Oracle® Healthcare Data Model

Operations Guide

11*g* Release 2 (11.2)

E18027-02

November 2010

ORACLE®

Oracle Healthcare Data Model Operations Guide, 11*g* Release 2 (11.2)

E18027-02

Contents

4 Working with an Oracle Healthcare Data Model Warehouse

Index

Preface

The *Oracle Healthcare Data Model Operations Guide* describes the tasks and procedures that must be performed after Oracle Healthcare Data Model is installed, and periodically afterwards to maintain useful performance. Since the needs of each Oracle Healthcare Data Model environment are unique, Oracle Healthcare Data Model is configurable so it can be modified to address each customer's needs.

Audience

The audience for the *Oracle Healthcare Data Model Operations Guide* includes the following:

- IT specialists, who maintain and adjust Oracle Healthcare Data Model. They are assumed to have a strong foundation in Oracle Database and PL/SQL, Analytic Workspace Manager, and BIEE.
- Database administrators, who will administer the data warehouse and the database objects that store the data. They are assumed to understand Intra-ETL, which is used to transfer data from one format to another, PL/SQL and the Oracle Database.

This document is also intended for business analysts, data modelers, data warehouse administrators, IT staff, and ETL developers.

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible to all users, including users that are disabled. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Accessibility standards will continue to evolve over time, and Oracle is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For more information, visit the Oracle Accessibility Program Web site at `http://www.oracle.com/accessibility/`.

Accessibility of Code Examples in Documentation

Screen readers may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.

Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit `http://www.oracle.com/support/contact.html` or visit `http://www.oracle.com/accessibility/support.html` if you are hearing impaired.

Related Documents

For more information about Oracle Oracle Healthcare Data Model, see the following documents in the Oracle Oracle Healthcare Data Model documentation set:

- *Oracle Healthcare Data Model Reference*
- *Oracle Healthcare Data Model Installation Guide*
- *Oracle Healthcare Data Model Release Notes*

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.

1

Introduction to Oracle Healthcare Data Model

This chapter introduces the Oracle Healthcare Data Model that gives you a jump-start for the design and implementation of a healthcare data warehouse solution:

- What is Oracle Healthcare Data Model?
- Oracle Healthcare Data Model Logical and Physical Models
- Oracle Products That Make Up the Oracle Healthcare Data Model
- Where Oracle Healthcare Data Model Fits in a Data Warehousing Project
- Prerequisite Knowledge for Customizers

> **See also:** The introduction to Oracle Healthcare Data Model in *Oracle Healthcare Data Model Reference*.

What is Oracle Healthcare Data Model?

You use the Oracle Healthcare Data Model to jump-start the design and implementation of an Oracle Healthcare Data Model data warehouse to quickly achieve a positive return on investment for your data warehousing and business intelligence project with a predictable implementation effort.

The Oracle Healthcare Data Model consists of logical and physical data models, intra-ETL that maps data from the Oracle Healthcare Data Model 3NF level to the aggregate level, sample reports, and the Oracle Interactive Dashboard using features of Oracle Business Intelligence Suite Enterprise Edition.

The Oracle Healthcare Data Model includes the following components:

- Logical model

  The logical model is introduced in "Logical Model" on page 1-2 and described in detail in *Oracle Healthcare Data Model Reference*.

- Physical model

  The physical model is introduced in "Physical Model" on page 1-5 and described in detail in *Oracle Healthcare Data Model Reference*.

- Intra-ETL database packages and SQL scripts to extract, transform, and load (ETL) data from one layer of the Oracle Healthcare Data Model to another.

  How to use these packages and scripts to populate a data warehouse based on the Oracle Healthcare Data Model is discussed in Chapter 3, "Populating an Oracle Healthcare Data Model Warehouse."

- Optional pre-defined data mining model.

Working with these data mining model is discussed in "Working with the Oracle Healthcare Data Model Data Mining Model" on page 4-5.

- Sample reports and dashboards using Oracle Business Intelligence Suite Enterprise Edition.

  Working with these reports are discussed in "Customizing the Reports Delivered with Oracle Healthcare Data Model" on page 4-1.

- DDL and installation scripts.

Oracle Healthcare Data Model Logical and Physical Models

The logical and physical models of the Oracle Healthcare Data Model have the following characteristics:

- Logical Model

  Healthcare industry-specific, 3rd Normal Form logical and physical relational models

- Physical Model

  Physical data model with:

  880+ tables and 11,460 non-key columns
  1380+ industry-specific measures
  20 pre-built clinical KPI's
  1 pre-built data mining model
  4 pre-built OLAP cubes

- 43 best-practice sample reports and dashboards

Logical Model

The Oracle Healthcare Data Model provides a predefined logical model. The logical data model defines the business entities and their relationships in order provide a clear understanding of the business and data requirements for the data warehouse.

For ease of use, the Oracle Healthcare Data Model is partitioned into three high-level logical categories which, in turn, are divided into functional subject areas. When designing your Healthcare data warehouse, you can implement some or all of these categories or subject areas.

- Supported Oracle Healthcare Data Model Categories and Subject Areas
- Common Patterns of Subject Areas

The logical data model is described in detail in *Oracle Healthcare Data Model Reference*.

Supported Oracle Healthcare Data Model Categories and Subject Areas

The logical categories supported by the Oracle Healthcare Data Model are:

- Common Infrastructure. For a description of the subject areas in this category, see Table 1–1 on page 1-3.
- Clinical Core. For a description of the subject areas in this category, see Table 1–2 on page 1-3.
- Financial and Billing. For a description of the subject areas in this category, see Table 1–3 on page 1-4.

Table 1–1 Subject Areas in the Common Infrastructure

Subject Area	Contains the entities and attributes that record the details of....
Encounter	The occurrence between a patient and healthcare participant(s) for the purpose of providing patient service(s) or assessing the health status of a patient. It contains data relevant to all aspects of an encounter such as interventions, results, diagnosis, admission types billing, charges, and reimbursements. This subject area is made up of a subtype to designate the type of encounter (that is to say, inpatient admission, office visit) and multiple relationship types to link the encounter to the activities, results and other clinical/financial occurrences in the encounter (for example, Concern Relationship Type, Facility Relationship Type, Intervention Relationship Type, and Observation Relationship Type). The relationship types are user defined.
Facility - Care Site	The geographical locations and the bricks and mortar qualities of the healthcare organization. It captures data relevant to the care site including such sites as state, region, city, county and all the way to unit, floor, room and bed. This subject area is made up of two subtypes; Facility and Care Site. The Facility subtype represents a licensed, certified, and/or accredited facilities that provides inpatient and outpatient services. Some examples of facilities are hospitals, nursing facilities, and ambulatory surgical facilities. A Care Site is a location within a building where care is delivered to the patient. For example an operating room, examination room, recovery room, bay, room, bed and so on.
Party	The direct and indirect participants in healthcare delivery. It captures data relevant to the care providers from both an individual and organizational perspective. A Party can be an individual or an organization. The Party primary entity is made up of two subtypes; Individual Party and Organization Party. The Individual Party subtype represents individual persons, while the Organization Party subtype represents any type of Organization consisting of zero, one, or more persons. Parties can further be described through the use of roles assumed by the parties in a given context. This is discussed in the following section. Party and its subtypes are modeled according to a generalization-specialization convention. Attributes that are common to the subtypes, either person or organization are modeled at the level of the Party supertype, such as phone number, email address, street address, and name. The subtypes of Party each have attributes that are specific for that subtype and not applicable to the others. For example, attributes for Individual Party subtype such as gender, and marital status may have no applicability to the Organization Party subtype.
Roles	(Not specifically a subject area.) A set of entities that are logical extensions to Party subtypes. The Role entities contain attributes that are specific to and necessary for defining a Party Role. The model supports a zero-to-many cardinality for the Party-Role relationship. For example, an Individual Party (Ind) may, at one time be in the role of a service provider, and at another time be in the role of a patient as well as employee and an Organization Party (Org) may be in the role of Service Provider an as a Legal Entity. The use of various roles, defined by the customer are associated to a Party to add role specific functionality as required. Roles are the building blocks that make up the unique characteristics of the individuals and organization in the delivery and receipt of healthcare. The following are examples of roles: Accreditation Organization (Org), Administrative Analyst (Party), Benefits Provider (Org), Business Unit (Org), Caregiver (Ind), Certification Organization (Org), Contact (Ind), Cost Center (Org), Dependent (Ind), Employee (Ind), Employer (Org), Government Organization (Org), Group Purchasing Organization (Org), Guarantor (Party), Guardian (Party), Insurer (Org), Legal Entity (Org), Licensing Organization (Org), Manufacturer (Org), Member (Ind), Patient (Ind), Payee (Party), Payer (Party), Pharmacy (Org), Recall Authority (Org), Service Provider (Party), and Vendor (Party).

Table 1–2 Subject areas in the Clinical Core Category

Subject Areas	Contains the entities and attributes that record the details of....
Advanced Directive	The receipt by a service provider of instructions from a patient regarding future medical care should he or she be unable to participate in medical decisions due to serious illness or incapacity. This subject area includes entities and attributes that support the identification of the recipient service provider and date and time specifics.
Concern	Any matter pertaining to a patient that may require a service provider's attention. Representative examples include any illness, condition, problem, diagnosis, injury, mental condition, disorder, environmental exposure (such as exposure to lead), or other concern.
Consent	The legal agreement or authorization given by a patient to a Service Provider to undergo a type of medical care or treatment. The entities and attributes in this subject area support the obtaining and providing of informed consent such as authorized interventions, related parties, service providers, and related consents among others.
Intervention	The actions taken to alter or achieve a patient's outcome by interfering or interceding. It contains data specific to all types of interventions from the most complex procedure and groups of procedures to simple bedside interventions. The entities in this subject area for which a relationship is provided include Intervention Concern, Intervention Consent, Intervention Service Provider, Intervention Substance, and Intervention Supply, among others.

Table 1–2 (Cont.) Subject areas in the Clinical Core Category

Subject Areas	Contains the entities and attributes that record the details of....
Observation	Clinical information collected directly or indirectly about a patient. Subtypes of Observation are: ADL, Adverse Event, Complication, Contraindication, Diagnostic Observation, Diagnostic Test Result, Discharge Note, Intolerance, Lab Result, Sign Symptom, Acuity Status, Physical Examination, Urinary, Skin Exam Results, Respiratory Test Result, Diagnostic Test Result, Psychiatric Assessment, Evaluation, Assessment, and Complication.
Order	The healthcare Order, a formal, pre or post documented instruction originated from an authorized party for initiation of actions directly or indirectly related to provision of care to patients. It contains the date relevant to the creation and fulfillment of the order as well as all relationships necessary to preserve the context of the Order. The entities found in the subject area support types of orders such as Laboratory, Radiology, Procedure, Observation, Substance Administration and Nursing Orders as well as relationship entities to Intervention, Service Provider, and Observation among others
Patient	The patient as the individual who receives medical care, attention or treatment from the healthcare service provider. It captures data relevant to the patient that are not represented in the Party subject area. For instance identification number, living arrangements, and birth order are attributes unique to Patient.
Patient History	The information communicated to the provider from the patient that identifies such things as the chief complaint, the history of the present illness, a review of systems, and the past, family and social history, and so on as the components of the patient history, and genetic history.
Specimen	The relevant to the collection and management of a medical patient's tissue, fluid, or other material derived from the patient used for laboratory analysis to assist differential diagnosis or staging of a disease process. The entities and attributes in this subject area support different types of specimens, collection method, anatomical site, specimen groups and specimen transactions.

Table 1–3 Subject areas contained in the Financial and Billing category

Subject Areas	Contains the entities and attributes that record the details of. .. .
Accounting and Financial	The financial condition and operating performance of the organization. The entities and attributes included in this subject area include but are not limited to Balance Sheet, Cash Flow Statement, Statement Of Income, Financial Statement, Journal, Ledger, and Ledger Account
Accounts Payable	Accounts of money the organization owes normally that arise from the purchase of products or services. The entities included in this subject area - but not limited to - are Invoice, Voucher, Payment, Return to Vendor.
Claims	An itemized statement of healthcare services and their costs provided by a Service Provider. The entities and attributes support the details of the submitted claims, requests for payment, appeals, and payments.
Contract	The legal agreement between a payer and a subscribing group or individual which specifies rates, performance covenants, the relationship among the parties, schedule of benefits and other pertinent conditions. This subject area contains entities and attributes that support capturing the details of the contract, section, clause, time period among others
Health Plan	The benefits received by a patient consisting of medical care (provided directly or through insurance or reimbursement) under any hospital or medical service policy, plan contract, or HMO contract offered by a health insurance company or a group health plan. Entities and attributes in this subject area include, but are not limited to drug, intervention and product coverage and charges, as well as Service Coverage and Provider Capitation. This subject area contains the entities and attributes that record the details of a specific health plan insurance policy written against a health plan offering.
HR and Payroll	Personnel management, including payroll, compliance with employment law, and human resources including managing the workforce. The entities and attributes included in this subject area include but are not limited to Benefit Plan, Dependent, Employee Certification, Employee Credential, Credential Type, Job, Position, and Employment Status
Inventory	Items stocked by an organization. Entities included are Inventory Transaction, Inventory Balance, Warehouse, Inventory Shipment, Material Stock Request, PAR request among others
Master Catalog	The product catalog, and the drug catalog among others.
Property and Equipment	The management of property, equipment, tooling and physical capital assets that are acquired and used to build, repair and maintain property and equipment. The entities and attributes included in this subject area include but are not limited to Asset, Asset Assessment and Condition, Depreciation and Location.
Purchasing	The organization attempts to acquire goods or services to accomplish the goals of the enterprise. The entities and attributes included in this subject area include but are not limited to Purchase Order, RFI, RFQ, RFP and Requisition.

Common Patterns of Subject Areas

Each subject area is made up of a primary entity, often represented by the name of the subject area, and related entities. Associations, relationships, or intersection entities link the entities, both within and across subject areas. Each type of link between entities describes the nature of the relationship and the cardinality. Intersection entities define and disambiguate a many-to-many relationship between entities. Each entity has the ability to represent concepts represented by coded standardized terminologies and local customer defined and developed terminologies or list of values, such as concern code, gender code, intervention code. These attributes have relationships to the coded entity that contains the following standard attributes: ID, CODE, NAME, and DESCRIPTION. This pattern is also followed for an attribute suffixed with TYPE such as Concern Transaction Type and Intervention Relationship Type. (Note, however, that the Inventory Transaction entity is an exception to this rule. This entity has attributes different from the Transaction pattern entity.)

Most subject areas share common patterns that perform similar functions across the model. These are patterns that are attached, via associations/relationships to the primary entity of a subject area to serve a specific purpose or perform a useful function.

Table 1–4 describes each of the common patterns.

Table 1–4 Common Patterns of Subject Areas

Pattern	Description
Related Party	Most relationships to Party are expressed by dedicated Roles such as Service Provider, Patient, and so on. Relationships to additional Party Roles can be created by using the Related Party Common pattern. For example, an implant device sales representative observing a surgical procedure can be represented using the Intervention Related Party entity. The same attributes are present in each Related Party entity across Oracle Healthcare Data Warehouse Foundation.
Transaction	This common pattern can be used to record user defined transactions or events that occur during the lifecycle of the primary entity of a subject area in the source system. The same attributes are present in each Transaction entity across Oracle Healthcare Data Warehouse Foundation.
Group	This common pattern is used to build a collection of instances of primary entities in a subject area. Groups can be named and then associated with one or more entities of the subject area. For example, OBSERVATION GROUP. The same attributes are present in each Group entity across Oracle Healthcare Data Warehouse Foundation.
Related <Primary Entity>	This common pattern is used to associate instances of a primary entity. It is a self-referential intersection entity that allows entities to be linked together in some manner. This pattern is used to support the management and definition of hierarchies. Other relationships are possible such as predecessor, successor, and sequential). The same attributes are present in each Related <Primary Entity> across Oracle Healthcare Data Warehouse Foundation.

Physical Model

Oracle Healthcare Data Model provides a predefined physical data model.

The physical data model includes the following types of tables and views:

- Oracle Healthcare Data Warehouse tables
- Reference tables
- Database sequences
- Derived tables
- OLAP dimension and cube views.

The physical data model is described in detail in *Oracle Healthcare Data Model Reference*.

Notes: When examining the predefined physical model, keep in mind the naming convention using DW (Data Warehouse) prefixes and suffixes to identify the types of tables and views:

DWC_ for intra-ETL control tables

DWD_ for derived tables

DWR_ for reference data tables

DWX_ for Facility-Shift and Code Repository (lookup) Cross Reference Table

HDM_ for Oracle Healthcare Data Model warehouse tables

Oracle Products That Make Up the Oracle Healthcare Data Model

Several Oracle technologies are involved in building the infrastructure for healthcare business intelligence:

- Oracle Database with OLAP, Data Mining and Partitioning Option
- Oracle Development Tools
- Oracle BI EE Presentation Tools

Oracle Database with OLAP, Data Mining and Partitioning Option

The Oracle Healthcare Data Model utilizes a complete Oracle technical stack. It leverages the following data warehousing features of the Oracle database: SQL model, compression, partitioning, advanced statistical functions, cube views, data mining, and online analytical processing (OLAP).

Tip: To save some money, you can consider using Oracle RAC and commodity hardware.

Oracle Development Tools

The following Oracle tools can be used to customize the predefined logical and physical models provided with the Oracle Healthcare Data Model, or to populate the target tables.

Table 1–5 Oracle Development Tools Used with Oracle Healthcare Data Model

Name	Use
Oracle SQL Data Modeler	To modify, customize, and extend the logical model
SQL Developer or SQL*Plus	To modify, customize, and extend database objects
Analytic Workspace Manager	To view, create, develop, and manage OLAP dimensional objects.

Oracle BI EE Presentation Tools

Oracle Business Intelligence Suite Enterprise Edition is a comprehensive suite of enterprise BI products that delivers a full range of analysis and reporting capabilities. You can use Oracle BI EE Answers and Dashboard presentation tools to customize the predefined sample dashboard reports that are provided with the Oracle Healthcare Data Model.

See: "Customizing the Reports Delivered with Oracle Healthcare Data Model" on page 4-1.

Where Oracle Healthcare Data Model Fits in a Data Warehousing Project

The Oracle Healthcare Data Model provides much of the data modeling work that you must do for a healthcare business intelligence solution. The Foundation Layer provides a solid foundation for a healthcare data warehouse. The Derived layer provides the infrastructure for creating KPI's, cube views, and reports.

Each healthcare organization is unique, and therefore the structure of the data warehouse will need to be different in order to match the needs of that unique healthcare environment. The Oracle Healthcare Data Model comes with a generic schema that requires modification. These modifications include, adding, deleting, modifying, or renaming tables and columns; or altering foreign keys, constraints, or indexes.

After the data warehouse is populated, you populate the derived layer and OLAP cube views to support the reporting requirements. The Oracle Healthcare Data Model includes a solid infrastructure for a range of reports.

Prerequisite Knowledge for Customizers

As discussed in "Oracle Products That Make Up the Oracle Healthcare Data Model" on page 1-6, the Oracle Healthcare Data Model uses much of the Oracle stack. Consequently, to successfully customize the Oracle Healthcare Data Model, you need:

- An understanding of the Oracle technology stack used for Oracle data warehouses, including Oracle Database and Data Warehouse, compression, partitioning, advanced statistical functions, Oracle OLAP (cube views), and, optionally, data mining.
- Hands-on experience using: Oracle database, PL/SQL; SQL DDL and DML syntax; Analytic Workspace Manager; Oracle SQL Developer; Oracle SQL Data Modeler; BI EE Administrator, Answers, and Dashboards.
- Experience developing and utilizing ETL to populate data warehouses.
- Experience in data analysis and data modeling.

2

Customizing the Oracle Healthcare Data Model

This chapter provides an overview of how you customize the Oracle Healthcare Data Model. It contains the following topics:

Overview: Customization Steps

Customizing the Oracle Healthcare Data Model involves the following tasks:

1. Perform fit-gap analysis as described in "Performing Fit-Gap Analysis" on page 2-2.
2. In a development environment, install an Oracle Healthcare Data Model instance.
3. Working in the copy you created in Step 2 and following the documentation you created when performing your fit-gap analysis, customize the Oracle Healthcare Data Model by making changes to its components. Document all of your changes. Make the changes in the following order:
    a. Logical model
    b. Logical to physical mappings
    c. Physical model. To determine which changes to make and the order in which to make these changes, determine dependences among objects by using the Metadata Browser.
    d. Intra-ETL. Keep in mind the issues discussed in Chapter 3, "Populating an Oracle Healthcare Data Model Warehouse".
4. In a test environment, make a copy of your customized version of the Oracle Healthcare Data Model.
5. In the test environment, following the documentation you created in Step 3, test the customized version of the Oracle Healthcare Data Model.
6. Roll the final customized version of the Oracle Healthcare Data Model out into production.

Note: Some logical and physical changes require specific changes to the Oracle Healthcare Data Model metadata. See "Modifying the Oracle Healthcare Data Model Metadata" on page 2-4 for more information.

Performing Fit-Gap Analysis

Fit-gap analysis is where you compare your information needs and healthcare business requirements with the structure that is available "out of the box" with the Oracle Healthcare Data Model. You identify any required functionality that is not included in the default schema, as well as other modifications that are necessary to meet your requirements.

The fit-gap analysis results in a customization report that provides a brief explanation of the adaptations and adjustments that will be required to customize the Oracle Healthcare Data Model to fit your healthcare organization requirements.

Note: Fit-gap analysis is a major undertaking, and normally requires a team performing multiple evaluations.

To perform the actual fit-gap analysis, take the following steps:

1. If previous evaluations have been performed, review the documentation from the previous phases, and if necessary add team members with the needed business and technical expertise.
2. Meet to review the data and map it into the Oracle Healthcare Data Model's schema. In other words, perform a comprehensive analysis of all of the source data in your OLTP systems and then compare your business with the Oracle Healthcare Data Model logical model to see how it maps to the Oracle Healthcare Data Model base and derived layer.
3. Produce a list of what people are going to try to do with the system (examples rather than models), and create use cases for appraising the functionality of Oracle Healthcare Data Model.

   Procedures are written based on the use cases. Keep in mind that deviations from the procedure can be useful, provided that functionality is not skipped.
4. Map your business procedures against the Oracle Healthcare Data Model functions, noting which processes are not available in Oracle Healthcare Data Model, or work differently in it. Be sure to check security requirements.
5. Determine the differences between your needs and the Oracle Healthcare Data Model's schema. Answer the following questions:
   - Which differences can you live with, and which must be reconciled?
   - What can you do about the differences you can't live with?
6. Based on the preceding steps, update the business process models, activity flow diagrams, entity object model, and object life cycle models to reflect the customized system.
7. Write the customization report, detailing what changes will be required to make the Oracle Healthcare Data Model's schema match your business needs. This includes any interfaces to existing systems, and additions and changes to the Oracle Healthcare Data Model.

8. Based on the customization report, update the Project Plan, and complete a phase section for the Logical Design phase.

Discovering the Oracle Healthcare Data Model Metadata

To customize the Oracle Healthcare Data Model model, you need to understand the dependency between each Oracle Healthcare Data Model component -- especially how the report KPIs are mapped to the physical tables and columns.

The Oracle Healthcare Data Model provides a tool called "Metadata Browser" that helps you discover the dependencies between Oracle Healthcare Data Model components, including how report KPIs are mapped to the physical tables and columns . The metadata browser is delivered as one Dashboard in Oracle Business Intelligence Suite Enterprise Edition.

There are four tabs in the Oracle Business Intelligence Suite Enterprise Edition Dashboard that is the Oracle Healthcare Data Model Metadata Browser:

- Measure-Entity tab
- Entity-Measure tab
- Program-Table tab
- Table-Program tab

Measure-Entity tab

On the Measure-Entity tab of the Metadata Browser, you can see the business area, measure name, measure description, responsible entity, and responsible attribute.

To browse the data, select the KPI that you are interested in.

Entity-Measure tab

Using the Entity-Measure tab of the Metadata Browser, you can discover the mappings between entities, attributes, supported measures, and calculations of the measures. You can discover information about particular entities and attributes.

For example, take the following steps to learn more about Encounter:

1. Select the Encounter entity.
2. Click GO.

Program-Table tab

Using the Program-Table tab of the Metadata Browser, you can browse for information on the intra-ETL mappings and report information. Take the following steps:

1. Select prompt program type (that is, Intra-ETL or report) and program name for sowing particular report or intra_ETL information.
2. Select GO.

For example, if you select a program type of **Report** for program named **Acute MI Patients Median Fibrinolytic Agent Administration Time**, that report is displayed.

Table-Program tab

By default, when you go to the Table-Program tab of the Metadata Browser, you see all tables used for all the reports.

To discover what reports use a particular table, you must move a particular table from the right pane to the left (Selected) pane.

For example, to see the reports that use the DWD_ENC_CASE table, take the following steps:

1. In the right pane of the Table-Program tab, select DWD_ENC_CASE.
2. Move it to the Selected list on the left by clicking on <, and click **OK**.
3. Select GO and the reports for DWD_ENC_CASE table are displayed.

Modifying the Oracle Healthcare Data Model Metadata

Changes to the logical and physical model require changes to the metadata. You make these changes by modifying various Oracle Healthcare Data Model tables. See Table 2–1, to identify what changes impact what table.

> **See also:** Discussion of the Metatdata Editor in *Oracle Healthcare Data Model Reference*.

Table 2–1 Tables to Update to Modify Metadata

Change	Table to Update	Example
Modify or add any entities or attributes and physical tables or columns mapped with those entities and attributes	DWR_MD_ENTY	Example 2–1, "Updating the DWR_MO_ENTY Table"
Modify any KPI	DWR_MD_KPI	Example 2–2, "Modifying the DWR_MD_KPI Table"
Modify the mapping between a KPI and a physical table	DWD_MD_REF_ ENTY_KPI	Example 2–3, "Modifying the DWD_MD_REF_ENTY_ KPI Table"
Modify an Intra-ETL mining programs	DWR_MD_PRG	Example 2–4, "Modifying the DWR_MD_PRG Table"

Example 2–1 Updating the DWR_MO_ENTY Table

Assume that you added a new entity named "ENT_NEW" with a new attribute named "ATTR_NEW", and the corresponding table and column name is "HDM_ENT_NEW" and "HDM_ATTR_NEW". Issue the following SQL statement to modify the DWR_ MD_ENTY table.

```
Insert into DWR_MD_ENTY (ENTITY_NAME,ATTRIBUTE_NAME,TABLE_NAME,COLUMN_NAME) values
('ENT_NEW',' ATTR_NEW ',' HDM_ENT_NEW ',' HDM_ATTR_NEW ');
```

Example 2–2 Modifying the DWR_MD_KPI Table

Assume that you modified an existing KPI, and changed its name from "Diabetes Mellitus: Diabetic Foot and Ankle Care, Peripheral Neuropathy Neurological Evaluation" to "TEST KPI". Issue the following SQL statement to modify the DWR_ MD_KPI table.

```
Update DWR_MD_KPI set KPI_NAME='TEST KPI' where KPI_ID=10
```

Example 2–3 Modifying the DWD_MD_REF_ENTY_KPI Table

Assume that you are now getting data from different table and column than originally designed (for example, "HDM_ENT_NEW" and "HDM_ATTR_NEW" for KPI 13). Issue the following SQL statement.

```
Insert into DWD_MD_REF_ENTY_KPI (KPI_NAME,PHY_AREA,BSNS_AREA,PHY_TAB_NAME,PHY_COL_
NAME,KPI_ID) values ('Pediatric End Stage Renal Disease : Influenza
Immunization','OHDM','CP','HDM_ENT_NEW','HDM_ATTR_NEW',13);
```

Example 2–4 Modifying the DWR_MD_PRG Table

Assume that you added one more data source(HDM_ENT_NEW) for DWD_ENC_CASE data population. Issue the following SQL statement.

```
Insert into DWR_MD_PRG (PRG_TYP,PRG_NAME,PHY_TAB_NAME,SB_PRG_TYP,PRG_MODEL_OR_
PATH,SB_PRG_DESC) values ('Intra-ETL Mapping','DWD_ENC_CASE','HDM_ENT_NEW','PL/SQL
Package','I','This program loads information about encounter - case
relationship');
```

3

Populating an Oracle Healthcare Data Model Warehouse

This chapter describes how to populate an Oracle Healthcare Data Model warehouse:

> **Note:** In general, the instructions in this chapter assume that after doing the fit-gap analysis described in "Performing Fit-Gap Analysis" on page 2-2, you have not identified or made any changes to the Oracle Healthcare Data Model logical or physical model. If you have made changes, you need to modify the ETL accordingly.

Pre-population Tasks

Before you populate your Oracle Healthcare Data Model warehouse for the first time, take the following steps:

1. Perform all of the post-installation tasks described in *Oracle Healthcare Data Model Installation Guide*.
2. Familiarize yourself with the ETL you use to populate an Oracle Healthcare Data Model warehouse as described in "Overview: The ETL for an Oracle Healthcare Data Model Warehouse" on page 3-2.
3. If the generic code names that are hard-coded in the ETL packages used by the KPI_FLW subprocess are not the names that are actually used by your healthcare organization, replace these codes as described in "Modifying the ETL to Change the Code Names" on page 3-6.

   > **See:** "Intra-ETL for Oracle Healthcare Data Model" on page 3-3 for information on the KPI_FLW subprocess.

4. Determine if you will be doing an initial load of your warehouse or an incremental load. Familiarize yourself with the process of performing which ever type of load you will be performing.
5. Familiarize yourself with how you can manage any errors that might occur during that initial load as described in "Managing Errors During Intra-ETL Execution" on page 3-4.
6. Estimate the size of your Oracle Healthcare Data Model warehouse. The size of Oracle Healthcare Data Model warehouse largely depends on the number of patients and encounters.

Overview: The ETL for an Oracle Healthcare Data Model Warehouse

As with any data warehouse, you use Extract, Transform, and Load (ETL) operations to populate an Oracle Healthcare Data Model data warehouse. You use the ETL to perform the initial load of an Oracle Healthcare Data Model data warehouse. After this initial load, you use the ETL to load new data into your Oracle Healthcare Data Model data warehouse regularly so that it can serve its purpose of facilitating business analysis.

These successive loads and transformations must be scheduled and processed in a specific order and will be determined by your business needs. Depending on the success or failure of the operation or parts of it, the result must be tracked and subsequent, alternative processes might be started. You can do a full incremental load of the relational tables and views, OLAP cubes, and the data mining model all at once, or you can refresh the data sequentially.

You perform ETL operations using different types of ETL that you execute in the following order:

1. Source-ETL that loads the HDM_ tables of the Oracle Healthcare Data Model warehouse. Source-ETL is not provided with the Oracle Healthcare Data Model. Guidelines for writing Source-ETL are provided in "Source-ETL for Oracle Healthcare Data Model" on page 3-2.
2. Intra-ETL that uses the data loaded by the source-ETL to populate the other components of the Oracle Healthcare Data Model. Sample Intra-ETL is provided with the Oracle Healthcare Data Model. The Intra-ETL is introduced in "Intra-ETL for Oracle Healthcare Data Model" on page 3-3.

Source-ETL for Oracle Healthcare Data Model

Source-ETL loads the HDM_ tables of the Oracle Healthcare Data Model warehouse.

Source-ETL is *not* provided with Oracle Healthcare Data Model. *You must design and write source-ETL processes yourself.*

Keep the following points in mind when designing and writing source-ETL:

- You can populate the calendar data by using the calendar population scripts provided with Oracle Healthcare Data Model and described in *Oracle Healthcare Data Model Reference*.
- Analyze all of the tables loaded by the source-ETL process before executing the intra-ETL.

Intra-ETL for Oracle Healthcare Data Model

Intra-ETL uses the data in the HDM_ tables that were populated by the source-ETL to populate the other tables, views, and models of the Oracle Healthcare Data Model.

Intra-ETL *is* provided with the Oracle Healthcare Data Model as the INTRA_ETL_FLW process flow which is designed using process flow of Oracle Warehouse Builder.

> **Note:** You do not have to use INTRA_ETL_FLW as the the Intra-ETL for Oracle Healthcare Data Model. You can write your own Intra-ETL.

INTRA_ETL_FLW is a complete Intra-ETL process composed of subprocess flows to populate the dimensions, derived tables, and KPI fact tables along with two other subprocess flows for Oracle OLAP and Data Mining. This process flow respects the dependency of each individual program and executes the programs in the proper order.

The INTRA_ETL_FLW is the complete Intra ETL process designed using process flow of Oracle Warehouse Builder, and is composed of the following individual subprocesses:

- **DIM_FLW** - This subprocess flow triggers the ETLs for populating dimension tables.
- **DRVD_FLW** - This subprocess flow triggers the ETLs for populating derived fact tables based on the content of the 3NF HDM tables.
- **KPI_FLW** - This subprocess flow triggers the ETLs for populating the fact tables based on the content of the 3NF HDM tables, dimension table and derived fact tables.
- **OLAP_FLW** - This subprocess flow triggers the execution of the OLAP package which loads data from data warehouse to Oracle Healthcare Data Model analytic workspace and prepares the analytic workspace for reporting. It reads ETL parameters from DWC_ETL_PARAM and DWC_OLAP_ETL_PARAM tables which are documented in *Oracle Healthcare Data Model Reference*.
- **MINING_FLW** - This subprocess flow triggers the data mining model.

You can install Oracle Health Care Data Model Intra-ETL as a project in Oracle Warehouse Builder as described in *Oracle Healthcare Data Model Installation Guide*. Once installed, you can execute the intra-ETL from Oracle Warehouse Builder.

Steps: Populating a Oracle Healthcare Data Model Warehouse

After you install INTRA_ETL_FLW as a project in Oracle Warehouse Builder as described in *Oracle Healthcare Data Model Installation Guide*, take the following to populate the Oracle Healthcare Data Model warehouse:

1. Ensure that the HDM_ tables in the Oracle Healthcare Data Model warehouse are up-to-date. If necessary, execute the source-ETL that you have written using the guidelines given in "Source-ETL for Oracle Healthcare Data Model" on page 3-2.
2. Update the parameters in the `DWC_ETL_PARAM` and the `DWC_OLAP_ETL_PARAM` control tables following the guidelines *Oracle Healthcare Data Model Reference*. (For an initial load, remember to specify `C` for the value of the BUILD_METHD column of the DWC_OLAP_ETL_PARAM table.)

See: *Oracle Healthcare Data Model Reference* for complete information on these control tables.

3. Ensure that the repository user (for example, OHDM_ETL) has the EXECUTE privilege for all the packages that are executed as part of OHDM_INTRA_ETL Process Flow.

   These packages are listed on the Transformation node of the data module. Following the examples in this chapter, they are listed on the Transformation node of OHDM_INTRA_ETL.

4. Within Oracle Warehouse Builder, execute the INTRA_ETL_FLW process.

Managing Errors During Intra-ETL Execution

This topic discusses how you can identify and manage errors during Intra-ETL execution. It contains the following topics:

Monitoring the Execution of the Intra-ETL Process

Two control tables in the `ohdm_sys` schema, `DWC_INTRA_ETL_PROC` and `DWC_INTRA_ETL_ACTVTY`, monitor the execution of the Intra-ETL process. These tables are documented in *Oracle Healthcare Data Model Reference*.

Each normal run (as opposed to an error-recovery run) of a separate Intra-ETL execution performs the following steps:

1. Inserts a record into the `DWC_INTRA_ETL_PROC` table with a monotonically increasing system generated unique process key, `SYSDATE` as process start time, `RUNNING` as the process status, and an input date range in the FROM_DT_ETL and TO_DT_ETL columns.
2. Invokes each of the individual Intra-ETL programs in the appropriate order of dependency. Before the invocation of each program, the procedure inserts a record into the Intra-ETL Activity detail table, `DWC_INTRA_ETL_ACTVTY`, with a system generated unique activity key in ACTVTY_KEY, the process key value corresponding to the Intra-ETL process in PRCSS_KEY, an individual program name as the ACTVTY_NAME, a suitable activity description in ACTVTY_DESC, `SYSDATE` as the value of activity start time, and `RUNNING` as the activity status
3. Updates the corresponding record in the `DWC_INTRA_ETL_ACTVTY` table for the activity end time and activity status after the completion of each individual ETL program (either successfully or with errors. For successful completion of the activity, the procedure updates the status as `'COMPLETED-SUCCESS'`. When an error occurs, the procedure updates the activity status as `'COMPLETED-ERROR'`, and also updates the corresponding error detail in the ERR_DTL column.
4. Updates the record corresponding to the process in the `DWC_INTRA_ETL_PROC` table for the process end time and status, after the completion of all individual intra-ETL programs. When all the individual programs succeed, the procedure updates the status to `'COMPLETED-SUCCESS'`, otherwise it updates the status to `'COMPLETED-ERROR'`.

You can monitor the execution state of the Intra-ETL, including current process progress, time taken by individual programs, or the complete process, by viewing the contents of the and DWC_INTRA_ETL_ACTVTY tables corresponding to the maximum process key. Monitoring can be done both during and after the execution of the Intra-ETL procedure.

Recovering an Intra ETL Process

To recover an intra-ETL process:

1. Identify the errors by looking at the corresponding error details that are tracked against the individual programs in the DWC_INTRA_ETL_ACTVTY table.

   See: *Oracle Healthcare Data Model Reference* for detailed information on the DWC_INTRA_ETL_ACTVTY table.

2. Correct the causes of the errors.
3. Re-run the Intra-ETL.

The Intra_ETL has the intelligence of identifying whether it is a normal run or recovery run by referring the DWC_INTRA_ETL_ACTVTY table. During a recovery run, the Intra_ETL executes only the necessary programs.

In this way, the Intra-ETL error recovery is almost transparent, without involving the Data Warehouse or ETL administrator. The administrator only needs to take correct the causes of the errors and re-invoke the Intra-ETL process once more. The Intra-ETL process identifies and executes the programs that generated errors.

Troubleshooting Intra-ETL Performance

To troubleshoot the Intra-ETL performance:

- Check the execution plan as described in "Checking the Execution Plan" on page 3-5
- Monitor parallel DML executions as described in "Monitoring PARALLEL DML Executions" on page 3-6

Checking the Execution Plan

Use SQLDeveloper or other tools to view the package body of the code.

For example, take the following steps to examine HDW_ANL_CLIN_KPI2_MAP.

1. Copy out the main query statement from code viewer.

   To do this, you copy from the beginning to end of the query, which is right above another "COMMIT".

2. In SQLDeveloper worksheet, issue the following command to turn on the parallel DML:

   ```
   Alter session enable parallel dml;
   ```

3. Paste the main query statement into another SQLDeveloper worksheet and view the execution plan by clicking F6.

   Carefully examine the execution plan to make the mapping runs according to an valid plan.

Monitoring PARALLEL DML Executions

Check that you are running mapping in parallel mode by executing the following SQL statement to count the executed "Parallel DML/Query" statement

```
column name format a50
column value format 999,999
SELECT NAME, VALUE
FROM GV$SYSSTAT
WHERE UPPER (NAME) LIKE '%PARALLEL OPERATIONS%'
  OR UPPER (NAME) LIKE '%PARALLELIZED%'
  OR UPPER (NAME) LIKE '%PX%'
;
```

If you are running mapping in parallel mode, you should see "DML statements parallelized" increased by 1 every time the mapping was invoked. If not you do not see this increase, then the mapping was not invoked as "parallel DML".

If you see "queries parallelized" increased by 1 (one) instead, then typically it means that the SELECT statement inside of the INSERT was parallelized but that INSERT itself was not.

See also: "Altering Parallelization" on page 4-7

Troubleshooting Data Mining Model Creation

Once the data mining model is created, check the error log in `DWC_INTRA_ETL_ACTVTY` table which is documented in *Oracle Healthcare Data Model Reference*.

For example, execute the following code.

```
set line 160
col ACTVTY_NAME format a30
col ACTVTY_STATUS format a20
col error_dtl format a80
select actvty_name, actvty_status,  error_dtl from dwc_intra_etl_actvty;
```

If all models are created successfully, the actvty_status will be `"COMPLETED-SUCCESS"`. If the actvty_status is `"COMPLETED-ERROR"` for a certain step, please check the ERR_DTL column, and fix the problem accordingly.

A common error message is shown below.

Message not available ... [Language=ZHS]

'ZHS' is a code for a language. The language name it relates to can appear as different name depending on the database environment. This error happens when `DWC_MESSAGE.LANG` does not contain messages for the current language.

Check the values in the `DWC_MESSAGE` table and, if required, update to the language code specified by the Oracle session variable `USERENV('lang')`.

Modifying the ETL to Change the Code Names

Some generic code names are hard-coded in the ETL packages used by the KPI_FLW subprocess. You can replace these codes with the code names actually used in your healthcare organization by changing the values of the names in the HDM_CD_REPOSITORY table.

To make these changes in an ETL package, use any edit tool to open package and then, change the code name as illustrated by Example 3–1, "Changing Code Names". This

examples changes names in HDW_ANL_CLIN_KPI1_MAP. The places in the code where you would put in your code names are marked by the comment `----Change to your own code`.

Example 3–1 Changing Code Names

```
CREATE OR REPLACE PACKAGE HDW_ANL_CLIN_KPI1_MAP


AS
  PROCEDURE LOADFACT(p_etl_strt_date in DATE default null,p_etl_end_date in DATE default null,p_
etl_strt_date_id NUMBER default null,p_etl_end_date_id NUMBER default null);
  FUNCTION LOADETL (p_etl_strt_date in DATE default null,p_etl_end_date in DATE default null,p_etl_
activity_name IN VARCHAR2,p_etl_process_no IN NUMBER DEFAULT NULL) RETURN VARCHAR2;

END HDW_ANL_CLIN_KPI1_MAP;
/

create or replace PACKAGE BODY HDW_ANL_CLIN_KPI1_MAP
AS
. . .
where
 DWD_Enc.ENC_END_DATE_ID between p_etl_strt_date_id  and p_etl_end_date_id
 And DWD_Enc.Enc_End_DT between p_etl_strt_date and p_etl_end_date
 ),
flt_enc as (
select /*+materialize */ enc_id from (
select enc_id from enc where
Enc_Strt_Dt - interval '18' year >= Prty_Bir_Dt
and
ENC_Cd_Nm  in (
    'Evaluation and Management New Outpatient Visit', --Change to your own code
    'Evaluation and Management Established Outpatient Visit', --Change to your own code
    'Initial Nursing Facility Care', --Change to your own code
    'Subsequent Nursing Facility Care', --Change to your own code
    'An Assisted Living Facility or Other Domicile', --Change to your own code
    'Home services ?new patient', --Change to your own code
    'Home services - established patient' ) --Change to your own code
    minus
select /*+ ordered */
       Enc.Enc_Id
    from enc Enc
    Inner join DWD_ENC_ORD enc_ord
    On enc_ord.Enc_Id=Enc.Enc_Id
     --And Enc.ENC_END_DATE_ID between p_etl_strt_date_id  and p_etl_end_date_id
     --And Enc.Enc_End_DT between p_etl_strt_date and p_etl_end_date
    Left outer join HDM_Subadmn_ord_subst saos
    On saos.ord_id=enc_ord.ord_id
    Left outer join DWX_Code_Rep_XREF  Saos_CR_Xref
    On Saos_CR_XREF.Cd_Id = SAOS.Subst_Cd_id
    And Saos_CR_XREF.Cd_Typ_Cd = '1032' --Change to your own code
    --and Saos_CR_XREF.Cd_Typ_Nm = 'Substance Code' --Change to your own code
    Left outer join HDM_Ord_Rsn ord_rsn
    On ord_rsn.ord_id=enc_ord.ord_id
    Left outer join DWX_Code_Rep_XREF  OrdRsn_CR_Xref
    On OrdRsn_CR_Xref.Cd_Id = Ord_Rsn.Ord_Rsn_Cd_Id
    and OrdRsn_CR_Xref.Cd_Typ_Cd = '1030'--Change to your own code
    -- and OrdRsn_CR_Xref.Cd_Typ_Nm = 'Order Reason Code'--Change to your own code
    Where
     Enc_Ord.Ord_Subtyp_Nm in ('Substance Administration Order' ) --Change to your own code
```

```
    And Enc_Ord.Ord_StCd_Nm in ('Not Prescribed') --Change to your own code
    And
    (
     Saos_CR_Xref.Cd_Desc in ('Beta Blocker') --Change to your own code
       OR
     Enc_Ord.Ord_Cd_desc in ('Beta Blocker') --Change to your own code
    )
    AND OrdRsn_CR_Xref.Cd_Nm in ('Not Eligible for Beta Blocker Therapy') --Change to your own code
)
),
Numerator as
(
select /*+materialize */ enc_id from
(
 SELECT enc21.enc_id
        From enc Enc21
        Inner Join DWD_ENC_ORD EO21
        On Enc21.Enc_Id = EO21.Enc_Id
        Left outer join HDM_Subadmn_Ord_Subst saos21
        On saos21.ord_id=EO21.ord_id
        Left outer join DWX_Code_Rep_XREF  Saos_CR_Xref21
        On Saos_CR_XREF21.Cd_Id = SAOS21.Subst_Cd_id
        And Saos_CR_XREF21.Cd_Typ_Cd = '1032'--Change to your own code
        --and Saos_CR_XREF21.Cd_Typ_Nm = 'Substance Code'
        WHERE
         (
          EO21.Ord_Subtyp_Nm in ('Substance Administration Order' ) --Change to your own code
          AND EO21.Ord_StCd_Nm in ('Prescribed') --Change to your own code
          AND
          (
            Saos_CR_Xref21.Cd_Desc in ('Beta Blocker') --Change to your own code
            OR
            EO21.Ord_Cd_Desc in ('Beta Blocker') --Change to your own code
            )
          )
union
      SELECT EI22.enc_id
      From  DWD_ENC_INTVN EI22
      Inner Join HDM_SUBADMN Sa22
      On EI22.Intvn_Id=Sa22.Intvn_Id
      Inner join DWX_Code_Rep_XREF  Sa_CR_Xref22
        On Sa_CR_XREF22.Cd_Id = SA22.SUBST_TYP_cd_ID
        And Sa_CR_XREF22.Cd_Typ_Cd = '1038'--Change to your own code
      --and Saos_CR_XREF21.Cd_Typ_Nm = 'Substance Type'--Change to your own code
      WHERE
          EI22.Intvn_Subtyp_Nm in ('Substance Administration') --Change to your own code
      AND    EI22.Intvn_Cd_Desc in ('Beta Blocker' ) --Change to your own code
      AND    Sa_CR_XREF22.Cd_Nm  in ('Medication') --Change to your own code

 union
      SELECT enc23.enc_id
      FROM    enc Enc23
      Inner Join HDM_PT_HX Pt_Hx23
      On Enc23.Pt_Id = Pt_Hx23.Pt_Id
      Inner Join DWX_CODE_Rep_XREF Hxsub_CR_Xref23
      On Hxsub_CR_Xref23.Cd_Id = Pt_Hx23.Pt_Hx_Subtyp_Id
      and Hxsub_CR_Xref23.Cd_Typ_Cd = '1046'--Change to your own code
      --and Hxsub_CR_Xref23.Cd_Typ_Nm = 'Patient History Subtype'--Change to your own code
      Inner Join DWX_CODE_Rep_XREF Hx_CR_Xref23
      On Hx_CR_XREF23.Cd_Id = Pt_Hx23.Pt_Hx_Cd_Id
```

```
      and Hx_CR_XREF23.Cd_Typ_Cd = '1047'--Change to your own code
      --and Hx_CR_XREF23.Cd_Typ_Nm = 'Patient History Code'--Change to your own code
      Inner Join HDM_Subst_Hx Sub_Hx23
      On Sub_Hx23.Pt_Hx_Id = Pt_Hx23.Pt_Hx_Id
      Inner Join DWX_CODE_Rep_XREF SubstHx_CR_Xref23
       On SubstHx_CR_Xref23.Cd_Id = Sub_Hx23.Subst_Catg_Id
      and SubstHx_CR_Xref23.Cd_Typ_Cd = '1082'--Change to your own code
      --and SubstHx_CR_Xref23.Cd_Typ_Nm = 'Substance Category Code'--Change to your own code
      WHERE
       Hxsub_CR_Xref23.Cd_Nm in ('Substance History') --Change to your own code
        AND SubstHx_CR_Xref23.Cd_Desc in ('Beta Blocker') --Change to your own code
      AND Hx_CR_XREF23.Cd_Nm in ('Current Medication') --Change to your own code
  )
)
SELECT
  Distinct
  Denominator.Enc_End_DATE_ID,
  Denominator.Enc_End_TOD_ID,
  Denominator.Fac_Mbr_Id,
  Denominator.Org_Mbr_Id,
  Denominator.AgeBnd_Id,
  Denominator.Shft_Id,
  Denominator.Pt_ID,
  Denominator.SvcprvPrct_Id,
  Denominator.Enc_Id,
  1 as KPI_DENM,
  Case WHEN Numerator.Enc_ID is not NULL then 1 else 0 end as KPI_NUMN,
  NULL as KPI_DENM_VAL,
  NULL as KPI_NUMN_VAL,
  1 as KPI_ID,
  UID AS CREATED_BY_ID,
  UID AS CHANGED_BY_ID,
  SYSDATE AS INSERT_DT,
  SYSDATE AS UPDATE_DT,
  -1 AS TENANT_ID
From
--Denominator:
(
  Select /*+ordered */
  Distinct
    DWD_Enc.Enc_End_DATE_ID,
    DWD_Enc.Enc_End_TOD_ID,
    DWD_Enc.Fac_Mbr_Id,
    DWD_Enc.Org_Mbr_Id,
    DWD_Enc.AgeBnd_Id,
    DWD_Enc.Shft_Id,
    DWD_Enc.Pt_ID,
    DWD_Enc.SvcprvPrct_Id,
    DWD_Enc.Enc_Id
   FROM
    enc DWD_Enc
  Inner join DWD_ENC_CNRN Enc_Cnrn
  On DWD_Enc.Enc_Id=Enc_Cnrn.Enc_Id
  And DWD_Enc.Enc_Strt_Dt - interval '18' year >= DWD_Enc.Prty_Bir_Dt
  Inner join DWD_ENC_OBSV  Enc_Obsv
  On DWD_Enc.Enc_Id=Enc_Obsv.Enc_Id
  Where   Enc_Cnrn.Cnrn_Subtyp_Nm  in ('Diagnosis') --Change to your own code
  And     Enc_Cnrn.Cnrn_Cd_Nm  in ('Heart Failure') --Change to your own code
  AND     Enc_Cnrn.Cnrn_Cd  in ('425.2','425.3') --Change to your own code
  And     Enc_Obsv.Obsv_Cd_Nm in ('LVEF') --Change to your own code
```

```
  And
(
  (
Enc_Obsv.Obsv_Val_Typ_Nm in ('Numeric' )
    And     Enc_Obsv.Obsv_Val_Nmeric < 40
    And     Enc_Obsv.Obsv_Val_UOMCd_Nm  in ('%')
  )
  Or
  (
    Enc_Obsv.Obsv_Val_Typ_Nm in ('Code')
    And     Enc_Obsv.Obsv_Val_Cd_Nm in ('Moderately or Severely Depressed LVF') --Change to your
own code
  )
  Or
    (
    Enc_Obsv.Obsv_Val_Typ_Nm in ('Code')
    And     Enc_Obsv.Obsv_Val_Cd_Nm in ('Depressed LVF') --Change to your own code
    And     Enc_Obsv.Obsv_Sevrty_Cd_Nm in ('Moderate','Severe') --Change to your own code
  )
)
-- Denominator Exclusion Criteria
AND DWD_Enc.Enc_Id  in
  ( select enc_id from flt_enc
    )
) Denominator  -- Denom query alias
--Numerator Query:
LEFT OUTER JOIN Numerator
ON Denominator.Enc_Id= Numerator.Enc_Id
;

COMMIT;

. . .
    IF l_error_dtl IS NULL THEN l_status := 'SUCCESS';
    ELSE l_status := 'ERROR';
    END IF;
    RETURN l_status;
    END LOADETL;


End HDW_ANL_CLIN_KPI1_MAP;
   /
```

4

Working with an Oracle Healthcare Data Model Warehouse

In general, you manage an Oracle Healthcare Data Model data warehouse in much the same way that you manage any other data warehouse. For example, after the initial data load, you perform incremental data loading.

Data loading is discussed in Chapter 3, "Populating an Oracle Healthcare Data Model Warehouse". This chapter discusses aspects of working with a data warehouse that are specific to an Oracle Healthcare Data Model data warehouse.

This chapter includes the following topics:

Customizing the Reports Delivered with Oracle Healthcare Data Model

Sample reports and dashboards are delivered with Oracle Healthcare Data Model. These sample reports illustrate the analytic capabilities provided with Oracle Healthcare Data Model -- including the OLAP and data mining capabilities.

The sample reports are based on Oracle Business Intelligence Suite Enterprise Edition which is a comprehensive suite of enterprise BI products that delivers a full range of analysis and reporting capabilities. Thus, the reports also illustrate the ease with which you can use Oracle Business Intelligence Suite Enterprise Edition Answers and Dashboard presentation tools to create useful reports.

You use Oracle Business Intelligence Suite Enterprise Edition tools to customize and troubleshoot the execution of reports, as described in:

- "Tools for Customizing Reports" on page 4-2
- "Troubleshooting Reporting Performance" on page 4-2

> **See:** Oracle Healthcare Data Model Reference for detailed information on the sample reports.

Note: The sample reports are based on Oracle Business Intelligence Suite Enterprise Edition 10*g* (10.1.3.4.1). and will work with Oracle Business Intelligence Suite Enterprise Edition 10*g*, or higher within the 10*g* release.

Note: The reports and dashboards that are used in examples and delivered with Oracle Healthcare Data Model are provided only for demonstration purposes. They are not supported by Oracle.

Tools for Customizing Reports

You can use Oracle Business Intelligence Suite Enterprise Edition Answers and Dashboard presentation tools to customize the predefined sample dashboard reports:

- **Oracle BI Answers**. Provides end user ad hoc capabilities in a pure Web architecture. Users interact with a logical view of the information -- completely hidden from data structure complexity while simultaneously preventing runaway queries. Users can easily create charts, pivot tables, reports, and visually appealing dashboards.
- **Oracle BI Interactive Dashboards**. Provide any knowledge worker with intuitive, interactive access to information. The end user can be working with live reports, prompts, charts, tables, pivot tables, graphics, and tickers. The user has full capability for drilling, navigating, modifying, and interacting with these results.

Troubleshooting Reporting Performance

Take the following actions to identify problems generating a report created using Oracle Business Intelligence Suite Enterprise Edition:

1. In the (Online) Oracle BI Administrator Tool, select **Manage,** then **Security**, then **Users**, and then **ohdm**.

   Ensure that the value for **Logging level** is 7.

2. Open the Oracle Healthcare Data Model Repository, select **Manage**, and then **Cache**.
3. In the right-hand pane of the Cache Manager window, select all of the records, then right-click and select **Purge**.
4. Run the report or query that you want to track the SQL log.
5. Open the query log file (`NQQuery.log`) under `OracleBI\server\Log`.

   The last query SQL is the log of the report you have just run. If an error was returned in your last accessed report, there will be an error at the end of this log.

For example:

```
Error Codes: OPR4ONWY:U9IM8TAC:OI2DL65P State: HY000. Code: 10058. [NQODBC] [SQL_
STATE: HY000] [nQSError: 10058] A general error has occurred. [nQSError: 17001]
Oracle Error code: 942, message: ORA-00942: table or view does not exist at OCI
call OCIStmtExecute. [nQSError: 17010] SQL statement preparation failed.
(HY000)SQL Issued: SELECT "Organization Dimension".Enterprise saw_0, "Facility
Dimension"."Caresite L2" saw_1, "Time Encounter Dimension"."Month" saw_2, "Time
Encounter Dimension".Quarter saw_3, "Kpi Dimension".KPI saw_4, "Encounter Kpi
Cube"."Represent the correct calculation needed for each of the Kpis"*100 saw_5,
AGGREGATE(saw_5 BY saw_1, saw_2, saw_3, saw_4), AGGREGATE(saw_5 BY saw_0, saw_2,
```

```
saw_3, saw_4) FROM OHDM WHERE ("Time Encounter Dimension"."Year" = 'CY 2010') AND
("Kpi Dimension".KPI = 'Heart Failure: Beta-Blocker Therapy for Left Ventricular
Systolic Dysfunction') ORDER BY saw_0, saw_1, saw_2, saw_3, saw_4, saw_5

Meaning: This error occurs when you did not create OLAP report view in DB

Action: Execute "ohdm_olap_report_view.sql" under $ORACLE_
HOME/ohdm/pdm/relational/ddl
```

Writing Your Own Queries and Reports

The ohdm_sys schema defines the relational tables and views in Oracle Healthcare Data Model. You can use any SQL reporting tool to query and report on these tables and views.

> **See:** Oracle Healthcare Data Model Reference for more information on Oracle Healthcare Data Model relational tables and views.

Oracle Healthcare Data Model also supports On Line Analytic Processing (OLAP) reporting through the use of OLAP cubes defined in the ohdm_sys schema. You can query and write reports on OLAP cubes by using SQL tools to query the views that are defined for the cubes or by using OLAP tools to directly query the OLAP components.

> **See:** Oracle Healthcare Data Model Referencefor more information on Oracle Healthcare Data Model OLAP cubes and the relational views by which you can access them, and *Oracle OLAP User's Guide* for information on how to create queries and reports on OLAP cube data.

Example 4–1 Creating a Relational Query

EXAMPLE:
Assume that you want to determine the number of inpatient encounters starting in March 2010 where the patient status is discharged and which had observations of 'Catheter in Place'.

The following SQL executes this query. (Note the comments which begin with "--".)

```
SELECT *

FROM

--InPatient Encounters

HDM_ENC Encounter

INNER JOIN HDM_CD_REPOSITORY EncSubTyp_Repos ON Encounter.ENC_SUBTYP_ID =
EncSubTyp_Repos.CD_ID

INNER JOIN HDM_CD_REPOSITORY_CD_TYP EncSubTyp_ReposTyp ON EncSubTyp_Repos.CD_ID =
EncSubTyp_ReposTyp.CD_ID

INNER JOIN HDM_CD_TYP EncSubTyp_Typ ON EncSubTyp_ReposTyp.TYP_ID = EncSubTyp_
Typ.TYP_ID

-- AND Patients discharged status = discharged

INNER JOIN HDM_CD_REPOSITORY EncPtCod_Repos ON Encounter.PT_DC_STCD_ID = EncPtCod_
Repos.CD_ID
```

```
INNER JOIN HDM_CD_REPOSITORY_CD_TYP EncPtCod_ReposTyp ON EncPtCod_Repos.CD_ID =
EncPtCod_ReposTyp.CD_ID

INNER JOIN HDM_CD_TYP EncPtCod_Typ ON EncPtCod_ReposTyp.TYP_ID = EncPtCod_Typ.TYP_
ID

--Corresponding Observations for the above Encounters

INNER JOIN HDM_ENC_OBSV EncObsv ON Encounter.ENC_ID = EncObsv.ENC_ID

INNER JOIN HDM_OBSV Obsv ON EncObsv.OBSV_ID = Obsv.OBSV_ID

--Above Observations with code = 'Catheter in Place'

INNER JOIN HDM_CD_REPOSITORY ObsvCd_Repos ON Obsv.OBSV_CD_ID = ObsvCd_Repos.CD_ID

INNER JOIN HDM_CD_REPOSITORY_CD_TYP ObsvCd_ReposTyp ON ObsvCd_Repos.CD_ID =
ObsvCd_ReposTyp.CD_ID

INNER JOIN HDM_CD_TYP ObsvCd_Typ ON ObsvCd_ReposTyp.TYP_ID = ObsvCd_Typ.TYP_ID

WHERE

EncSubTyp_Typ.TYP_NM = 'Encounter Subtype' AND EncSubTyp_Repos.CD_NM = 'Inpatient'

AND EncPtCod_Typ.TYP_NM = 'Patient Discharge Status Code' AND EncPtCod_Repos.CD_NM
In ('Discharged to Home' ,'Discharged/Transferred to Another Inpatient Care'
,'Discharged/Transferred to Federal Health Care Facility' ,'Discharged/Transferred
to Hospice')

AND to_date(to_char(Encounter.STRT_DT,'MON-YY'),'MON-YY') = to_date ('MAR-10',
'MON-YY')

AND ObsvCd_Typ.TYP_NM = 'Observation Code' AND ObsvCd_Repos.CD_NM = 'Catheter in
Place'
```

Example 4–2 Median Time From Hospital Arrival to PCI

Assume that you want to know the median time from hospital arrival to primary percutaneous coronary intervention (PCI) in acute myocardial infarction (AMI) patients with ST-segment elevation or left bundle branch block (LBBB) on the electrocardiogram (ECG) performed closest to hospital arrival time.

The functional query logic and report query to answer this question follows.

Functional Query Logic

```
1. Identify distinct Encounters using the following query logic:

   Where the Encounter End Date and Time is within the reporting period and
         Encounter Subtype = Inpatient
   Where (Encounter.Start Date and Time in years – Patient (Individual Party)
        birth date and time in years >= 18 years) and
        Where Concern Subtype = 'DIAGNOSIS' and
        Concern Code = 'ACUTE MYOCARDIAL INFARCTION'
AND all of the following:
      * Observation Subtype ='DIAGNOSTIC TEST RESULT' and
      * Observation Code= 'ECG' and
      * Observation Value Note= 'ST-SEGMENT ELEVATION' OR
      * Observation Value Note= 'LEFT BUNDLE BRANCH BLOCK'
AND Where Encounter Start Date and Time -Observation Date and Time <= 24 hours
```

```
       and Where Intervention Subtype = 'PROCEDURE'
         and Intervention Code= 'PERCUTANEOUS TRANSLUMINAL CORONARY ANGIOPLASTY'

EXCLUDE the Encounter if ANY of the following is present:
   *  Where Encounter. Actual Length of Stay >120 and Encounter.Actual Length of
       Stay UoM= 'DAYS'
   *  Where Patient. Clinical Trials Status= 'Enrolled'
   *  Where Encounter.Source of Admission Code = 'TRANSFER'
   *  Encounter Intervention relationship exists with
      Intervention Subtype = 'SUBSTANCE ADMINISTRATION' and
      Substance Type Code = 'MEDICATION' and
      Intervention Code = 'FIBRINOLYTIC AGENTS' and
      Where Substance Administration Status Code= 'ADMINISTERED' or
      'GIVEN' AND Intervention Start Date and Time is <= CASE EVENT DATE and TIME
            Where Intervention Subtype= 'PROCEDURE' and
            Intervention Code= 'PERCUTANEOUS TRANSLUMINAL CORONARY ANGIOPLASTY'
 *  Where the Encounter Intervention relationship exists and
    Where Intervention Subtype = 'PROCEDURE' and
    Where Intervention Code= 'PERCUTANEOUS TRANSLUMINAL CORONARY ANGIOPLASTY' and
    Where Intervention Reason.Intervention Reason Subtype=
       Intervention Delay Reason

2. THEN Where Encounter Intervention intersecting entity exists,
    Select first CASE EVENT DATE and TIME among the identified Encounters
      Where Intervention Subtype= 'PROCEDURE' and
      Intervention Code= 'PERCUTANEOUS TRANSLUMINAL CORONARY ANGIOPLASTY' and
      Case Event Type= 'REPERFUSION EVENT'
```

Report Query

Within the selected Encounters, report Median (Door to Primary PCI) as a numeric value with minutes UoM where Door to Primary PCI = Case Event Date and Time minus Encounter Start Date and Time.

Creating New OLAP Cubes

You create new OLAP cubes in Oracle Healthcare Data Model in much the same way as you would in any other data warehouse. See *Oracle OLAP User's Guide* for more information.

Working with the Oracle Healthcare Data Model Data Mining Model

Working with data mining model involves:

- Modifying the Data Mining Model
- Refreshing the Data Model

Modifying the Data Mining Model

Oracle does not support modified or new data models. Consequently, do not change the data model that is defined and delivered with Oracle Healthcare Data Model. If changes are required to meet your organization requirements, create a copy of the delivered Oracle Healthcare Data Model where you can make your changes.

For information on the data mining model supplied with Oracle Healthcare Data Model, see *Oracle Healthcare Data Model Reference*.

See also: "Troubleshooting Data Mining Model Creation" on page 3-6, and "Working with User Privileges" on page 4-9

Refreshing the Data Model

Over time, the Oracle Healthcare Data Model database information and behaviors may change. Therefore, you may want to refresh the trained mining model based on the latest stored usage data. For more information about the Oracle Mining training and Scoring (applying) process, see *Oracle Data Mining Concepts*.

To refresh the mining model based on latest data, call the procedure `pkg_ohdm_mining.refresh_mining_source`. This procedure performs the following tasks:

1. Refreshes the mining source table `dwd_advr_evt_fall`.
2. Creates a predictive model: `advr_fall_mod` and also deciphers the fall factors for each patient.

Improving Performance

If you find that the performance of your Oracle Healthcare Data Model data warehouse is not as good as you would like, you can tune your Oracle Healthcare Data Model data warehouse as you would any Oracle database as described in *Oracle Database Performance Tuning Guide*.

In particular, consider:

- Adding b-tree Indexes
- Altering Parallelization
- Adding Partitions
- Changing the Tablespaces and Partitions Used by Tables

Adding b-tree Indexes

Oracle Healthcare Data Model provides following B-tree indexes out of the box:

- A primary key index and unique index on composite natural key for each table.
- All non-code foreign keys of potentially large volume tables are indexed.
- Special consideration given to code foundation tables and appropriate indexes are created on those tables.

You can create additional B-tree indexes as appropriate to optimize query and ETL performance. You can leverage the SQL access advisor to further fine tune indexing.

Tip: Bitmap indexes are not recommended for 3NF EDW. They are typically effective in star schema model to achieve star transformation where ad-hoc queries containing a number of disparate AND and OR conditions on low cardinality columns are anticipated.

See: For more information on:

- Indexes in the Oracle Healthcare Data Model, see *Oracle Healthcare Data Model Reference*.
- B-tree indexing, see the b-tree indexing section in *Oracle Database Performance Tuning Guide*.
- SQL Access Advisor, see the discussion on the Advisor in *Oracle Database Performance Tuning Guide*.

Altering Parallelization

Oracle Healthcare Data Model leverages the parallel query and parallel DML feature of Oracle Database. Parallel operations speed up DML statement execution by dividing the work among multiple child processes. Each child process executes its portion of the work under its own parallel process transaction.

If you have a bigger machine, (for example 16 CPUs), then you can alter the degree of parallelization for the entire session as descried in "Enabling Parallel Execution for a Session" on page 4-7 or for certain DML operations as described in "Enabling Parallel Execution of DML Operations" on page 4-7.

Regardless at which level you enable parallelism, the setting of parallelism for a table influences the optimizer. Consequently, when using parallel query, also enable parallelism at the table level as described in "Enabling Parallel Execution at the Table Level" on page 4-8.

Enabling Parallel Execution for a Session

Parallel query is the most commonly used of Oracle's parallel execution features. Parallel execution can significantly reduce the elapsed time for large queries. To enable parallelization for an entire session, execute the following statement.

```
alter session enable parallel query;
```

Enabling Parallel Execution of DML Operations

Data Manipulation Language (DML) operations such as INSERT, UPDATE, and DELETE can be parallelized by Oracle. Parallel execution can speed up large DML operations and is particularly advantageous in data warehousing environments. To enable parallelization of DML statements, execute the following statement.

```
alter session enable parallel dml;
```

When you issue a DML statement such as an INSERT, UPDATE, or DELETE, Oracle applies a set of rules to determine whether that statement can be parallelized. The rules vary depending on whether or not the statement is a DML INSERT statement, or a DML UPDATE or DELETE statement.

Rules for Parallelizing DML UPDATE and DELETE statements

The following rules apply when determining how to parallelize DML UPDATE and DELETE statements:

- Oracle can parallelize UPDATE and DELETE statements on partitioned tables, but only when multiple partitions are involved.
- You cannot parallelize UPDATE or DELETE operations on a nonpartitioned table or when such operations affect only a single partition.

Rules for Parallelizing DML INSERT statements

The following rules apply when determining how to parallelize DML INSERT statements:

- Standard INSERT statements using a VALUES clause cannot be parallelized.
- Oracle can parallelize only INSERT . . . SELECT . . . FROM statements.

Enabling Parallel Execution at the Table Level

The setting of parallelism for a table influences the optimizer. Consequently, when using parallel query, also enable parallelism at the table level by issuing the following statement.

```
alter table <table_name> parallel 32;
```

Adding Partitions

The partitioning strategy for Oracle Healthcare Data Model is primarily driven by following aspects:

- Improved query performance through partition pruning technique
- Performance improvement by enabling parallel execution for DML and DDL
- Allow database to scale for large datasets
- Facilitate information lifecycle management including data retention/purging

Oracle Healthcare Data Model comes with some partitioned tables out of the box based on following criterion:

- Identify potential large volume tables.
- Identify partitioning key for each table, (typically the key date field in each table) based on anticipation of the queries going against that table.
- Create range partitions that are based on date field identified in previous step – one partition for each period (month/year and so on.)
- Create hash partitions for tables where appropriate partition key is a unique ID (surrogate key) rather than the date field.
- Leverage interval partitioning feature to extend range partitioning to simplify manageability by automatically creating the new partitions as needed by data.
- Create local partitioned indexes inheriting the partitioning strategy from respective partitioned table. All local partitioned indexes are local prefixed indexes. (In other words, the index column is same as the column on which index is partitioned.).

You can custom partition additional tables as appropriate depending upon the data volumes applicable for specific implementation. You can also change the tablespaces and partitions used by Oracle Healthcare Data Model tables.

See: For more information on:

- On the partitions defined by Oracle Healthcare Data Model, see *Oracle Healthcare Data Model Reference*.
- Changing the tablespaces and partitions used by Oracle Healthcare Data Model tables, see "Changing the Tablespaces and Partitions Used by Tables" on page 4-9.

Changing the Tablespaces and Partitions Used by Tables

You can change the tablespace and partitions used by tables. What you do depends on whether or not the Oracle Healthcare Data Model table has partitions:

- For tables that do not have partitions (that is, code tables and reference tables), you can change the existing tablespace for a table as described in "Diverting Partitions into New Tablespaces" on page 4-9.
- For tables that have partitions (that is, base and derived tables), you can specify that new interval partitions be generated into new tablespaces as described in "Changing an Existing Tablespace" on page 4-9.

Diverting Partitions into New Tablespaces

By default, Oracle Healthcare Data Model defines the partitioned tables as interval partitioning, which means the partitions are created only when new data arrives.

Consequently, for Oracle Healthcare Data Model tables that have partitions (that is, base and derived tables), if you want the new interval partitions to be generated in new tablespaces rather than current ones, please issue the following statements.

```
ALTER TABLE table_name MODIFY DEFAULT ATTRIBUTES
TABLESPACE new_tablespace_name;
```

When new data is inserted in the table specified by *table_name*, a new partition is automatically created in the tablespace specified by *tablespace new_tablespace_name*.

Changing an Existing Tablespace

For Oracle Healthcare Data Model tables that do not have partitions (that is, code tables and reference tables), if you want to change the existing tablespace for a table then issue the following statement.

```
ALTER TABLE table_name MOVE TABLESPACE new_tablespace_name;
```

Working with Compression

By using a compression algorithm specifically designed for relational data, Oracle is able to compress data much more effectively than standard compression techniques. More significantly, unlike other compression techniques, Oracle incurs virtually no performance penalty for SQL queries accessing compressed tables.

Oracle Healthcare Data Model leverages the compress feature for all base and derived tables which reduces the amount of data being stored, reduces memory usage (more data per memory block), and increases query performance.

You can specify table compression by using the COMPRESS clause of the CREATE TABLE statement or you can enable compression for an existing table by using ALTER TABLE statement as shown below.

```
alter table <tablename> move compress;
```

Working with User Privileges

Installing the Oracle Healthcare Data Model component creates the OHDM_SYS account. Installing the Oracle Healthcare Data Model sample reports create the OHDM_SAMPLE_SYS account. Please make sure you unlock those two accounts with new password.

See: *Oracle Healthcare Data Model Installation Guide* for information on installing Oracle Healthcare Data Model and for unlocking the OHDM_SYS account.

The OHDM_SYS account is for **ohdm_sys** which is the schema for Oracle Healthcare Data Model. This schema contains all components for the Oracle Healthcare Data Model.

The installation process grants the necessary privileges required for users of the accounts. Once you have installed the product, you only need to consider user privileges in the following situations:

- The Intra-ETL programs are executed inside the `ohdm_sys` schema, therefore, they require the full access to the that schema.
- By default, the Oracle Healthcare Data Model sample reports connect to the ohdm_sys schema directly. For security reason, you may want to grant only select privileges to those reporting users. To do this, take the steps outlined in*Oracle Healthcare Data Model Installation Guide*.

Index