tbs_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_P2011JAN_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_P2011JAN_ARC;
COMMIT;
INSERT /*+ APPEND PARALLEL */   INTO CISADM.CI_TD_MSG_PARM SELECT /*+ PARALLEL */ *
FROM CM_XT04_P2011JAN_ARC;
COMMIT;

INSERT /*+ APPEND PARALLEL */   INTO CISADM.CI_TD_LOG SELECT /*+ PARALLEL */ * FROM
CM_XT721_P2011JAN_ARC;
COMMIT;

INSERT /*+ APPEND PARALLEL */   INTO CISADM.CI_TD_SRTKEY SELECT /*+ PARALLEL */ * FROM
CM_XT041_P2011JAN_ARC;
```

```
COMMIT;

    INSERT /*+ APPEND PARALLEL */   INTO CISADM.CI_TD_DRLKEY SELECT /*+ PARALLEL */ *
FROM CM_XT037_P2011JAN_ARC;
    COMMIT;
```

7. Drop the archive tablespace after import is import and data loading is successful.

```
DROP TABLESPACE CM_XT039_P2011JAN_ARC INCLUDING CONTENTS AND DATAFILES;
```

Restore SubPartition

1. Create separate tablespace to restore the partition.

```
CREATE BIGFILE TABLESPACE CM_D1T304_P2011JAN_S181 DATAFILE 'DATADG' 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.D1_INIT_MSRMT_DATA SPLIT SUBPARTITION P2011JAN_SMAX AT (181)
 INTO
 (
SUBPARTITION P2011JAN_S181 TABLESPACE CM_D1T304_P2011JAN_S181 LOB(IMD_BO_DATA_AREA,
PREVEE_BO_DATA_AREA, POSTVEE_BO_DATA_AREA, TRACE_BO_DATA_AREA, RAW_BO_DATA_AREA) STORE
AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM CACHE TABLESPACE
CM_D1T304_P2011JAN_S181)
, SUBPARTITION P2011JAN_SMAX) UPDATE INDEXES;
```

3. Enable advanced compression after SPLIT partition as it will disable the compression

```
ALTER TABLE D1_INIT_MSRMT_DATA_CHAR ROW STORE COMPRESS ADVANCED;
ALTER TABLE D1_INIT_MSRMT_DATA_LOG ROW STORE COMPRESS ADVANCED;
ALTER TABLE D1_INIT_MSRMT_DATA_LOG_PARM ROW STORE COMPRESS ADVANCED;
```

4. Import tablespace using TRANSPORT_TABLESPACES method.

```
impdp system/manager DIRECTORY=DUMP_DIR DUMPFILE=CM_D1T304_P2011JAN_S181_ARC.DMP
PARTITION_OPTIONS=DEPARTITION LOGFILE=IMP_CM_D1T304_P2011JAN_S181_ARC.LOG
TRANSPORT_DATAFILES=/tugbu_perf_02/BACKUPS/test_verification/
cm_d1t304_p2011jan_tbs_ar.553.913864937
```

5. Load data into parent table first from the staging table

```
 ALTER SESSION ENABLE PARALLEL DML;

INSERT /*+ APPEND PARALLEL */   INTO CISADM.D1_INIT_MSRMT_DATA SELECT /*+ PARALLEL */
* FROM CM_D1T304_P2011JAN_S181_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 D1_INIT_MSRMT_DATA_CHAR SELECT /*+ PARALLEL */  *
FROM CM_D1T305_P2011JAN_S181_ARC;

COMMIT;
INSERT /*+ APPEND PARALLEL */   INTO D1_INIT_MSRMT_DATA_LOG SELECT /*+ PARALLEL */  *
FROM CM_D1T306_P2011JAN_S181_ARC;

COMMIT;

INSERT /*+ APPEND PARALLEL */   INTO D1_INIT_MSRMT_DATA_LOG_PARM SELECT /*+ PARALLEL */
* FROM CM_D1T307_P2011JAN_S181_ARC;

COMMIT;
```

7. Drop the archive tablespace after import is import and data loading is successful.

```
DROP TABLESPACE CM_D1T304_P2011JAN_S181_ARC INCLUDING CONTENTS AND DATAFILES;
```

Compress Partition (D1_MSRMT table only)

1. Create Compressed Partition Tablespace.

```
CREATE BIGFILE TABLESPACE CM_D1T298_P2011JAN_C DATAFILE '+DATADG' SIZE 50M
AUTOEXTEND ON MAXSIZE UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;
```

Note: Perform Steps 2 - 9 for each subpartition (S01 – SMAX)

2. Create and Load Data Into Staging Table.

```
CREATE TABLE D1_MSRMT_P2011JAN_S01 PARALLEL NOLOGGING TABLESPACE CM_D1T298_P2011JAN_C
AS
SELECT /*+ PARALLEL */ * FROM D1_MSRMT SUBPARTITION(P2011JAN_S01)
ORDER BY MEASR_COMP_ID, MSRMT_DTTM;
```

3. Enable Logging on Newly Created Staging Table

```
ALTER TABLE D1_MSRMT_P2011JAN_S01 NOPARALLEL LOGGING;
```

4. Create Primary Unique Index on Staging Table

```
CREATE UNIQUE INDEX D1T298P0_P2011JAN_S01
ON D1_MSRMT_P2011JAN_S01(MEASR_COMP_ID, MSRMT_DTTM)
PARALLEL NOLOGGING COMPRESS ADVANCED LOW TABLESPACE CM_D1T298_P2011JAN_C;
```

5. Create Primary Key Constraint on Staging Table

```
ALTER TABLE D1_MSRMT_P2011JAN_S01 ADD CONSTRAINT D1T298P0_P2011JAN_S01 PRIMARY
KEY(MEASR_COMP_ID, MSRMT_DTTM) USING INDEX;
```

6. Enable Logging on Primary Key Index.

```
ALTER INDEX D1T298P0_P2011JAN_S01 NOPARALLEL LOGGING;
```

7. Exchange D1_MSRMT Table Subpartition With Newly Created Staging Table.

```
ALTER TABLE D1_MSRMT EXCHANGE SUBPARTITION(P2011JAN_S01) WITH TABLE
D1_MSRMT_P2011JAN_S01 INCLUDING INDEXES;
```

Note: Ensure that steps 2-9 have been executed for each subpartition (S01 – SMAX) before continuing:

8. Drop Original Uncompressed Tablespace.

```
DROP TABLESPACE CM_D1T298_P2011JAN INCLUDING CONTENTS AND DATAFILES;
```

9. Change Partition Metadata to Reflect Compression Tablespace.

```
ALTER TABLE D1_MSRMT MODIFY DEFAULT ATTRIBUTES FOR PARTITION P2011JAN TABLESPACE
CM_D1T298_P2011JAN_C;
```

10. Rename Tablespace to Original Tablespace Name.

```
ALTER TABLESPACE CM_D1T298_P2011JAN_C RENAME TO CM_D1T298_P2011JAN;
```

Appendix E

Partitioning and Compression Recommendations

This section specifies the partitioning and compression strategies recommended for an initial Oracle Utilities Meter Data Management database configuration. It includes the following topics:

- Partitioning Recommendations
- Compression Recommendations

> **Note**: If Information Lifecycle Management is part of your implementation, please refer to the chapter Information Lifecycle Management and Data Archiving in MDM in this guide for instructions on partitioning objects when using ILM.

Partitioning Recommendations

In general, the recommendation is for a minimum of 'n' partitions for selective database objects, where 'n' is number of RAC nodes. The specific table level partitioning recommendations are as follows:

- The Table Partitioning scheme for Transaction tables is focused primarily on tables associated with Measurement MO, Measurement Log MO and Initial-Measurement-Data MO.
- D1_MSRMT, D1_MSRMT_CHAR, D1_MSRMT_LOG, D1_MSRMT_LOG_PARM tables can be partitioned by MSRMT_DTTM. Bi-monthly partitions is a good start. Subpartition these tables by MEASR_COMP_ID (8 subpartitions should be a good number to start with).
- D1_INIT_MSRMT_DATA table can be partitioned by D1_TO_DTTM. Bi-monthly partitions is a good start. Subpartition D1_INIT_MSRMT_DATA table by MEASR_COMP_ID (8 subpartitions should be a good number to start with).
- D1_INIT_MSRMT_DATA_CHAR, D1_INIT_MSRMT_DATA_LOG, D1_INIT_MSRMT_DATA_LOG_PARM tables are reference partitioned to the parent table.
- D1_INIT_MSRMT_DATA_K table can be partitioned by INIT_MSRMT_DATA_ID (8 sub partitions should be a good number to start with).

The following sections gives partition recommendation and can be used as reference. Create one tablespace per partition as needed. It includes the following:

- D1_MSRMT
- D1_MSRMT_CHAR

- D1_MSRMT_LOG
- D1_MSRMT_LOG_PARM
- D1_INIT_MSRMT_DATA
- D1_INIT_MSRMT_DATA_CHAR
- D1_INIT_MSRMT_DATA_K
- D1_INIT_MSRMT_DATA_LOG
- D1_INIT_MSRMT_DATA_LOG_PARM

D1_MSRMT

```
CREATE BIGFILE TABLESPACE CM_D1T298_P2011JAN DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED ;
CREATE BIGFILE TABLESPACE CM_D1T298_P2011MAR DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED ;
CREATE BIGFILE TABLESPACE CM_D1T298_P2011MAY DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED ;
CREATE BIGFILE TABLESPACE CM_D1T298_P2011JUL DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED ;
CREATE BIGFILE TABLESPACE CM_D1T298_P2011SEP DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED ;
CREATE BIGFILE TABLESPACE CM_D1T298_P2011NOV DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED ;
CREATE BIGFILE TABLESPACE CM_D1T298_PMAX DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON MAXSIZE
UNLIMITED ;

CREATE TABLE D1_MSRMT (

MEASR_COMP_ID CHAR(12) NOT NULL ENABLE,
MSRMT_DTTM DATE NOT NULL ENABLE,
BO_STATUS_CD   CHAR(12) DEFAULT ' ' NOT NULL ENABLE,
MSRMT_COND_FLG CHAR(6) DEFAULT ' ' NOT NULL ENABLE,
MSRMT_USE_FLG  CHAR(4) DEFAULT ' ' NOT NULL ENABLE,
MSRMT_LOCAL_DTTM DATE NOT NULL ENABLE,
MSRMT_VAL          NUMBER(16,6) DEFAULT 0 NOT NULL ENABLE,
ORIG_INIT_MSRMT_ID CHAR(14) DEFAULT ' ' NOT NULL ENABLE,
PREV_MSRMT_DTTM DATE,
MSRMT_VAL1  NUMBER(16,6) DEFAULT 0 NOT NULL ENABLE,
MSRMT_VAL2  NUMBER(16,6) DEFAULT 0 NOT NULL ENABLE,
MSRMT_VAL3  NUMBER(16,6) DEFAULT 0 NOT NULL ENABLE,
MSRMT_VAL4  NUMBER(16,6) DEFAULT 0 NOT NULL ENABLE,
MSRMT_VAL5  NUMBER(16,6) DEFAULT 0 NOT NULL ENABLE,
MSRMT_VAL6  NUMBER(16,6) DEFAULT 0 NOT NULL ENABLE,
MSRMT_VAL7  NUMBER(16,6) DEFAULT 0 NOT NULL ENABLE,
MSRMT_VAL8  NUMBER(16,6) DEFAULT 0 NOT NULL ENABLE,
MSRMT_VAL9  NUMBER(16,6) DEFAULT 0 NOT NULL ENABLE,
MSRMT_VAL10 NUMBER(16,6) DEFAULT 0 NOT NULL ENABLE,
BUS_OBJ_CD  CHAR(30) DEFAULT ' ' NOT NULL ENABLE,
CRE_DTTM DATE NOT NULL ENABLE,
STATUS_UPD_DTTM DATE NOT NULL ENABLE,
USER_EDITED_FLG CHAR(4) DEFAULT ' ' NOT NULL ENABLE,
VERSION NUMBER(5,0) DEFAULT 1 NOT NULL ENABLE,
LAST_UPDATE_DTTM DATE,
READING_VAL NUMBER(16,6),
COMBINED_MULTIPLIER NUMBER(12,6)
) ENABLE ROW MOVEMENT
PARTITION BY RANGE (MSRMT_DTTM)
SUBPARTITION BY range (MEASR_COMP_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 (744999999999),
subpartition S07 values less than (874999999999),
subpartition SMAX values less than (maxvalue)
)
(
PARTITION "P2011JAN" VALUES LESS THAN (TO_DATE('2011-02-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
TABLESPACE CM_D1T298_P2011JAN,
PARTITION "P2011MAR" VALUES LESS THAN (TO_DATE('2011-04-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
TABLESPACE CM_D1T298_P2011MAR,
PARTITION "P2011MAY" VALUES LESS THAN (TO_DATE('2011-06-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
TABLESPACE CM_D1T298_P2011MAY,
PARTITION "P2011JUL" VALUES LESS THAN (TO_DATE('2011-08-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
TABLESPACE CM_D1T298_P2011JUL,
PARTITION "P2011SEP" VALUES LESS THAN (TO_DATE('2011-10-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
TABLESPACE CM_D1T298_P2011SEP,
PARTITION "P2011NOV" VALUES LESS THAN (TO_DATE('2011-12-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
TABLESPACE CM_D1T298_P2011NOV,
PARTITION "PMAX" VALUES LESS THAN (MAXVALUE)
TABLESPACE CM_D1T298_PMAX
);

CREATE UNIQUE INDEX D1T298P0 ON D1_MSRMT(MEASR_COMP_ID, MSRMT_DTTM) LOCAL COMPRESS ADVANCED
LOW;
```

```
ALTER TABLE D1_MSRMT ADD CONSTRAINT D1T298P0 PRIMARY KEY(MEASR_COMP_ID, MSRMT_DTTM) USING
INDEX;
```

D1_MSRMT_CHAR

```
CREATE BIGFILE TABLESPACE CM_D1T299_P2011JAN DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED ;
CREATE BIGFILE TABLESPACE CM_D1T299_P2011MAR DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED ;
CREATE BIGFILE TABLESPACE CM_D1T299_P2011MAY DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED ;
CREATE BIGFILE TABLESPACE CM_D1T299_P2011JUL DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED ;
CREATE BIGFILE TABLESPACE CM_D1T299_P2011SEP DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED ;
CREATE BIGFILE TABLESPACE CM_D1T299_P2011NOV DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED ;
CREATE BIGFILE TABLESPACE CM_D1T299_PMAX DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON MAXSIZE
UNLIMITED ;

CREATE TABLE D1_MSRMT_CHAR(
MEASR_COMP_ID CHAR(12) NOT NULL ENABLE,
MSRMT_DTTM DATE 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,
LAST_UPDATE_DTTM DATE
) ENABLE ROW MOVEMENT
PARTITION BY RANGE (MSRMT_DTTM)
SUBPARTITION BY range (MEASR_COMP_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 (744999999999),
subpartition S07 values less than (874999999999),
subpartition SMAX values less than (maxvalue)
)
(
PARTITION "P2011JAN" VALUES LESS THAN (TO_DATE('2011-02-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
TABLESPACE CM_D1T299_P2011JAN,
PARTITION "P2011MAR" VALUES LESS THAN (TO_DATE('2011-04-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
TABLESPACE CM_D1T299_P2011MAR,
PARTITION "P2011MAY" VALUES LESS THAN (TO_DATE('2011-06-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
TABLESPACE CM_D1T299_P2011MAY,
PARTITION "P2011JUL" VALUES LESS THAN (TO_DATE('2011-08-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
TABLESPACE CM_D1T299_P2011JUL,
PARTITION "P2011SEP" VALUES LESS THAN (TO_DATE('2011-10-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
TABLESPACE CM_D1T299_P2011SEP,
PARTITION "P2011NOV" VALUES LESS THAN (TO_DATE('2011-12-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
TABLESPACE CM_D1T299_P2011NOV,
PARTITION "PMAX" VALUES LESS THAN (MAXVALUE)
TABLESPACE CM_D1T299_PMAX
);

CREATE BIGFILE TABLESPACE CM_D1T299_IND DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON MAXSIZE
UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;

CREATE UNIQUE INDEX  D1T299P0 ON D1_MSRMT_CHAR (
MEASR_COMP_ID, MSRMT_DTTM, CHAR_TYPE_CD, SEQ_NUM
)  LOCAL COMPRESS ADVANCED LOW;

ALTER TABLE D1_MSRMT_CHAR  ADD CONSTRAINT D1T299P0 PRIMARY KEY (MEASR_COMP_ID, MSRMT_DTTM,
CHAR_TYPE_CD, SEQ_NUM) USING INDEX ;

CREATE INDEX D1T299S1 ON D1_MSRMT_CHAR(SRCH_CHAR_VAL)
GLOBAL PARTITION BY HASH(SRCH_CHAR_VAL)
 (
 PARTITION P1 TABLESPACE CM_D1T299_IND,
 PARTITION P2 TABLESPACE CM_D1T299_IND,
 PARTITION P3 TABLESPACE CM_D1T299_IND,
 PARTITION P4 TABLESPACE CM_D1T299_IND,
```

```
PARTITION P5 TABLESPACE CM_D1T299_IND,
PARTITION P6 TABLESPACE CM_D1T299_IND,
PARTITION P7 TABLESPACE CM_D1T299_IND,
PARTITION P8 TABLESPACE CM_D1T299_IND
)
TABLESPACE CM_D1T304_IND;
```

D1_MSRMT_LOG

```
CREATE BIGFILE TABLESPACE CM_D1T300_P2011JAN DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED ;
CREATE BIGFILE TABLESPACE CM_D1T300_P2011MAR DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED ;
CREATE BIGFILE TABLESPACE CM_D1T300_P2011MAY DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED ;
CREATE BIGFILE TABLESPACE CM_D1T300_P2011JUL DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED ;
CREATE BIGFILE TABLESPACE CM_D1T300_P2011SEP DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED ;
CREATE BIGFILE TABLESPACE CM_D1T300_P2011NOV DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED ;
CREATE BIGFILE TABLESPACE CM_D1T300_PMAX DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON MAXSIZE
UNLIMITED ;

CREATE TABLE D1_MSRMT_LOG (
MEASR_COMP_ID CHAR(12),
MSRMT_DTTM DATE,
SEQNO NUMBER(5,0),
ORIG_INIT_MSRMT_ID CHAR(14) DEFAULT ' ' NOT NULL ENABLE,
BUS_OBJ_CD CHAR(30) DEFAULT ' ' 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,
LOG_DTTM DATE NOT NULL ENABLE,
MESSAGE_CAT_NBR NUMBER(5,0) DEFAULT 0 NOT NULL ENABLE,
MESSAGE_NBR NUMBER(5,0) DEFAULT 0 NOT NULL ENABLE,
USER_ID CHAR(8) DEFAULT ' ' NOT NULL ENABLE,
VERSION NUMBER(5,0) DEFAULT 1 NOT NULL ENABLE,
MSRMT_LOG_ENTRY_TYPE_FLG CHAR(4) DEFAULT ' ' NOT NULL ENABLE,
BO_DATA_AREA CLOB
)
  LOB (BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM CACHE)
ENABLE ROW MOVEMENT
PARTITION BY RANGE (MSRMT_DTTM)
SUBPARTITION BY range (MEASR_COMP_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 (744999999999),
subpartition S07 values less than (874999999999),
subpartition SMAX values less than (maxvalue)
)
(
PARTITION "P2011JAN" VALUES LESS THAN (TO_DATE('2011-02-01 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_D1T300_P2011JAN )
TABLESPACE CM_D1T300_P2011JAN,
PARTITION "P2011MAR" VALUES LESS THAN (TO_DATE('2011-04-01 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_D1T300_P2011MAR )
TABLESPACE CM_D1T300_P2011MAR,
PARTITION "P2011MAY" VALUES LESS THAN (TO_DATE('2011-06-01 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_D1T300_P2011MAY )
TABLESPACE CM_D1T300_P2011MAY,
PARTITION "P2011JUL" VALUES LESS THAN (TO_DATE('2011-08-01 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_D1T300_P2011JUL )
TABLESPACE CM_D1T300_P2011JUL,
PARTITION "P2011SEP" VALUES LESS THAN (TO_DATE('2011-10-01 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_D1T300_P2011SEP )
TABLESPACE CM_D1T300_P2011SEP,
PARTITION "P2011NOV" VALUES LESS THAN (TO_DATE('2011-12-01 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_D1T300_P2011NOV )
TABLESPACE CM_D1T300_P2011NOV,
```

```
PARTITION "PMAX" VALUES LESS THAN (MAXVALUE)
LOB (BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM CACHE
TABLESPACE CM_D1T300_PMAX )
TABLESPACE CM_D1T300_PMAX
);

CREATE UNIQUE INDEX D1T300P0 ON D1_MSRMT_LOG (
MEASR_COMP_ID, MSRMT_DTTM, SEQNO
)   LOCAL COMPRESS ADVANCED LOW;

ALTER TABLE D1_MSRMT_LOG ADD CONSTRAINT D1T300P0 PRIMARY KEY (MEASR_COMP_ID, MSRMT_DTTM,
SEQNO) USING INDEX ;
```

D1_MSRMT_LOG_PARM

```
CREATE BIGFILE TABLESPACE CM_D1T301_P2011JAN DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED ;
CREATE BIGFILE TABLESPACE CM_D1T301_P2011MAR DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED ;
CREATE BIGFILE TABLESPACE CM_D1T301_P2011MAY DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED ;
CREATE BIGFILE TABLESPACE CM_D1T301_P2011JUL DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED ;
CREATE BIGFILE TABLESPACE CM_D1T301_P2011SEP DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED ;
CREATE BIGFILE TABLESPACE CM_D1T301_P2011NOV DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED ;
CREATE BIGFILE TABLESPACE CM_D1T301_PMAX DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON MAXSIZE
UNLIMITED ;


CREATE TABLE D1_MSRMT_LOG_PARM (
MEASR_COMP_ID CHAR(12),
MSRMT_DTTM DATE,
SEQNO NUMBER(5,0), PARM_SEQ NUMBER(3,0),
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
)
ENABLE ROW MOVEMENT
PARTITION BY RANGE (MSRMT_DTTM)
SUBPARTITION BY range (MEASR_COMP_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 (744999999999),
subpartition S07 values less than (874999999999),
subpartition SMAX values less than (maxvalue)
)
(
PARTITION "P2011JAN" VALUES LESS THAN (TO_DATE('2011-02-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
TABLESPACE CM_D1T301_P2011JAN,
PARTITION "P2011MAR" VALUES LESS THAN (TO_DATE('2011-04-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
TABLESPACE CM_D1T301_P2011MAR,
PARTITION "P2011MAY" VALUES LESS THAN (TO_DATE('2011-06-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
TABLESPACE CM_D1T301_P2011MAY,
PARTITION "P2011JUL" VALUES LESS THAN (TO_DATE('2011-08-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
TABLESPACE CM_D1T301_P2011JUL,
PARTITION "P2011SEP" VALUES LESS THAN (TO_DATE('2011-10-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
TABLESPACE CM_D1T301_P2011SEP,
PARTITION "P2011NOV" VALUES LESS THAN (TO_DATE('2011-12-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
TABLESPACE CM_D1T301_P2011NOV,
PARTITION "PMAX" VALUES LESS THAN (MAXVALUE)
TABLESPACE CM_D1T301_PMAX
);
CREATE UNIQUE INDEX D1T301P0 ON D1_MSRMT_LOG_PARM (
MEASR_COMP_ID, MSRMT_DTTM, SEQNO, PARM_SEQ
)  INDEX LOCAL COMPRESS ADVANCED LOW;

ALTER TABLE D1_MSRMT_LOG_PARM ADD CONSTRAINT D1T301P0 PRIMARY KEY (MEASR_COMP_ID,
MSRMT_DTTM, SEQNO, PARM_SEQ) USING INDEX;
```

D1_INIT_MSRMT_DATA

```
CREATE BIGFILE TABLESPACE CM_D1T304_P2011JAN DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED ;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011MAR DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED ;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011MAY DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED ;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011JUL DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED ;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011SEP DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED ;
CREATE BIGFILE TABLESPACE CM_D1T304_P2011NOV DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON
MAXSIZE UNLIMITED ;
CREATE BIGFILE TABLESPACE CM_D1T304_PMAX DATAFILE '+DATADG' SIZE 50M AUTOEXTEND ON MAXSIZE
UNLIMITED ;

CREATE TABLE D1_INIT_MSRMT_DATA
  (
   INIT_MSRMT_DATA_ID CHAR(14) NOT NULL ENABLE,
    MEASR_COMP_ID      CHAR(12) DEFAULT ' ' NOT NULL ENABLE,
    D1_FROM_DTTM DATE,
    D1_TO_DTTM DATE,
    DATA_SRC_FLG        CHAR(4) DEFAULT ' ' NOT NULL ENABLE,
    TIME_ZONE_CD        CHAR(10) DEFAULT ' ' NOT NULL ENABLE,
    BUS_OBJ_CD          CHAR(30) DEFAULT ' ' NOT NULL ENABLE,
    BO_STATUS_CD        CHAR(12) DEFAULT ' ' NOT NULL ENABLE,
    BO_STATUS_REASON_CD VARCHAR2(30) DEFAULT ' ' NOT NULL ENABLE,
    IMD_BO_DATA_AREA CLOB,
    STATUS_UPD_DTTM DATE NOT NULL ENABLE,
    CRE_DTTM DATE NOT NULL ENABLE,
    VERSION    NUMBER(5,0) DEFAULT 1 NOT NULL ENABLE,
    IMD_EXT_ID VARCHAR2(120),
    PREVEE_BO_DATA_AREA CLOB,
    POSTVEE_BO_DATA_AREA CLOB,
    TRACE_BO_DATA_AREA CLOB,
    RAW_BO_DATA_AREA CLOB,
    LAST_UPDATE_DTTM DATE,
    ILM_DT DATE,
    ILM_ARCH_SW CHAR(1),
    RETENTION_PERIOD NUMBER(5,0) DEFAULT 99999 NOT NULL ENABLE
  )
ENABLE ROW MOVEMENT
LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM CACHE)
LOB ( POSTVEE_BO_DATA_AREA ) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM
CACHE)
LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM CACHE)
LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM CACHE)
LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (ENABLE STORAGE IN ROW COMPRESS MEDIUM CACHE)
PARTITION BY RANGE (D1_TO_DTTM)
SUBPARTITION BY range (MEASR_COMP_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 SMAX VALUES LESS THAN (MAXVALUE)
)
(
PARTITION "P2011JAN" VALUES LESS THAN (TO_DATE('2011-02-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JAN)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JAN)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JAN)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JAN)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JAN)
TABLESPACE CM_D1T304_P2011JAN,
PARTITION "P2011MAR" VALUES LESS THAN (TO_DATE('2011-04-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAR)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAR)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAR)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAR)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAR)
TABLESPACE CM_D1T304_P2011MAR,
PARTITION "P2011MAY" VALUES LESS THAN (TO_DATE('2011-06-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAY)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAY)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAY)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAY)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011MAY)
```

```
TABLESPACE CM_D1T304_P2011MAY,
PARTITION "P2011JUL" VALUES LESS THAN (TO_DATE('2011-08-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUL)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUL)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUL)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUL)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011JUL)
TABLESPACE CM_D1T304_P2011JUL,
PARTITION "P2011SEP" VALUES LESS THAN (TO_DATE('2011-10-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011SEP)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011SEP)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011SEP)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011SEP)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011SEP)
TABLESPACE CM_D1T304_P2011SEP,
PARTITION "P2011NOV" VALUES LESS THAN (TO_DATE('2011-12-01 00:00:01', 'SYYYY-MM-DD
HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011NOV)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011NOV)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011NOV)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011NOV)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_P2011NOV)
TABLESPACE CM_D1T304_P2011NOV,
PARTITION "PMAX" VALUES LESS THAN (MAXVALUE)
    LOB (PREVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_PMAX)
    LOB (POSTVEE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_TABLESPACE
CM_D1T304_PMAX)
    LOB (TRACE_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_PMAX)
    LOB (RAW_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_PMAX)
    LOB (IMD_BO_DATA_AREA) STORE AS SECUREFILE (TABLESPACE CM_D1T304_PMAX)
TABLESPACE CM_D1T304_PMAX
);

CREATE BIGFILE TABLESPACE CM_D1T304_IND DATAFILE '+DATA' SIZE 50M AUTOEXTEND ON MAXSIZE
UNLIMITED DEFAULT ROW STORE COMPRESS ADVANCED;

CREATE UNIQUE INDEX D1T304P0 ON D1_INIT_MSRMT_DATA (
INIT_MSRMT_DATA_ID
) TABLESPACE CM_D1T304_IND
GLOBAL PARTITION BY RANGE (INIT_MSRMT_DATA_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 D1_INIT_MSRMT_DATA ADD CONSTRAINT D1T304P0 PRIMARY KEY (INIT_MSRMT_DATA_ID)
USING INDEX ;

CREATE INDEX D1T304S1 ON D1_INIT_MSRMT_DATA (MEASR_COMP_ID, BO_STATUS_CD, BUS_OBJ_CD,
D1_TO_DTTM, D1_FROM_DTTM) TABLESPACE CM_D1T304_IND
GLOBAL PARTITION BY RANGE (MEASR_COMP_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;


CREATE UNIQUE INDEX D1T304S2 ON D1_INIT_MSRMT_DATA(BUS_OBJ_CD, BO_STATUS_CD,
INIT_MSRMT_DATA_ID)
GLOBAL PARTITION BY HASH (BUS_OBJ_CD, BO_STATUS_CD,  INIT_MSRMT_DATA_ID)
(
 PARTITION P1 TABLESPACE CM_D1T304_IND,
 PARTITION P2 TABLESPACE CM_D1T304_IND,
 PARTITION P3 TABLESPACE CM_D1T304_IND,
 PARTITION P4 TABLESPACE CM_D1T304_IND,
 PARTITION P5 TABLESPACE CM_D1T304_IND,
 PARTITION P6 TABLESPACE CM_D1T304_IND,
 PARTITION P7 TABLESPACE CM_D1T304_IND,
 PARTITION P8 TABLESPACE CM_D1T304_IND
)
COMPRESS ADVANCED LOW;
```

```
CREATE UNIQUE INDEX D1T304S3 ON D1_INIT_MSRMT_DATA(IMD_EXT_ID,INIT_MSRMT_DATA_ID)
GLOBAL PARTITION BY HASH(IMD_EXT_ID)
(
 PARTITION P1 TABLESPACE CM_D1T304_IND,
 PARTITION P2 TABLESPACE CM_D1T304_IND,
 PARTITION P3 TABLESPACE CM_D1T304_IND,
 PARTITION P4 TABLESPACE CM_D1T304_IND,
 PARTITION P5 TABLESPACE CM_D1T304_IND,
 PARTITION P6 TABLESPACE CM_D1T304_IND,
 PARTITION P7 TABLESPACE CM_D1T304_IND,
 PARTITION P8 TABLESPACE CM_D1T304_IND
 )
COMPRESS ADVANCED LOW;
```

D1_INIT_MSRMT_DATA_CHAR

```
CREATE TABLE D1_INIT_MSRMT_DATA_CHAR
  (
    INIT_MSRMT_DATA_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,
    LAST_UPDATE_DTTM DATE,
     CONSTRAINT D1_INIT_MSRMT_DATA_CHAR_FK FOREIGN KEY(INIT_MSRMT_DATA_ID) REFERENCES
D1_INIT_MSRMT_DATA
    )
PARTITION BY REFERENCE (D1_INIT_MSRMT_DATA_CHAR_FK)
ENABLE ROW MOVEMENT;

CREATE UNIQUE INDEX D1T305P0 ON D1_INIT_MSRMT_DATA_CHAR(INIT_MSRMT_DATA_ID, CHAR_TYPE_CD,
SEQ_NUM) TABLESPACE CM_D1T304_IND
GLOBAL PARTITION BY RANGE(INIT_MSRMT_DATA_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 D1_INIT_MSRMT_DATA_CHAR ADD CONSTRAINT D1T305P0 PRIMARY KEY
(INIT_MSRMT_DATA_ID, CHAR_TYPE_CD, SEQ_NUM) USING INDEX ;

CREATE INDEX D1T305S1 ON D1_INIT_MSRMT_DATA_CHAR(SRCH_CHAR_VAL)
GLOBAL PARTITION BY HASH(SRCH_CHAR_VAL)
 (
 PARTITION P1 TABLESPACE CM_D1T304_IND,
 PARTITION P2 TABLESPACE CM_D1T304_IND,
 PARTITION P3 TABLESPACE CM_D1T304_IND,
 PARTITION P4 TABLESPACE CM_D1T304_IND,
 PARTITION P5 TABLESPACE CM_D1T304_IND,
 PARTITION P6 TABLESPACE CM_D1T304_IND,
 PARTITION P7 TABLESPACE CM_D1T304_IND,
 PARTITION P8 TABLESPACE CM_D1T304_IND
 );
```

D1_INIT_MSRMT_DATA_K

```
CREATE TABLE D1_INIT_MSRMT_DATA_K (
INIT_MSRMT_DATA_ID CHAR(14),
ENV_ID NUMBER(6,0) NOT NULL ENABLE,
CONSTRAINT D1T314P0 PRIMARY KEY (INIT_MSRMT_DATA_ID, ENV_ID) ENABLE
)
ORGANIZATION INDEX ENABLE ROW MOVEMENT
PARTITION BY RANGE (INIT_MSRMT_DATA_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_D1T314_IND ;
```

D1_INIT_MSRMT_DATA_LOG

```
CREATE TABLE D1_INIT_MSRMT_DATA_LOG
  (
    INIT_MSRMT_DATA_ID  CHAR(14) NOT NULL ENABLE,
    SEQNO               NUMBER(5,0) NOT NULL ENABLE,
    BO_STATUS_CD        CHAR(12) DEFAULT ' ' NOT NULL ENABLE,
    BO_STATUS_REASON_CD VARCHAR2(30) DEFAULT ' ' 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,
    LOG_DTTM            DATE NOT NULL ENABLE,
    LOG_ENTRY_TYPE_FLG  CHAR(4) 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,
    USER_ID             CHAR(8) DEFAULT ' ' NOT NULL ENABLE,
    VERSION             NUMBER(5,0) DEFAULT 1 NOT NULL ENABLE,
    LAST_UPDATE_DTTM DATE,
     CONSTRAINT D1_INIT_MSRMT_DATA_LOG_FK FOREIGN KEY(INIT_MSRMT_DATA_ID) REFERENCES
D1_INIT_MSRMT_DATA
    )
PARTITION BY REFERENCE (D1_INIT_MSRMT_DATA_LOG_FK)
ENABLE ROW MOVEMENT;

CREATE UNIQUE INDEX D1T306P0 ON D1_INIT_MSRMT_DATA_LOG (INIT_MSRMT_DATA_ID, SEQNO)
TABLESPACE CM_D1T304_IND
GLOBAL PARTITION BY RANGE(INIT_MSRMT_DATA_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 D1_INIT_MSRMT_DATA_LOG ADD CONSTRAINT D1T306P0 PRIMARY KEY
(INIT_MSRMT_DATA_ID, SEQNO) USING INDEX  ;
```

D1_INIT_MSRMT_DATA_LOG_PARM

```
CREATE TABLE D1_INIT_MSRMT_DATA_LOG_PARM
  (
    INIT_MSRMT_DATA_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,
    LAST_UPDATE_DTTM DATE,
    CONSTRAINT D1_INIT_MSRMT_DATA_LOG_PARM_FK FOREIGN KEY(INIT_MSRMT_DATA_ID) REFERENCES
D1_INIT_MSRMT_DATA
    )
PARTITION BY REFERENCE (D1_INIT_MSRMT_DATA_LOG_PARM_FK)
ENABLE ROW MOVEMENT;


CREATE UNIQUE INDEX D1T307P0 ON D1_INIT_MSRMT_DATA_LOG_PARM(INIT_MSRMT_DATA_ID, SEQNO,
PARM_SEQ) TABLESPACE CM_D1T304_IND
GLOBAL PARTITION BY RANGE(INIT_MSRMT_DATA_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 D1_INIT_MSRMT_DATA_LOG_PARM ADD CONSTRAINT D1T307P0
PRIMARY KEY (INIT_MSRMT_DATA_ID, SEQNO, PARM_SEQ) USING INDEX ;
```

Compression Recommendations

It is highly recommended to use the following guidelines with regard to compression.

1. For all transactional data tables including ILM enabled tables (except D1_MSRMT* tables)
    a. For easier operational manageability, it is recommended to enable the compression at tablespace level while creating separate tablespaces for each logical unit of archival (like a parent table partition and the corresponding referenced child table partitions).
    b. Use securefile medium compression for LOBs.
    b. On Oracle database 12c
        - Use advanced compression for table data compression.
        - Compress indexes using advanced low compression (using 'compress advanced low' clause).
    b. On Oracle database 11g
        - Use OLTP compression for table data and compression indexes using default compression.
3. For D1_MSRMT* tables:
    a. Keep current table partitions uncompressed for D1_MSRMT. Other D1_MSRMT* tables should use compressed tablespaces for all partitions.
    b. For the D1_MSRMT table- Periodically (recommended monthly), compress the data by reloading into a staging table followed by partition exchange. It is highly recommended to use bulk load CTAS operation with parallel clause during the reload.
        - Use 'QUERY HIGH' compression for Exadata implementations.
        - For non-Exadata implementations, on 12c use 'row store compress advanced' and on 11g use OLTP compression.
    b. For indexes
        - On Oracle database 12c, Compress indexes using advanced low compression (using 'compress advanced low' clause).
        - On Oracle database 11g, use default index compression.

Appendix F

Oracle Application Framework System Table Guide

This section lists the system tables owned by the Oracle Utilities Application Framework V4.2.0.3 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.
- The standards 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. This section contains:

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).

DB Process

Properties	Description
Tables	CI_DB_PROC, CI_DB_PROC_L, CI_DB_INSTR, CI_DB_INSTR_L, L, CI_DB_INSTR_OVRD
Initial Data	Copy DB Process (CL-COPDB). This DB process allows users to copy a DB process from one database to another using Config Lab utility.

Display Profile

The Display Profile Code is referenced in the User (SC_USER) table.

Properties	Description
Tables	CI_DISP_PROF, CI_DISP_PROF_L
Initial Data	North America (NORTHAM) and Europe (EURO).

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).

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

Properties	Description
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 Meter Data Management 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, 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)
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) 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)

Properties	Description
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)
Customer Modification	Adhoc Characteristic Value Validation Rule (ADHOC_VAL_ALG_CD) on Characteristic Entity Table (CI_CHAR_ENTITY)

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

Properties	Description
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

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
Standard Data Fields	Business Object (BUS_OBJ_CD), Extendable Lookup Value (F1_EXT_LOOKUP_VALUE)
Customer Modification	Override Description (DESCR_OVRD) on Extendable Lookup Field Value Language Table (F1_EXT_LOOKUP_VAL_L)

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)
Customer Modification	Debug (DEBUG_SW), Active (ACTIVE_SW), Trace (TRACE_SW), Post Error (POST_ERROR_SW), Request XSL (REQUEST_XSL), Response XSL (RESPONSE_XSL)

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 an Oracle Utilities Application Framework owned lookup field. When a new value is added, the Owner Flag is used to prefix the value. For example, when Oracle Utilities Customer Care and Billing 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 8000 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

Properties	Description
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	Audit Switches (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

Properties	Description
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

Properties	Description
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

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

Portal and Zone

Properties	Description
Tables	CI_PORTAL, CI_PORTAL_L, CI_PORTAL_ZONE, 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).

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)

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 XAI Option Table (CI_XAI_OPTION)

The following XAI tables might have system data installed upon the initial installation but a subsequence system data upgrade process will not update the content of these table unless the change is documented in the database upgrade guide: 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

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)

Oracle Utilities Application Framework Only Tables

All data of the tables in this group belong to the Oracle Utilities Application Framework. No data modification or addition is allowed for these tables by base product development and customer modification. When an environment is upgraded to the next release of the Oracle Utilities Application Framework, the upgrade process will refresh the data in these tables.

- CI_MD_AT_DTL / CI_MD_AT_DTL_L
- CI_MD_ATT_TY
- CI_MD_CTL / CI_MD_CTL_L
- CI_MD_CTL_TMPL
- CI_MD_ELTY / CI_MD_ELTY_L
- CI_MD_ELTY_AT
- CI_MD_LOOKUP / CI_MD_LOOKUP_F
- CI_MD_PDF / CI_MD_PDF_VAL
- CI_MD_MSG / CI_MD_MSG_L
- CI_MD_SRC_TYPE / CI_MD_SRC_TYPE_L
- CI_MD_TMPL / CI_MD_TMPL_L
- CI_MD_TMPL_ELTY
- CI_MD_TMPL_VAR / CI_MD_TMPL_VAR_L
- CI_MD_VAR / CI_MD_VAR_DTL / CI_MD_VAR_DTL_L
- CI_XAI_EXECUTER / CI_XAI_EXECUTER_L

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.

> **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

Table Name	Upgrade Action	Description
CI_MD_PRG_LOC	MG	Program Location
CI_MD_PRG_MOD	MG	Program Module
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

Table Name	Upgrade Action	Description
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
CI_PORTAL	MG	Portal
CI_PORTAL_L	MG	Portal Language
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	XAI Class
CI_XAI_CLASS_L	MG	XAI Class Language
CI_XAI_ENV_HNDL	MG	XAI Envelope Handler

Table Name	Upgrade Action	Description
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
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

Table Name	Upgrade Action	Description
F1_DATA_AREA_L	MG	Data Area Language
F1_DB_OBJECTS_REPO	MG	Database Objects Repository
F1_EXT_LOOKUP_VAL	MG	Extendable Lookup
F1_EXT_LOOKUP_VAL_L	MG	Extendable Lookup Language
F1_IWS_ANN	MG	Inbound Web Service Annotation
F1_IWS_ANN_L	MG	Inbound Web Service Annotation Language
F1_IWS_ANN_PARM	MG	Inbound Web Service Annotation Parameter
F1_IWS_ANN_TYPE	MG	Inbound Web Service Annotation Type
F1_IWS_ANN_TYPE_L	MG	Inbound Web Service Annotation Type Language
F1_IWS_ANN_TYPE_PARM	MG	Inbound Web Service Annotation Type Parm
F1_IWS_ANN_TYPE_PARM_L	MG	Inbound 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_IWS_SVC_OPER_L	MG	Inbound Web Service Operations Language
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

Table Name	Upgrade Action	Description
F1_MIGR_REQ	MG	Migration Request
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_SCHEMA	MG	Schema
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_ACC_GRP	KP	Access Group
CI_ACC_GRP_DAR	KP	Access Group / Data Access Group
CI_ACC_GRP_L	KP	Access Group Language
CI_APP_SVC_SCTY	KP	Security Type Application Service
CI_CAL_HOL	KP	Work Calendar Holidays
CI_CAL_HOL_L	KP	Work Calendar Holidays Language
CI_CAL_WORK	KP	Work Calendar
CI_CAL_WORK_L	KP	Work Calendar Language
CI_CHTY_TDTY	KP	To Do Type Template Characteristics
CI_COUNTRY	KP	Country
CI_COUNTRY_L	KP	Country Language
CI_CURRENCY_CD	KP	Currency Code
CI_CURRENCY_CD_L	KP	Currency Code Language
CI_DAR	KP	Data Access Role
CI_DAR_L	KP	Data Access Language
CI_DAR_USR	KP	Data Access User
CI_DB_INSTR	KP	DB Process Instruction

Table Name	Upgrade Action	Description
CI_DB_INSTR_ALG	KP	DB Process Instruction Algorithm
CI_DB_INSTR_L	KP	DB Process Instruction Language
CI_DB_INST_OVRD	KP	DB Process Instruction Override
CI_DB_PROC	KP	DB Process
CI_DB_PROC_L	KP	DB Process Language
CI_DISP_PROF	KP	Display Profile
CI_DISP_PROF_L	KP	Display Profile Language
CI_ENV_REF	KP	Environment Reference
CI_ENV_REF_L	KP	Environment Reference Language
CI_FUNC	KP	Function
CI_FUNC_FLD	KP	Function Field
CI_FUNC_FLD_L	KP	Function Field Language
CI_FUNC_L	KP	Function Language
CI_GEO_TYPE	KP	Geographic Type
CI_GEO_TYPE_L	KP	Geographic Type Language
CI_INSTALL_ALG	KP	Installation Algorithm
CI_INSTALL_MSG	KP	Installation Message
CI_INSTALL_MSG_L	KP	Installation Message Language
CI_INSTALL_PROD	KP	Installation Product
CI_MD_RPT	KP	Report Definition
CI_MD_RPT_L	KP	Report Language
CI_MD_RPT_LBL	KP	Report Labels
CI_MD_RPT_PARM	KP	Report Parameters
CI_MD_RPT_PARM_L	KP	Report Parameters Language
CI_MD_TOOLREP_XML	KP	MD Tool Reference XML
CI_MD_TOOL_REP	KP	MD Tool Reference
CI_NT_DNTY_CTXT	KP	Notification Download Type Context
CI_NT_DWN_FORM	KP	Notification Download Format

Table Name	Upgrade Action	Description
CI_NT_DWN_FORM_L	KP	Notification Download Format Language
CI_NT_DWN_PROF	KP	Notification Download Profile
CI_NT_DWN_PROF_L	KP	Notification Download Profile Language
CI_NT_DWN_TYPE	KP	Notification Download Type
CI_NT_DWN_TYPE_L	KP	Notification Download Type Language
CI_NT_UP_XTYPE	KP	Notification Upload Type
CI_NT_UP_XTYPE_L	KP	Notification Upload Type Language
CI_NT_XID	KP	External System
CI_NT_XID_L	KP	External System Language
CI_PHONE_TYPE	KP	Phone Type
CI_PHONE_TYPE_L	KP	Phone Type Language
CI_ROLE	KP	Role
CI_ROLE_L	KP	Role Language
CI_ROLE_USER	KP	Role User
CI_RPT_OPTION	KP	Report Options
CI_SC_AUTH_LVL	KP	Security Type Auth Level
CI_SC_AUTH_LVL_L	KP	Security Type Auth Level Language
CI_SC_TYPE	KP	Security Type
CI_SC_TYPE_L	KP	Security Type Language
CI_SEAS_SHIFT	KP	Seasonal Time Shift Schedule
CI_SEAS_TM_SHIFT	KP	Seasonal Time Shift
CI_SEAS_TM_SHIFT_L	KP	Seasonal Shift Language
CI_STATE	KP	State
CI_STATE_L	KP	State Language
CI_TD_EX_LIST	KP	To Do Type Message Overrides
CI_TD_TYPE_ALG	KP	To Do Type Algorithms
CI_TD_TYPE_CHAR	KP	To Do Type Characteristic
CI_TD_VAL_ROLE	KP	To Do Type Role

Table Name	Upgrade Action	Description
CI_TIME_ZONE	KP	Time Zone
CI_TIME_ZONE_L	KP	Time Zone Language
CI_USR_GRP_SC	KP	User Group Security Type
CI_USR_PORTAL	KP	User Portal
CI_USR_SCR	KP	User Scripts
CI_USR_ZONE	KP	User Zone
CI_WFM	KP	Feature Configuration
CI_WFM_L	KP	Feature Configuration Language
CI_WFM_MSG	KP	Feature Configuration Message
CI_WFM_OPT	KP	Feature Configuration Options
CI_WF_EVT_TYPE	KP	WF Event Type
CI_WF_EVT_TYPE_L	KP	WF Event Type Language
CI_WF_PP	KP	WF Process Profile
CI_WF_PP_L	KP	WF Process Profile Language
CI_WF_PP_NT	KP	WF Process Notification
CI_WF_PP_NT_CRT	KP	WF Process Notification Criteria
CI_WF_PROC_SCHED	KP	WF Process Creation Schedule
CI_WF_PROC_SCHED_K	KP	WF Process Creation Schedule Key
CI_WF_PROC_TMPL	KP	WF Process Template
CI_WF_PROC_TMPL_L	KP	WF Process Template Language
CI_WF_RESP	KP	WF Response
CI_WF_RESP_DEP	KP	WF Response Dependency
CI_XAI_JDBC_CON	KP	XAI JDBC Connection
CI_XAI_JDBC_CON_L	KP	XAI JDBC Connection Language
CI_XAI_JMS_CON	KP	XAI JMS Connection
CI_XAI_JMS_CON_L	KP	XAI JMS Connection Language
CI_XAI_JMS_Q	KP	XAI JMS Queue

Table Name	Upgrade Action	Description
CI_XAI_JMS_Q_L	KP	XAI JMS Queue Language
CI_XAI_JMS_TPC	KP	XAI JMS Topic
CI_XAI_JMS_TPC_L	KP	XAI JMS Topic Language
CI_XAI_JNDI_SVR	KP	XAI JNDI Server
CI_XAI_JNDI_SVR_L	KP	XAI JNDI Server Language
CI_XAI_OPTION	KP	XAI Option
CI_XAI_RCVR	KP	XAI Receiver
CI_XAI_RCVR_CTX	KP	XAI Receiver Context
CI_XAI_RCVR_L	KP	XAI Receiver Language
CI_XAI_RCVR_RGRP	KP	XAI Receiver Rule Group
CI_XAI_RCVR_RSP	KP	XAI Receiver Response
CI_XAI_RGRP	KP	XAI Rule Group
CI_XAI_RGRP_ATT	KP	XAI Rule Group Attachment
CI_XAI_RGRP_L	KP	XAI Rule Group Language
CI_XAI_ROUTING	KP	XAI Routing
CI_XAI_RT_TYPE	KP	XAI Route Type
CI_XAI_RT_TYPE_L	KP	XAI Route Type Language
CI_XAI_RULE	KP	XAI Rule
CI_XAI_SENDER	KP	XAI Sender
CI_XAI_SENDER_L	KP	XAI Sender Language
CI_XAI_SNDR_CTX	KP	XAI Sender Context
F1_BKT_CONFIG	KP	Bucket Configuration
F1_BKT_CONFIG_L	KP	Bucket Configuration Language
F1_BKT_CONFIG_REL_OBJ	KP	Bucket Configuration Related Object
F1_BKT_CONFIG_VAL	KP	Bucket Configuration Value
F1_BKT_CONFIG_VAL_L	KP	Bucket Configuration Value Language
F1_BUS_OBJ_STATUS_RSN_CHAR	KP	Status Reason Characteristic
F1_EXTSYS_OUTMSG_PROF	KP	External System Outbound Message Type
F1_INSTALLATION	KP	Installation Option - Framework

Table Name	Upgrade Action	Description
F1_IWS_ANN_CHAR	KP	Inbound Web Service Annotation Characteristics
F1_IWS_ANN_TYPE_CHAR	KP	Inbound Web Service Annotation Type Characteristics
F1_IWS_SVC_ANN	KP	Inbound Web Service Link to Annotation
F1_IWS_SVC_CHAR	KP	Inbound Web Service Characteristics
F1_IWS_SVC_LOG	KP	Inbound Web Service Log
F1_IWS_SVC_LOG_PARM	KP	Inbound Web Service Log Parameter
F1_MAP_OVRD	KP	UI Map Override
F1_MD_DB_OBJ	KP	MD Database Object
F1_MST_CONFIG	KP	Master Configuration
F1_OUTMSG_TYPE	KP	Outbound Message Type
F1_OUTMSG_TYPE_L	KP	Outbound Message Type Language
F1_REQ_TYPE	KP	Request Type
F1_REQ_TYPE_L	KP	Request Type Language
F1_REQ_TYPE_LOG	KP	Request Type Log
F1_REQ_TYPE_LOG_PARM	KP	Request Type Log Parameters
F1_SVC_TASK_TYPE	KP	Service Task Type
F1_SVC_TASK_TYPE_CHAR	KP	Service Task Type Characteristics
F1_SVC_TASK_TYPE_L	KP	Service Task Type Language
F1_WEB_SVC	KP	Web Service Adapter
F1_WEB_SVC_CHAR	KP	Web Service Adapter Characteristics
F1_WEB_SVC_L	KP	Web Service Adapter Language
F1_WEB_SVC_LOG	KP	Web Service Adapter Log
F1_WEB_SVC_LOG_PARM	KP	Web Service Adapter Log Parameter
F1_WEB_SVC_OPERATIONS	KP	Web Service Adapter Operations
SC_USER	KP	User

Table Name	Upgrade Action	Description
SC_USER_CHAR	KP	User Characteristic
SC_USER_GROUP	KP	User Group
SC_USER_GROUP_L	KP	User Group Language
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

Appendix G

License and Copyright Notices

This section provides license and copyright information for the associated products. This includes the following:

Third-Party Products

The following sections provide notices and information about the third party products indicated.

Notice Concerning Usage of ANTLR

[The BSD License]

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

Neither the name of the author nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

\THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Notice Concerning Usage of Apache Software

The following files are covered under the Apache 2.0 license:

- bsf-2.4.0.jar
- castor-1.3.2-core.jar
- castor-1.3.2-xml-schema.jar
- castor-1.3.2-xml.jar
- cglib-2.2.jar
- commonj-3.7.1.jar
- commons-beanutils-core-1.8.3.jar
- commons-cli-1.1.jar
- commons-codec-1.6.jar
- commons-collections-3.2.1.jar
- commons-fileupload-1.2.2.jar
- commons-httpclient-3.0.1.jar

- commons-io-1.3.2.jar
- commons-lang-2.2.jar
- ehcache-core-2.5.2.jar
- joda-time-2.3.jar
- jackson-core-asl-1.9.2.jar
- jackson-jaxrs-1.9.2.jar
- jackson-mapper-asl-1.9.2.jar
- jackson-xc-1.9.2.jar
- log4j-1.2.17.jar
- serializer-2.7.1.jar
- stax2-2.1.jar
- stax2-api-3.0.4.jar
- velocity.1.4.jar
- wstx-asl-3.2.7.jar
- xalan-mod-2.7.1.jar
- xmlparserv2-11.1.1.3.0.jar

Apache License

Version 2.0, January 2004

http://www.apache.org/licenses/

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

You must give any other recipients of the Work or Derivative Works a copy of this License; and

You must cause any modified files to carry prominent notices stating that You changed the files; and

You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and

If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License. You may add Your own copyright statement to Your modifications and may provide additional or different

license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

Notice Concerning Usage of ASM

Copyright (c) 2000-2005 INRIA, France Telecom

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. Neither the name of the copyright holders nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES,

INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)

ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Notice Concerning Usage of Concurrent

All classes are released to the public domain and may be used for any purpose whatsoever without permission or acknowledgment. http://g.oswego.edu/dl/classes/EDU/oswego/cs/dl/util/concurrent/intro.html

Notice Concerning Usage of DOM4J

Copyright 2001-2010 (C) MetaStuff, Ltd. All Rights Reserved.

Redistribution and use of this software and associated documentation ("Software"), with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain copyright statements and notices. Redistributions must also contain a copy of this document.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. The name "DOM4J" must not be used to endorse or promote products derived from this Software without prior written permission of MetaStuff, Ltd. For written permission, please contact dom4j-info@metastuff.com.

4. Products derived from this Software may not be called "DOM4J" nor may "DOM4J" appear in their names without prior written permission of MetaStuff, Ltd. DOM4J is a registered trademark of MetaStuff, Ltd.

5. Due credit should be given to the DOM4J Project - http://dom4j.sourceforge.net

THIS SOFTWARE IS PROVIDED BY METASTUFF, LTD. AND CONTRIBUTORS

``AS IS'' AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT

NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND

FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL METASTUFF, LTD. OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Notice Concerning Usage of International Components for Unicode (ICU4J)

COPYRIGHT AND PERMISSION NOTICE

Notice Concerning Usage of Jaxen

Notice Concerning Usage of SLF4J

Notice Concerning Usage of Staxmate

Notice Concerning Usage of XMLPULL

Initial authors: Stefan Haustein and Aleksander Slominski

2001-12-12

Notice Concerning Usage of XStream

Notice Concerning Usage of YUI

PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Appendix H

Database Changes in Oracle Utilities Meter Data Management

This section specifies the database changes in this release of Oracle Utilities Meter Data Management. .

New Tables

Table_Name
D1_CONS_EXT_TYPE
D1_CONS_EXT_TYPE_ALG
D1_CONS_EXT_TYPE_CHAR
D1_CONS_EXT_TYPE_L
D1_CONS_EXT_TYPE_SP_TYPE
D1_GTT3
D1_GTT4
D1_SNAPSHOT_DL_CTRL
D1_SP_SNAP_DL
D1_SP_SNAP_DL_K
D1_SP_UNR_USG_SNAP_DL
D1_SP_UNR_USG_SNAP_DL_K
D1_SP_USG_SNAP_DL
D1_SP_USG_SNAP_DL_K
D1_SP_VEE_EXCP_SNAP_DL
D1_SP_VEE_EXCP_SNAP_DL_K
D1_US_MC
D1_US_USG_GRP_DL

New Views

None

Dropped Columns

None

Added Columns

Table_Name	Column_Name	Required
D1_ACTIVITY	ILM_ARCH_SW	N
D1_ACTIVITY	ILM_DT	N
D1_COMM_IN	ILM_ARCH_SW	N
D1_COMM_IN	ILM_DT	N
D1_COMM_OUT	ILM_ARCH_SW	N
D1_COMM_OUT	ILM_DT	N
D1_COMPL_EVT	ILM_ARCH_SW	N
D1_COMPL_EVT	ILM_DT	N
D1_DVC_EVT	ILM_ARCH_SW	N
D1_DVC_EVT	ILM_DT	N
D1_DVC_TYPE	DVC_SUB_CLASS_FLG	N
D1_GTT1	TIME_ZONE_CD	N
D1_INIT_MSRMT_DATA	ILM_ARCH_SW	N
D1_INIT_MSRMT_DATA	ILM_DT	N
D1_MSRMT	READING_VAL	N
D1_MSRMT	COMBINED_MULTIPLIER	N
D1_USAGE	ILM_ARCH_SW	N
D1_USAGE	ILM_DT	N
D1_VEE_EXCP	ILM_ARCH_SW	N
D1_VEE_EXCP	ILM_DT	N

Column Format Change

None

Primary Key Change

None

Added Indexes

Table_Name	Index_Name
D1_CONS_EXT_TYPE	D1C410P0
D1_CONS_EXT_TYPE	D1C410S0
D1_CONS_EXT_TYPE	D1C410S1
D1_CONS_EXT_TYPE_ALG	D1C414P0
D1_CONS_EXT_TYPE_CHAR	D1C412P0
D1_CONS_EXT_TYPE_CHAR	D1C412S1
D1_CONS_EXT_TYPE_L	D1C411P0
D1_CONS_EXT_TYPE_SP_TYPE	D1C413P0
D1_CONS_EXT_TYPE_SP_TYPE	D1C413S0
D1_DVC_CFG	D1M244S3
D1_DVC_EVT	D1T400S1
D1_GTT3	D1T406S0
D1_GTT4	D1T407S0
D1_INIT_MSRMT_DATA	D1T304S2
D1_SNAPSHOT_DL_CTRL	D1T433P0
D1_SP	D1M100S13
D1_SP_SNAP_DL	D1T434P0
D1_SP_SNAP_DL	D1T434S0
D1_SP_SNAP_DL_K	D1T435P0
D1_SP_UNR_USG_SNAP_DL	D1T438P0
D1_SP_UNR_USG_SNAP_DL	D1T438S0
D1_SP_UNR_USG_SNAP_DL_K	D1T439P0
D1_SP_USG_SNAP_DL	D1T436P0
D1_SP_USG_SNAP_DL	D1T436S0
D1_SP_USG_SNAP_DL_K	D1T437P0
D1_SP_VEE_EXCP_SNAP_DL	D1T440P0
D1_SP_VEE_EXCP_SNAP_DL	D1T440S0
D1_SP_VEE_EXCP_SNAP_DL_K	D1T441P0
D1_USAGE_REL	D1T316S0
D1_US_MC	D1M268S1
D1_US_MC	D1M319S0
D1_US_MC	D1M429P0

Table_Name	Index_Name
D1_US_USG_GRP_DL	D1T442P0

Dropped Indexes

Table_Name	Index_Name
D1_MSRMT	D1T298S1

Index Changes

None