ORACLE
INSURANCE

Oracle Health Insurance Data Marts

Administrator Reference

version 9.42

Part number: E91383-01

November, 2017

CHANGE HISTORY

Release	Version	Changes
10.12.2.0.0	9.29	• Added change history paragraph. • Added description for Oracle Business Intelligence translations external file (WBX_OBIEE_TRANSLATIONS.csv) • Added description for configuring generic structure of claim properties • Added description how to configure DWH_OHI_BI.rpd connection pools.
10.12.3.0.0	9.30	• Added new fact table DWH_ONDERHANDEN_WERK (Work in Progress Claims) • Replaced generic role OZG_ROL_SELECT by secure role OBD_SELECT_ROLE.
10.13.1.0.0	9.31	• Added Exadata Support, Migrating OWB from single instance to multi node RAC, added OBI_SELECT_USER • Added new batch Correct monitorcode (ZRGO005S)
10.13.2.0.0	9.32	• Added description for generic dimension properties (DIM_EIGENSCHAPPEN.dat) • Added new parameter P_MAX_DCE_VERSCHIL; a threshold that is taken into account when performing crosschecks for claims. • Added details for the Members check (three differente reference dates) • Added new description for setting 'Policies to date' • Corrected examples for dynamic claim properties • Added Default Secure Install and Enable Security Audit
10.13.2.1.0	9.33	• Added new fact table DWH_OPENSTAANDE_VORDERINGEN (Open Receivables) • Added new batch Load Open Receivables (ZRGO006S)
10.13.3.0.0	9.34	• Updated batch ZRGO005S for new purpose (Update external codings for claims)
10.13.3.3.0	9.34.1	• Added new fact table DWH_DECL_PRESTATIE_DETAILS (Claim Details)
10.14.1.0.0	9.35	• Added new fact tables DWH_OPENSTAANDE_VERPLICHTINGEN (Open Payables), DWH_VERPLICHTINGEN (Payables) and DWH_VORDERINGEN (Receivables)
10.14.1.3.0	9.36	• Added new batch Load Open Policies (ZRGO020S)
10.15.1.0.0	9.37	• Moved documentation of ZRGOE01S, ZRGOS01S, ZRGOD01S, ZRGO003S, ZRGO004S, ZRGO005S, ZRGO006S, ZRGO009S, ZRGO020S, ZRGO021S and ZRGO022S to OLH of OHI Back Office.
10.15.1.1.0	9.38	• Changed paragraph 'SETTING UP ORACLE HEALTH INSURANCE DATA MARTS ENVIRONMENT' due to migration to RDBMS 12.1.0.2 (theme M-4064).
10.15.3.0.0	9.39	• Added a note in the Oracle Warehouse Builder Software section about OBDXMLPS.sql and root.sh
10.16.1.0.0	9.40	• Added Appendix H: Virtual Private Database
10.16.2.0.0	9.41	• Changed version number, no changes.
10.17.1.0.0	9.41	• No changes.
10.17.2.0.0	9.42	• Updated Appendix D: OWB 11gR2 post-cloning process for OHI Data Marts. Reduced the number of manual steps involved in performing the OWB post-cloning process

CONTENTS

INTRODUCTION

The purpose of this document is to give an overview of the architecture of Oracle Health Insurance Data Marts, as well as information on installation and management.

This document may be used as both a training material and a reference material. It is assumed that the reader has a basic knowledge of the Oracle tools being used.

With this document the Oracle Health Insurance Data Marts administrator should be in a position to install and maintain the Data Warehouse.

IMPORTANT DOCUMENTATION

The documentation below is important for the activities of the functional administrator. These describe the design of Oracle Health Insurance Data Marts and contain information that is (potentially) required for installation and management.

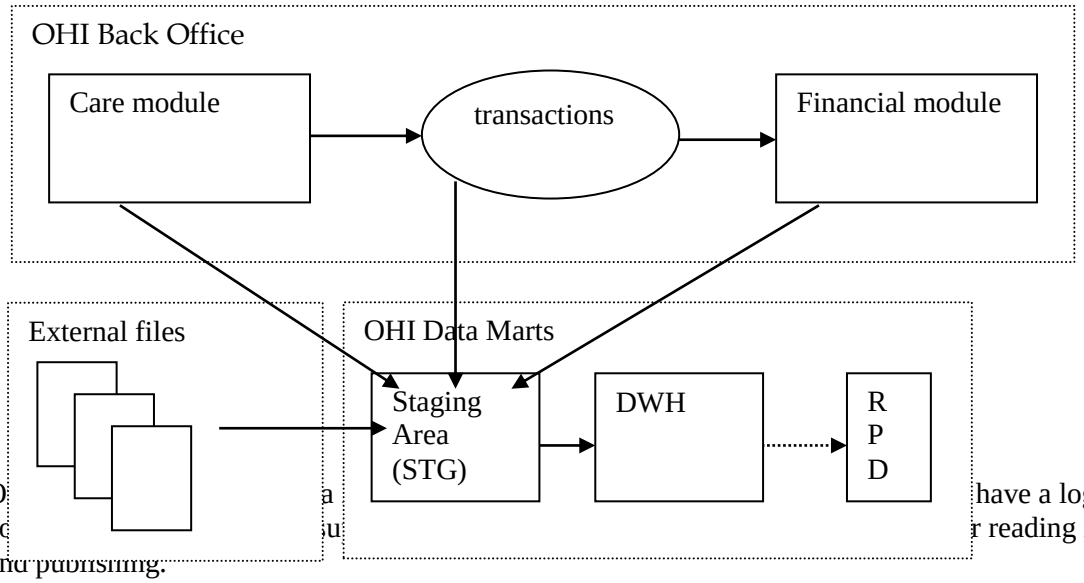
1. Standard product manuals from Oracle DBMS, Oracle Warehouse Builder and Oracle Business Intelligence Enterprise Edition (OBI EE). Documentation on these Oracle products can be found on the Oracle support website (http://support.oracle.com)
2. CTA13508.doc: Oracle Health Insurance Installation, Configuration and DBA Manual
3. CTA13535.doc: Oracle Health Insurance Security Aspects

SOFTWARE USED

Oracle Health Insurance Certification on MOS (logon to MOS, choose tab ‘Certifications’, specify product name ‘ Oracle Health Insurance Data Marts’ and the relevant versions)

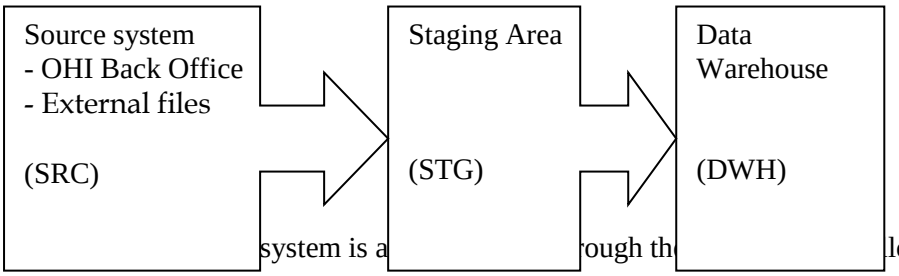
OVERVIEW

The architecture of Oracle Health Insurance Data Marts broadly looks like this:



O a have a logical mutual
co u r reading in, checking
and publishing.

The figure below shows the mutual relationships between the various data areas:



system is a rough th llowing way:

1. 'Raw' data is loaded into the staging area.
2. Validated data is moved from the staging area to the Data Warehouse. There, the data is stored optimally for querying purposes and brought into alignment with the previously-loaded data.

PART I - INSTALLATION

INTRODUCTION

Various components make up Oracle Health Insurance Data Marts. This chapter contains a short description of each.

The set-up and installation of these components is dealt with in the following paragraphs.

SETTING UP ORACLE HEALTH INSURANCE DATA MARTS ENVIRONMENT

OS set-up

Oracle Health Insurance Data Marts runs on a database server. The load run of Oracle Health Insurance Data Marts requires that there is an application server on which the OHI Back Office application is installed. Oracle Health Insurance Data Marts makes use of OHI Back Office functionality including, for example, the batch scheduler for starting and scheduling the load run.

For set-up of the application and database server environment, including set-up of the environmental variables and directory structure, see:

Oracle Health Insurance Installation, Configuration and DBA Manual (chapter 3)

Software installation

Database

Oracle Health Insurance Data Marts must be installed in a dedicated database. It is not permitted to install Oracle Health Insurance Data Marts in the same database as OHI Back Office, as Oracle Health Insurance Data Marts is a Data Warehouse and OHI Back Office is an OLTP system. Different user settings are used in each database.

Please note that the JServer option must also be installed in the database (and the *java_pool_size* parameter is filled in in "init.ora"). This is a requirement to be able to install the OWB Runtime environment.

For more details regarding the installation of the database software, see:

Oracle Health Insurance Installation, Configuration and DBA Manual (chapter 2)

Oracle Warehouse Builder

An OWB Runtime environment is required in order to be able to execute the Oracle Health Insurance Data Marts ETL process. As of release 10.15.1.1.0 of OHI Health Insurance Oracle 12cR1 is used on the database but OWB still uses Oracle 11gR2 (11.2.0.4).
Oracle Health Insurance Data Marts releases and patches are installed from an application server. This application server can be used in 2 configurations. As stated before

1. Configuration with database server and application server on different hosts (separate database and application server)

   The installation software of Oracle Health Insurance only uses the OWB software in the 11g Oracle Home on the database server for the installation of Oracle Health Insurance Data Marts. Therefore, installation of the full Oracle 11g software on the application server is not necessary.

2. Configuration with database server and application server on same host (combined database and application server)

For the right version of the OWB software, see:

Oracle Health Insurance Certification on MOS (logon to MOS, choose tab ‘Certifications’, specify product name ‘ Oracle Health Insurance Data Marts’ and the relevant versions)

Creation of an Oracle Health Insurance Data Marts database

A pluggable database needs to be created first before Oracle Health Insurance Data Marts can be installed.

Parameters

The following database parameters must be set to the specified values:

```
DB_BLOCK_SIZE                 = 16384 or DB_BLOCK_SIZE=8192
NLS_SORT                      = BINARY          # For performance reasons
OPTIMIZER_MODE                = ALL_ROWS        # For performance reasons
STAR_TRANSFORMATION_ENABLED   = TRUE            # For performance reasons
STATISTICS_LEVEL              = TYPICAL         # For self-tuning capabilities
UNDO_MANAGEMENT               = AUTO            # Automatic Undo
WORKAREA_SIZE_POLICY          = AUTO            # Automatic SQL Execution Memory Management
```

The following parameters are influenced by the environment variables as used by the process of the connecting session. When they are not set or specified the database instance values apply. Below example values are shown for a Dutch environment.

```
NLS_LANGUAGE                  = DUTCH           # Choose the appropriate language
NLS_NUMERIC_CHARACTERS        = ",." ***        # personal choice
NLS_TERRITORY                 = ”THE NETHERLANDS”
```

*** = this setting cannot be changed once the application is used

The following database parameters must be set to the specified values at a *minimum*: When a *maximum* value applies, this is specified:

```
DML_LOCKS                     = 500
OPEN_CURSORS                  = 500
SESSION_CACHED_CURSORS        = 500
PROCESSES                     = 200
JOB_QUEUE_PROCESSES           = 10
```

As well as the parameters above, there is also a parameter that is extremely important for parallel processing during both the loading process and for queries by end-users:

```
PARALLEL_MAX_SERVERS=number
```

If this setting is left out of the init.ora, it will be set to a very high value by default. It is therefore highly advisable to set a value for this parameter. However, the correct value for this parameter depends on the system (speed of I/O, number and speed of CPUs, among other things).

Unfortunately, there is no hard and fast rule. More details on setting these parameters can be found on Oracle Support under note 280939.1 "Checklist for Performance Problems with Parallel Execution". The manual 'Oracle Database Data Warehousing Guide' describes how the execution of parallel statements works. To find out whether parallel statements have actually been executed serially due to a lack of parallel servers, the following statement, for example, can be used:

```
select * from gv$sysstat where name like 'Parallel operation%';
```

Note 1: In the production environment it is *not* permitted to have activated the database events, unless explicitly requested by Oracle Health Insurance Development or Oracle Support Services.

Note 2: If user settings that have not been recommended or prescribed are used in the Database or Application Server, the customer may be asked to reset these user settings if problems arise that may be connected.

The underlying reason for this is to avoid unnecessary instability risks. Use in custom applications also requires special consideration.

Tip 1: Oracle Health Insurance recommends the use of *Oracle Resource Management.* See paragraph 8.4.2 of CTA13508.doc: Oracle Health Insurance Installation, Configuration and DBA Manual.

Tablespaces

The tablespaces below must be created for data and indexes:

Tablespace
STG_STAD
STG_STAI
STG_DYND
STG_DYNI
DWH_STAD
DWH_STAI
DWH_DYND
DWH_DYNI
OZG_DIM_SYS_TAB
OZG_DIM_SYS_IND
OZG_FACT_SYS_TAB
OZG_FACT_SYS_IND
OZG_LOG_TAB
OZG_LOG_IND
OWBSYS
WBXRUN

The created tablespaces must comply with the following requirements:

1. Locally Managed
2. System managed extent allocation
3. Automatic Segment Space Management
4. 8K or 16K blocksize

Note 3: Oracle Health Insurance requires the use of a *default temporary* tablespace for temporary segments.

Database users

The following users should be created in the Oracle Health Insurance Data Marts database:

User	ID	Description
Oracle Health Insurance Data Marts repository owner	OWBSYS	This is the owner of the OWB repository on the Oracle Health Insurance Data Marts database server. This user is created during the installation of the OWB 11g software. The data/index tablespace is OWBSYS.

User	ID	Description
Oracle Health Insurance Data Marts Workspace owner	OHI_BI_WS_OWN	This is the owner or the workspace in which the Oracle Health Insurance Data Marts objects have been created and are executed.
Oracle Health Insurance Data Marts owner	OBD_OWN	This is the owner of the Oracle Health Insurance Data Marts objects. **This user must be created using the Oracle Health Insurance Data Marts installation software** with USERS as data/index tablespace
Batch User	BATCH	This is the user with which the Batch Scheduler scripts that are requested in the OHI Back Office application are executed. This user does not own any objects and therefore does not need its own separate tablespace. The user should be '**externally identified**' so that the loading process can be started remotely from the OHI Back Office application server. **This user is created by the Oracle Health Insurance Data Marts installation software.** (see **'**Oracle Health Insurance Security Aspects' document on iProjects Files for security of the BATCH account)

The following user should be created in the OHI Back Office database:

User	ID	Description
Select user for extractions	OBD_SELECT_USER	This is the user that performs the selections on the OHI Back Office database. This user should be assigned the following privileges: CREATE SESSION OBD_SELECT_ROLE SELECT ON V_$DATABASE

Database link

A database link should be created from the Oracle Health Insurance Data Marts database to the OHI Back Office database with the name SRC_OPENZORG. The link should be created in the following way (under user OBD_OWN) in the Oracle Health Insurance Data Marts database:

```
create database link SRC_OPENZORG
connect to OBD_SELECT_USER
identified by [password]
using '[servicename]'
;
```

Database directories

For a number of external tables the following database directories need to be created under the OBD_OWN schema:

Directory	Value	Description
OBD_INPUT	Value of $OZG_BASE	This is the location of the source files that are used for the external tables
OBD_LOG	/tmp	This is the location for the log, discard and bad files of the external tables.

Example:

if $OZG_BASE = /ozg/app/oracle/product/Zorg/oton

```
create or replace directory OBD_INPUT
as '/ozg/app/oracle/product/Zorg/oton';
```

Default Secure Install

To avoid default passwords, and as general good security practice it is recommendable to change your passwords on a regular basis. This section describes how to change your passwords for the various accounts within the OHIBI environment.

The database user OHI_BI_WS_OWN is the Oracle Warehouse Builder user that owns the workspace. To change the password of this user the following actions have to be performed:

1. log on the OHI Data Marts database as sys or system user:
   sqlplus sys/<PASSWORD>@<ENVIRONMENT_NAME> as sysdba

   SQL> alter user ohi_bi_ws_own identified by <NEW_PASSWORD>
   SQL>grant execute on dbms_workload_repository to ohi_bi_ws_own;
   type exit to exit sqlplus

SQL> exit

2. log on to the application server as the batch user:
   ssh batch@<HOSTNAME>

3. Set your environment
   . ozg_init.env < ENVIRONMENT_NAME >

4. navigate to $OZG_BASE/conf
   cd $OZG_BASE/conf

5. open ohipatch.conf in vi
   vi ohipatch.conf

6. Edit the entry rtr_pw
   rtr_pw=<NEW_PASSWORD>

7. Make sure the file ohipatch.conf is not world readable

The database user OWBSYS is the system user for Oracle Warehouse Builder, to reset the password (see also MOS 1305938.1)

1. Connect to SQLplus as dba user and change the OWBSYS password using:

   SQL> alter user OWBSYS identified by <owbsys_new_password>;

2. Connect as OWBSYS and execute the following:

   SQL>@<Oracle_home>/owb/rtp/sql/set_repository_password.sql <owbsys_new_password>

3. Execute :

   SQL>@<Oracle_home>/owb/rtp/sql/service_doctor.sql

4. If the service doctor does not report any issue like "cannot connect user=OWBSYS host=hostname port=portnumber service_name=your_servicename" then you should be able to stop and start the service using start_service.sql/stop_service.sql.

NOTE: It is highly recommendable to restart the service with the start / stop scripts in the step above.

The database user OBD_OWN is the database user that actually owns the Data Marts in the OHIBI database. The OBD_OWN password is used on the application server to log on to OHIBI. The password is also used in Oracle Warehouse Builder locations, therefore a change in the OBD_OWN requires multiple actions.

1. Connect to SQLplus as dba user and change the **OBD_OWN** password using:
   SQL> alter user **OBD_OWN** identified by <PASSWORD>;
2. Log on to the application server
   ssh batch@<hostname>
3. Set the environment:
   . ozg_init.env DB1202
4. Navigate to your wallet
   cd network/admin/special
5. To list all wallet entries:
   mkstore -wrl . -listCredential

6. Modify the credentials:
   mkstore -wrl . -modifyCredential <ENVIRONMENT_NAME>_INSTALL OBD_OWN <PASSWORD>
7. Check the connection string with:
   sqlplus /@<ENVIRONMENT_NAME>_INSTALL
8. Follow the Appendix: OWB 11gR2 post-cloning process for OHI Data Marts step 8 up to step 11, and update the locations for OBD_OWN.

Create a Default Password Policy

It is mandatory to have a password policy that conforms to these rules:

- Password length must be at least 8 characters
- Passwords must expire after 180 days
- After 10 failed login attempts, login must be suspended for 5 minutes.

Below is an example password policy function that may be used.

```
--
-- Security Profile
--
CREATE OR REPLACE
 FUNCTION ohibi_pass_validation(
      username     VARCHAR2,
      password     VARCHAR2,
      old_password VARCHAR2)
    RETURN BOOLEAN
  AS
  BEGIN
    IF LENGTH(password) < 8 THEN
      RETURN FALSE;
    ELSE
      RETURN TRUE;
   END IF;
END;
/
create PROFILE ohibi_profile LIMIT
FAILED_LOGIN_ATTEMPTS 3  -- Account locked after 3 failed logins.  PASSWORD_LOCK_TIME
(1 / 24 / 60) * 5     -- Number of days account is locked for. UNLIMITED required
explicit unlock by DBA.  PASSWORD_LIFE_TIME 180    -- Password expires after 180
days.  PASSWORD_GRACE_TIME 3    -- Grace period for password expiration.
PASSWORD_REUSE_TIME 120  -- Number of days until a specific password can be reused.
UNLIMITED means never.
PASSWORD_REUSE_MAX 10    -- The number of changes required before a password can be
reused. UNLIMITED means never.
PASSWORD_VERIFY_FUNCTION ohibi_pass_validation;
/
```

All OHI Data Marts accounts should use this profile.

Enable Security Audit

To be able to track who tried to logon to the OHIBI database it is mandatory to have auditing enabled for this. Make sure your audit_trail is set to DB and then as DBA run the following stattement:

```
SQL> audit create session whenever not successful;
```

General instructions

For instructions relating to active management of Oracle Health Insurance Data Marts, see:

Oracle Health Insurance Installation, Configuration and DBA Manual

(chapter 8)

INSTALLATION OF ORACLE WAREHOUSE BUILDER (OWB) REPOSITORY

CLIENT INSTALLATION

Oracle Health Insurance Data Marts was developed with the help of OWB. The metadata relating to the Oracle Health Insurance Data Marts objects is stored in an OWB (Design) repository. This doesn't play a part in the use of Oracle Health Insurance Data Marts. A client installation is therefore not necessary.

SERVER INSTALLATION

Software

From Oracle Database 11g Release 1, OWB is automatically installed in the same Oracle home as the database. This happens at the same time as the installation of the Oracle database software. As of release 2011.01, the OWB software only has to be present on the database server (see also Software Installation - Oracle Warehouse Builder).

NOTE: File OBDXMLPS.sql is used to deploy an OWB XML Mapping Package by writing the package to the filesystem on the database server, and then run a Database Scheduler job to load it. Remove the file from database and filesystem afterwards.

For the remote job to run successfully, it should be validated that following files have correct access; if this is not the case the Oracle Database software installation has not been completed correctly; most probably the root.sh script has not been run. Correct the properties by running root.sh.

```
$ORACLE_HOME/rdbms/admin/externaljob.ora
$ORACLE_HOME/bin/extjob
```

Properties should be:

```
-rw-r--r-- 1 root oinstall 1.5K Mar 19  2010 externaljob.ora
-rwsr-x--- 1 root oinstall 934K Mar 19  2010 extjob
```

For more details, see http://www.orafaq.com/forum/t/88882/2.

Repository

An OWB repository has to be present on the Oracle Health Insurance Data Marts database server in order to run the mappings generated by OWB. Information is stored in the repository relating to installation and running of mappings.

Installation of an OWB repository is done as follows:

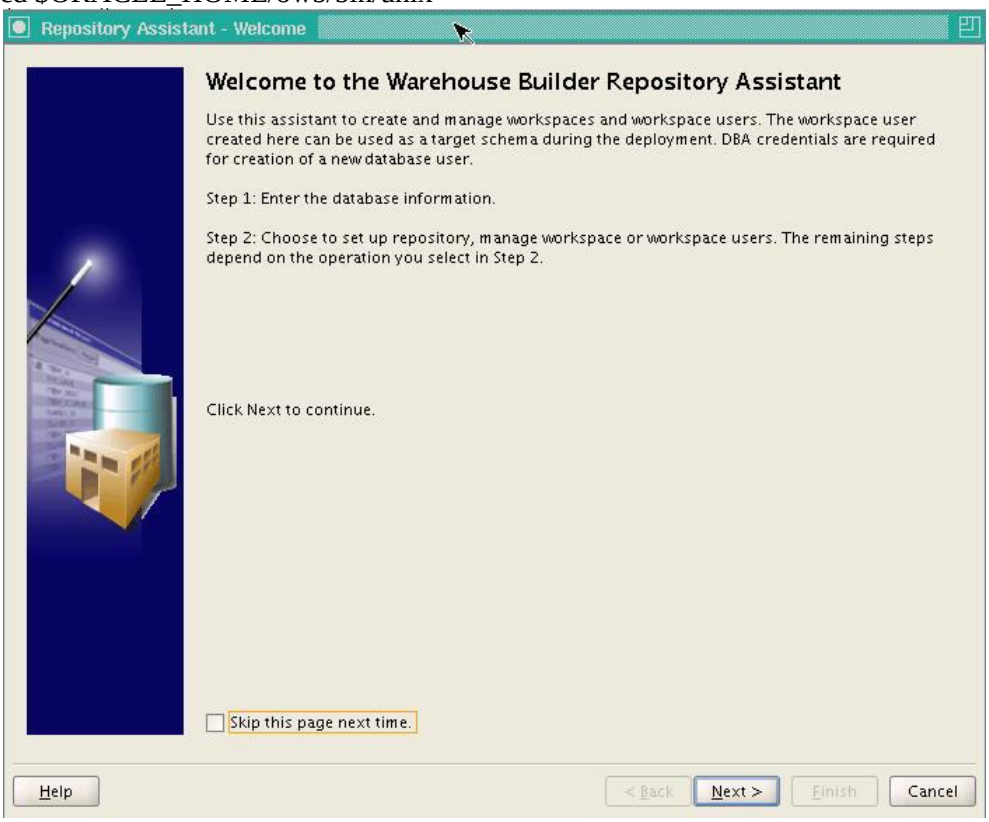
1) . ozg_init.env <*ohi_db_name*>
2) . ozg_init.env DB11G
3) Set read (R) rights for OTHERS on the whole of $ORACLE_HOME/owb/bin/unix and explicit execution (X) rights for OTHERS on OMBPlus.sh and setowbenv.sh in the $ORACLE_HOME/owb/bin/unix directory.
4) Login to the database with SYS (as SYSDBA)
5) create tablespace OWBSYS logging
   datafile '<*datafiledir*>/<*datafilename*>.dbf'
   size 2048M
   autoextend on

next 32M
extent management local;

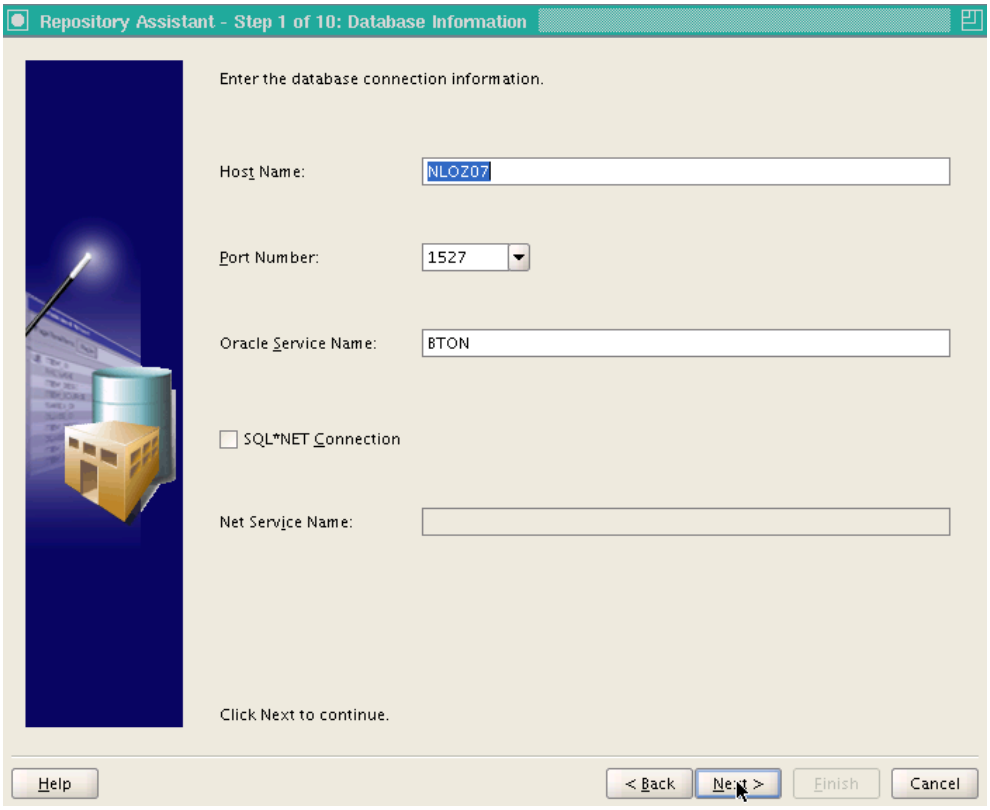
6) Install the new OWB 11g repository with:
@$ORACLE_HOME/owb/UnifiedRepos/cat_owb.sql OWBSYS
7) Reset the ORACLE_HOME of the Control Center with:
@$ORACLE_HOME/owb/UnifiedRepos/reset_owbcc_home.sql
8) Unlock the OWBSYS account and set a password:
alter user OWBSYS account unlock;
alter user OWBSYS identified by <*password*>;
grant restricted session to OWBSYS;

Exit;

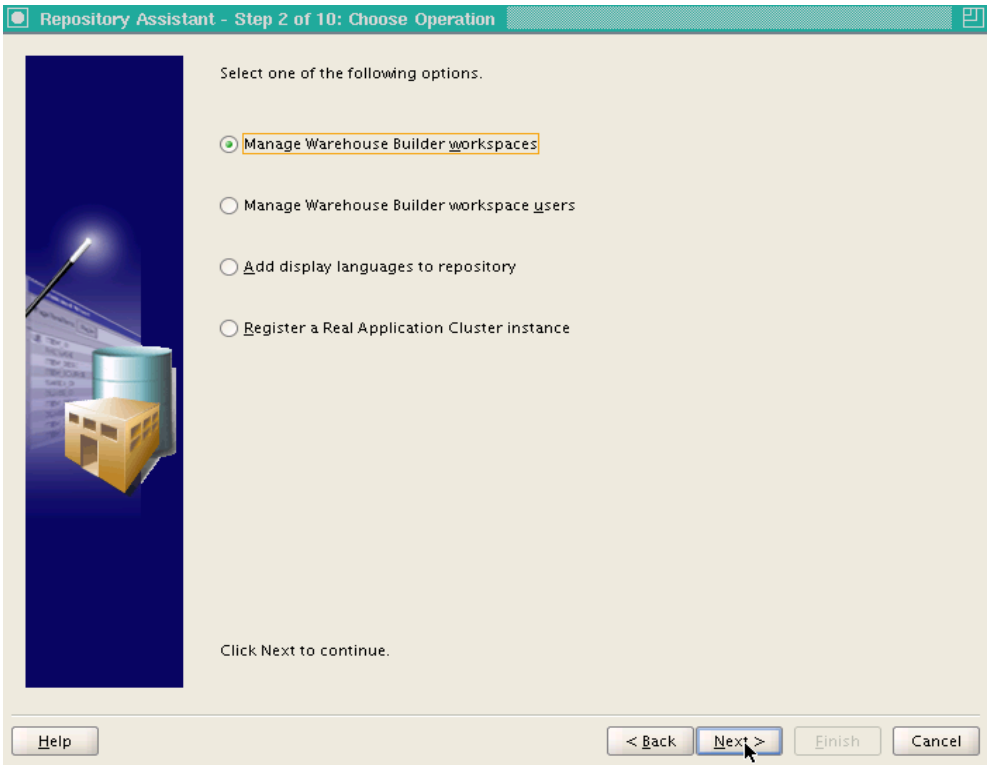
9) cd $ORACLE_HOME/owb/bin/unix



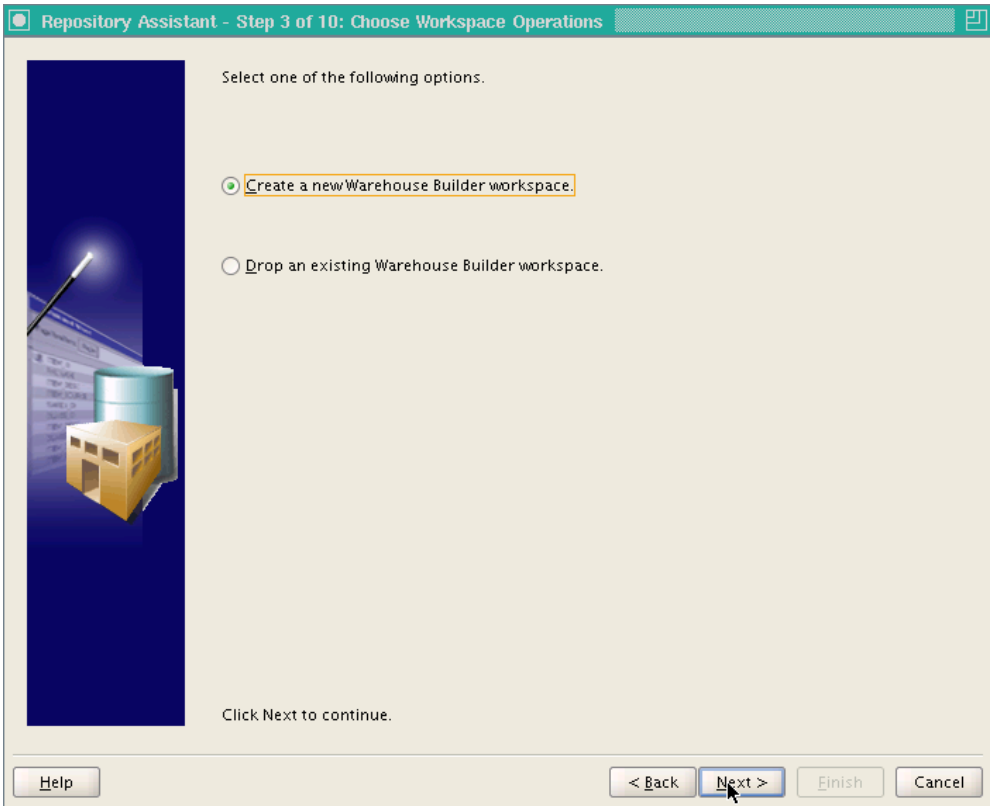
10) ./reposinst.sh
Click on Next>



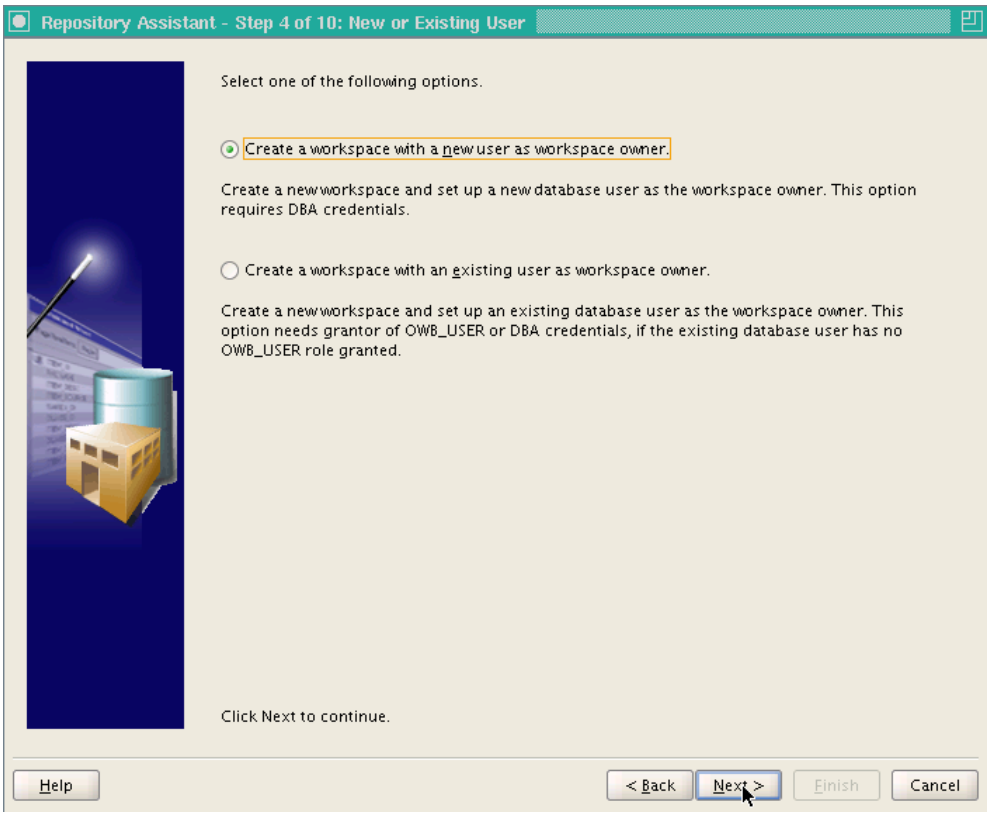
Host Name: Name of the host server
Port Number: Port of the listener for the database
Oracle Service Name: Service name of the OHI Data Marts database
Then click on Next>



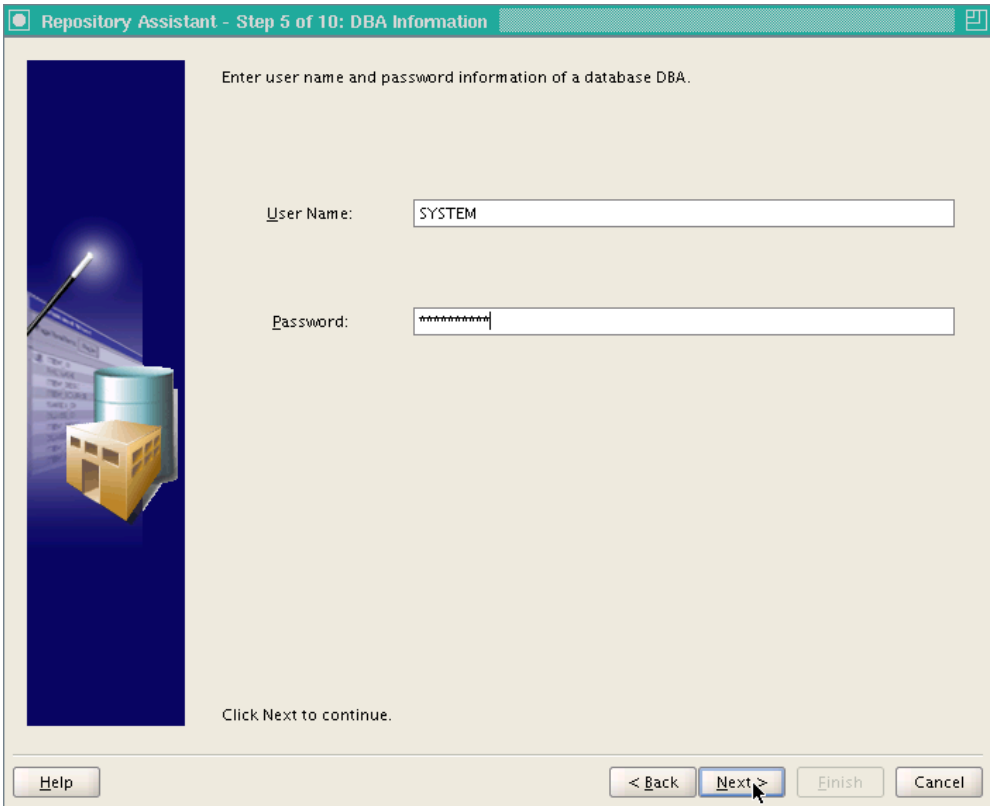
Choose "Manage Warehouse Builder workspaces" and click on Next>



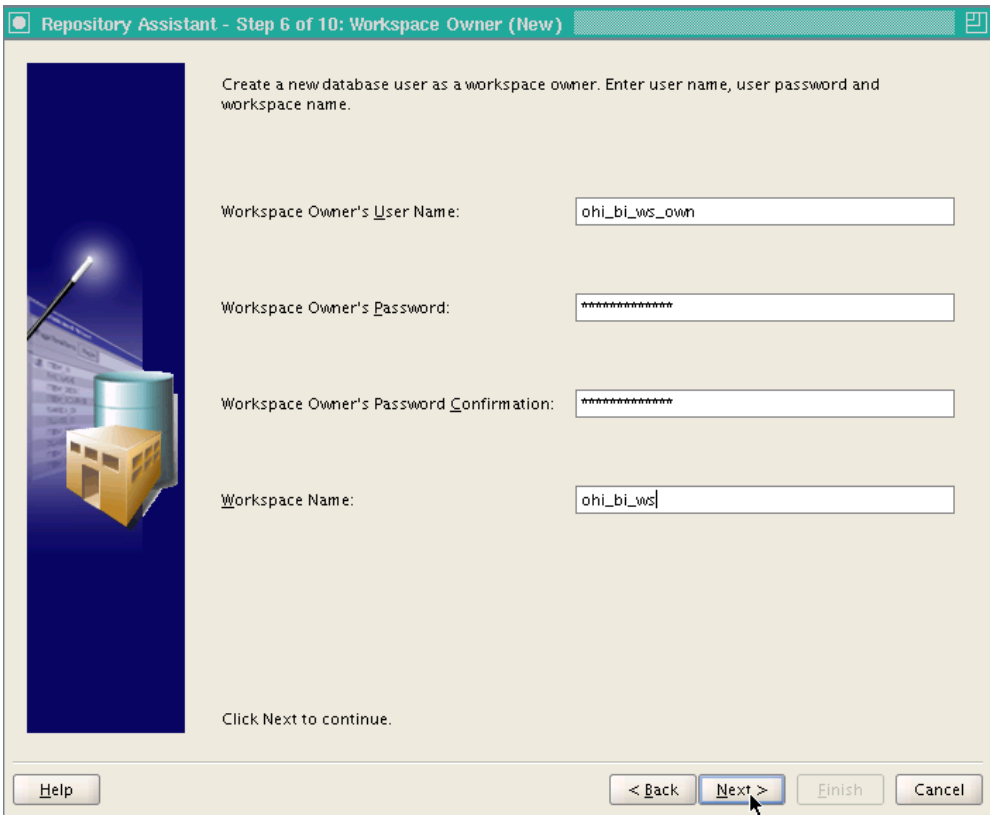
Choose "Create a new Warehouse Builder workspace" and click on Next>



Choose "Create a workspace with a new user as workspace owner" and click on Next>

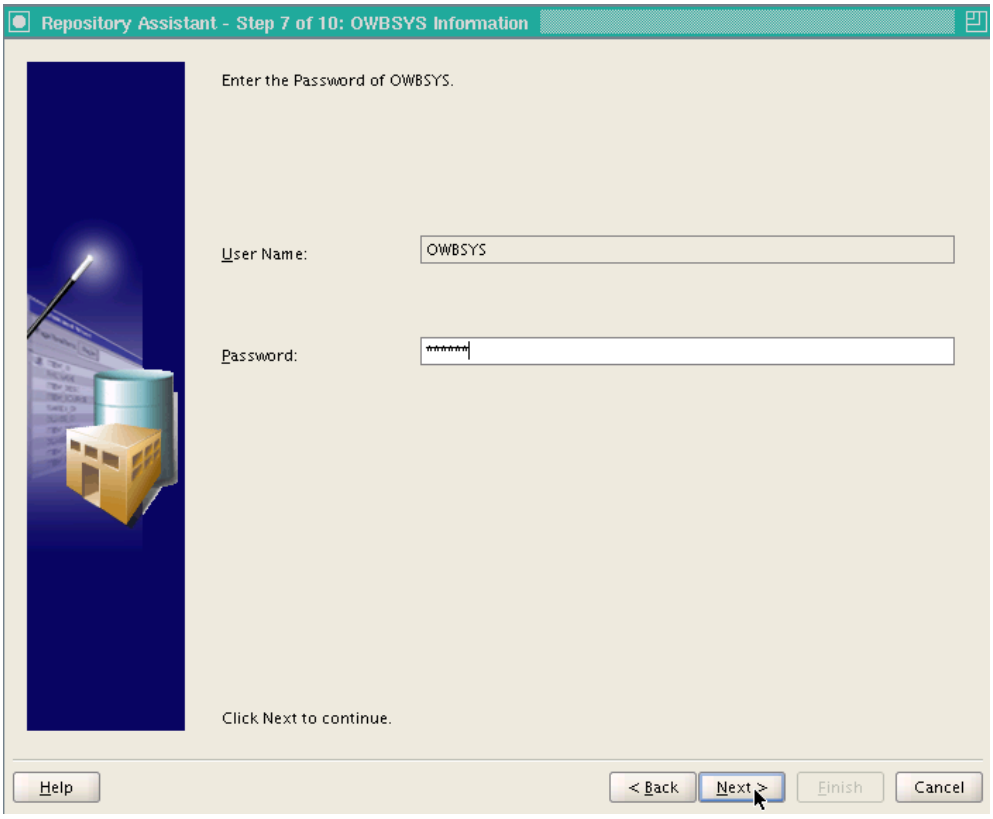


Enter the password for the SYSTEM user and click on Next>



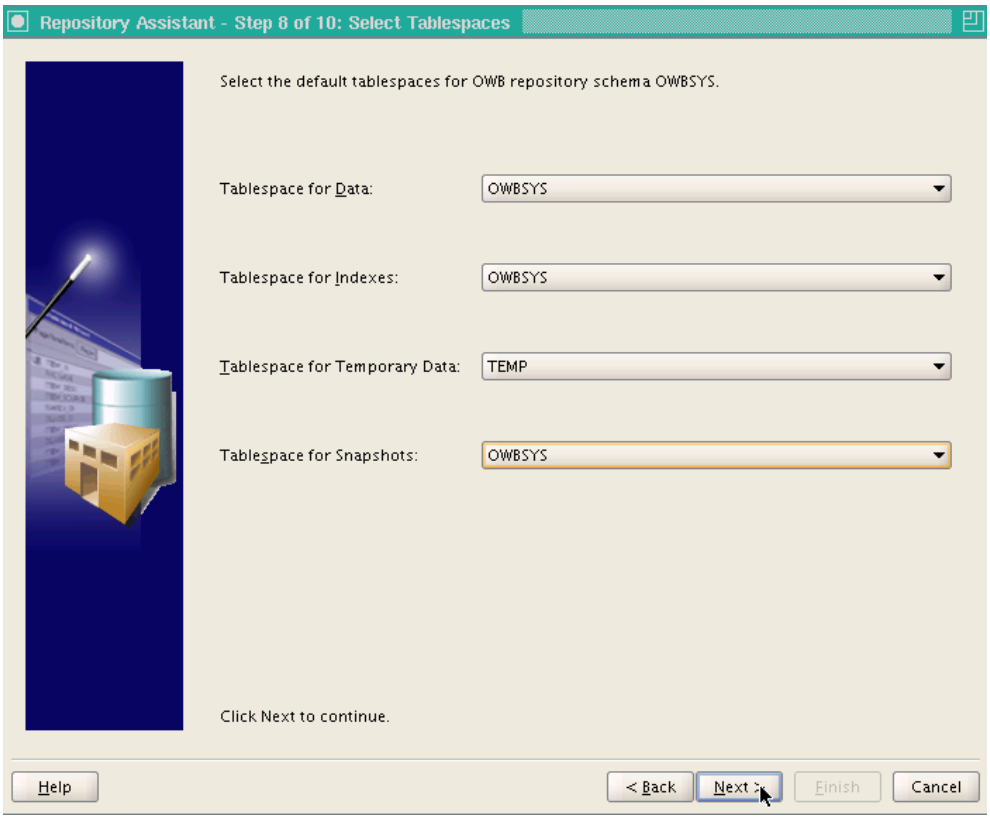
Workspace Owner's User Name:	ohi_bi_ws_own
Workspace Owner's Password:	<password>
Workspace Owner's Password confirmation:	<password>
Workspace Name:	ohi_bi_ws

Click on Next>



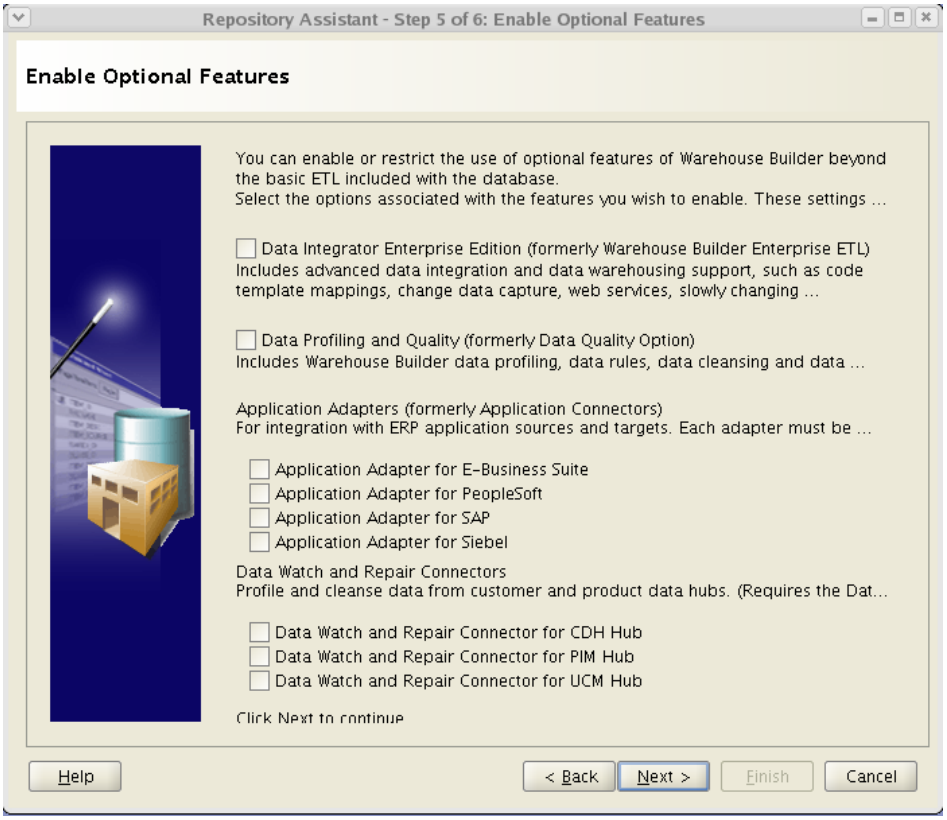
User Name: OWBSYS
Password: <password>

Click on Next>

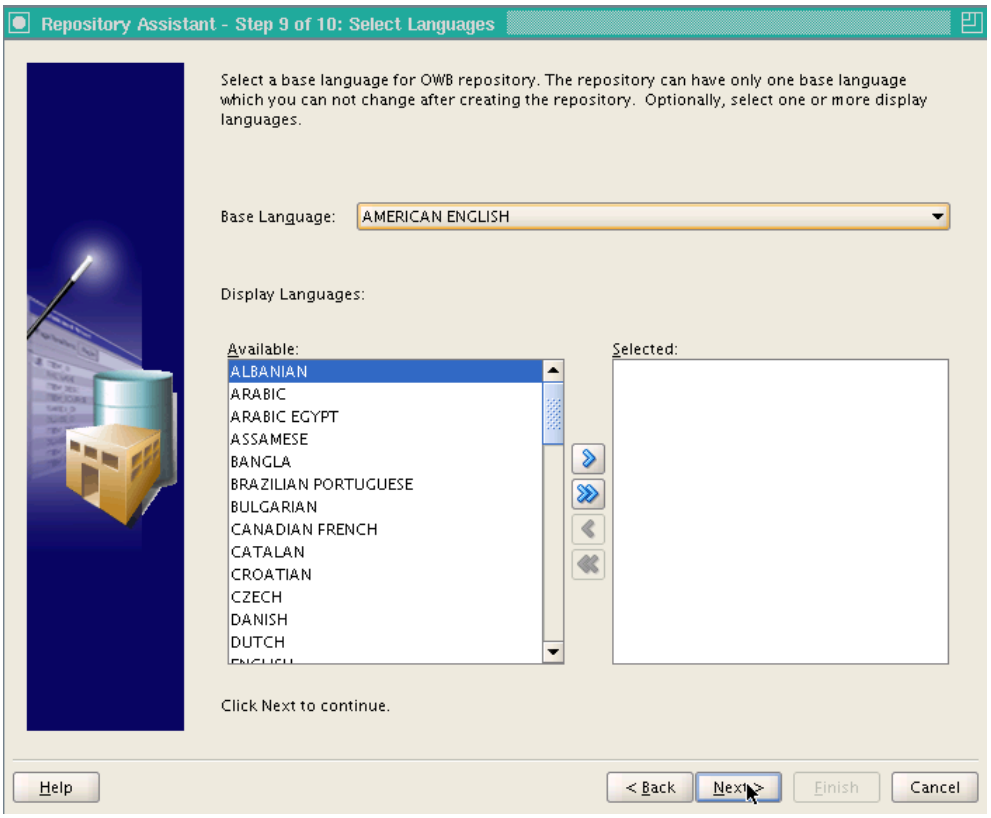


Tablespace for Data: OWBSYS
Tablespace for Indexes: OWBSYS
Tablespace for Temporary Data: TEMP
Tablespace for Snapshots: OWBSYS

Click on Next>

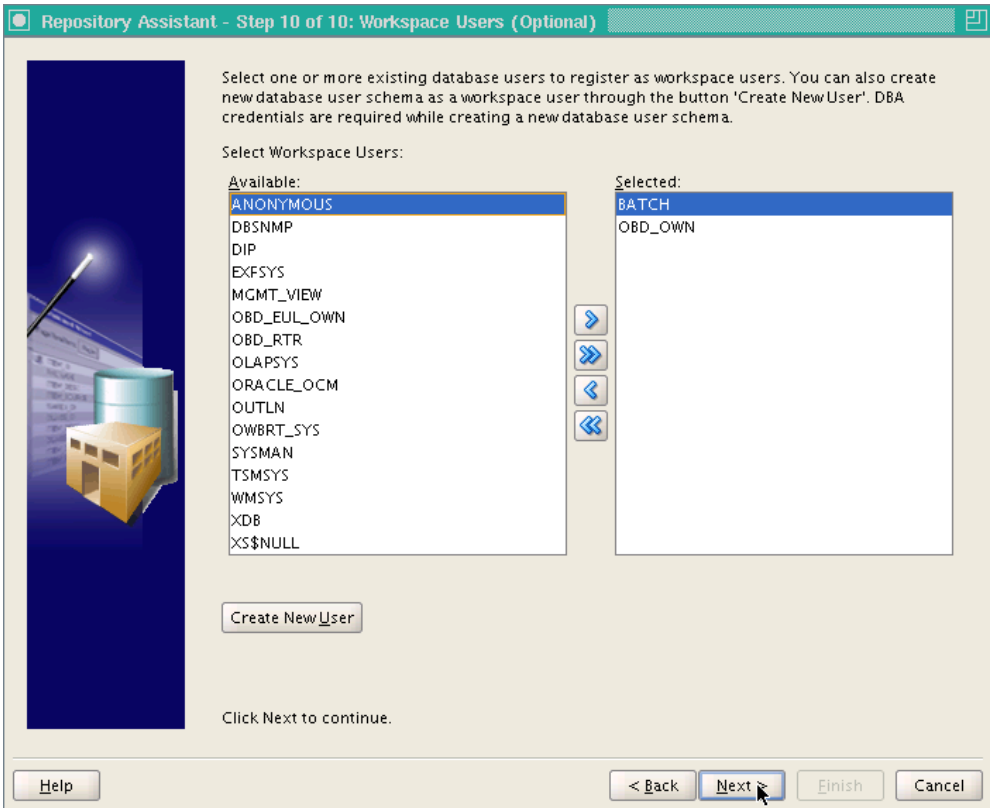


Deselect all optional features.

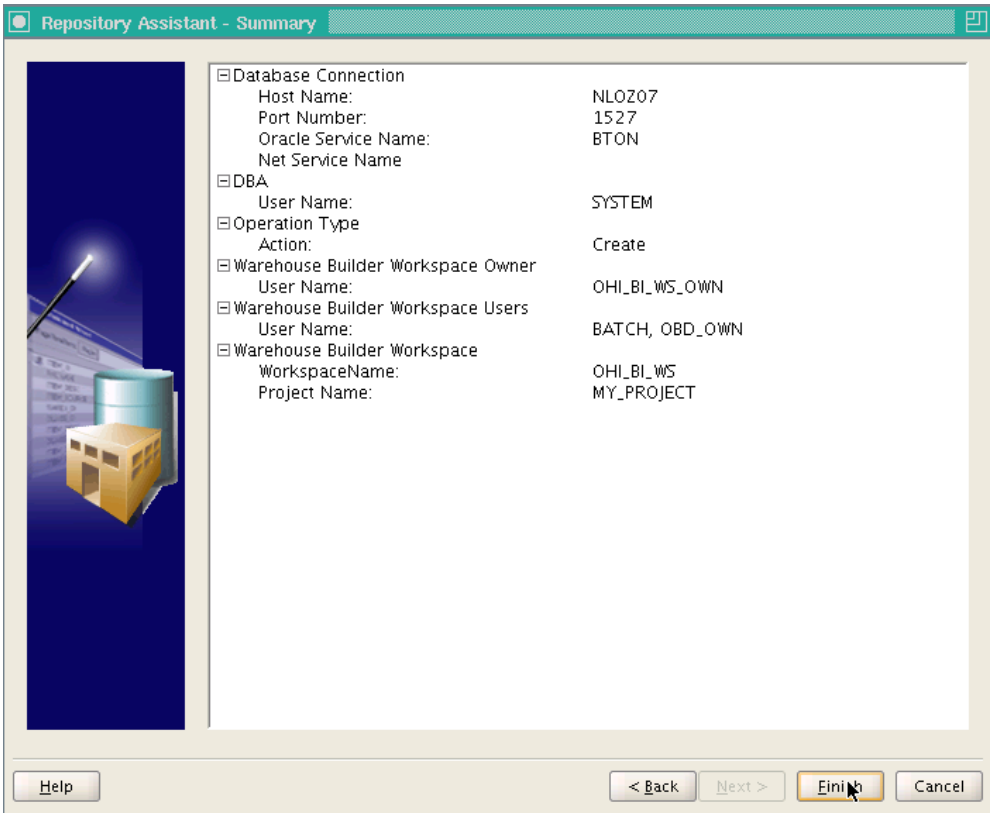


Base Language: AMERICAN ENGLISH

Click on Next>

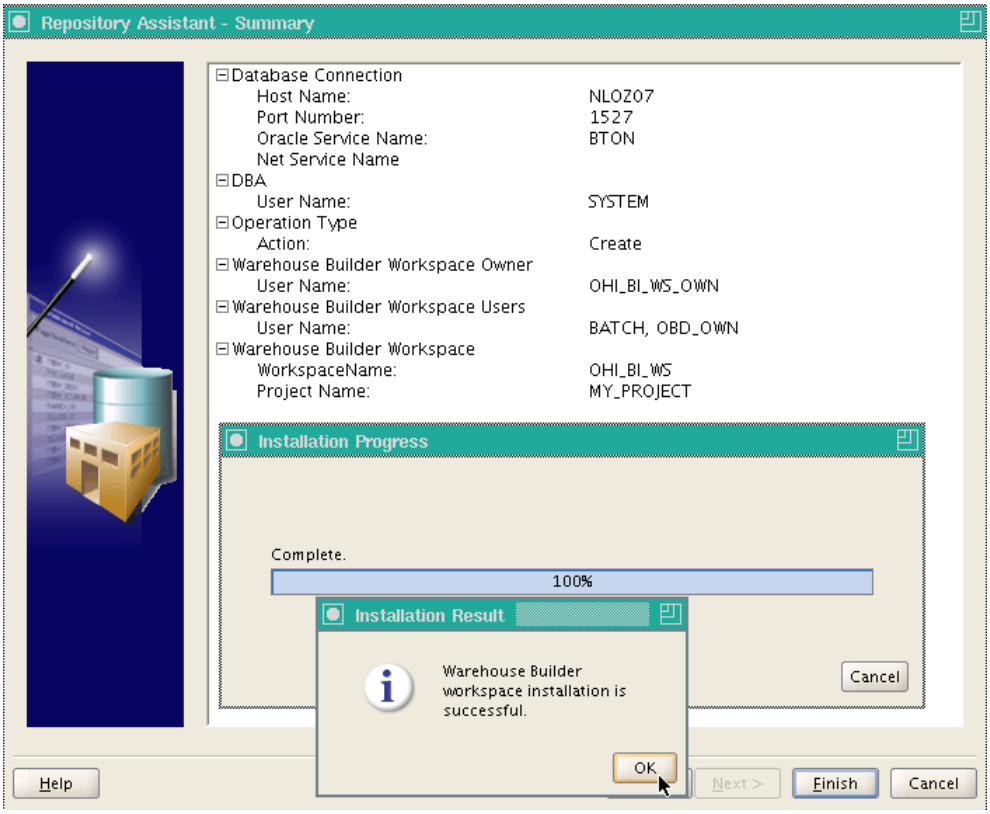


Add the users BATCH and OBD_OWN as Workspace Users if these already exist. If not, these users will have to be added later using /reposinst.sh. Click on Next>



Check the displayed 'Summary'. This should also show that no optional features will be installed. If everything is correct, click on Finish.

The repository with the OHI_BI_WS_OWN workspace is now installed.



Click on OK.

11) sqlplus / as sysdba

12) Assign grant from SYS to OHI_BI_WS_OWN:

```
grant restricted session to ohi_bi_ws_own;
```

13) Assign grants from OWBSYS:

```
connect OWBSYS/<ww>
```

The following grants should be assigned to OHI_BI_WS_OWN:

```
grant select  on owbsys.wb$_iv_control_centers  to ohi_bi_ws_own;
grant select  on owbsys.wb_rt_service_nodes     to ohi_bi_ws_own;
```

If the OBD_OWN user exists already, the following grants can be assigned. If the OBD_OWN user does not exist yet, this must be done later.

```
grant execute on owbsys.wb_workspace_management to obd_own with grant option;
grant select  on owbsys.wb$_iv_control_centers  to obd_own with grant option;
grant select  on owbsys.wb_rt_audit             to obd_own with grant option;
grant select  on owbsys.wb_rt_errors            to obd_own with grant option;
grant select  on owbsys.wb_rt_warehouse_objects to obd_own with grant option;

exit;
```

14) Extract the file DWH_M4064_01.tcl from 10.15.1.1.0.zip (xml directory) and place these in the $OZG_BASE/xml directory.

15) Navigate to the $ORACLE_HOME/owb/bin/unix directory

16) ./OMBPlus.sh $OZG_BASE/xml/DWH_M4064_01.tcl

```
This script creates and registers new 12c OHI Data Marts locations.
What is the password of the OHI_BI_WS_OWN workspace owner?          ohi_bi_ws_own
What is the name of the host of the OHI Data Marts database? slc00tds.us.oracle.com
What is the portnumber of the OHI Data Marts database? 1521
What is the service_name of the OHI Data Marts database? oton
What is the password of the OBD_OWN user on the OHI Data Marts database? obd_own
What is the name of the host of the OHI Back Office database? slc00tds.us.oracle.com
What is the portnumber of the OHI Back Office database? 1521
```

```
What is the service_name of the OHI Back Office database? roon
What is the password of the OBD_SELECT_USER user on the OHI Back Office database?
obd_select_user
Connecting to the oton database...
Switching to the project...
Creating the new locations...
Connecting to the control center...
Linking the locations to the default control center...
Registering the locations...
Locations created, altered and registered.
DWH_M4064_01.tcl script ready.
```

The installation of the OWB repository is now complete.

GENERATION AND INSTALLATION OF ORACLE HEALTH INSURANCE DATA MARTS OBJECTS.

INSTALLATION

Release

Installation of Oracle Health Insurance Data Marts (patch) releases is described in the OHI Back Office Release Installation Manual and is performed on the application server.

ORACLE HEALTH INSURANCE Installation of releases

Because Oracle Health Insurance Data Marts is dependent on OHI Back Office as source environment, the patch level of both must be the same (this can differ at interim patch level if the patches between are only Oracle Health Insurance Data Marts patches or only OHI Back Office patches).
In addition, when installing patches the OHI Back Office environment must always be patched first, and then the Oracle Health Insurance Data Marts environment, otherwise packages may be invalidated during installation.

System parameters

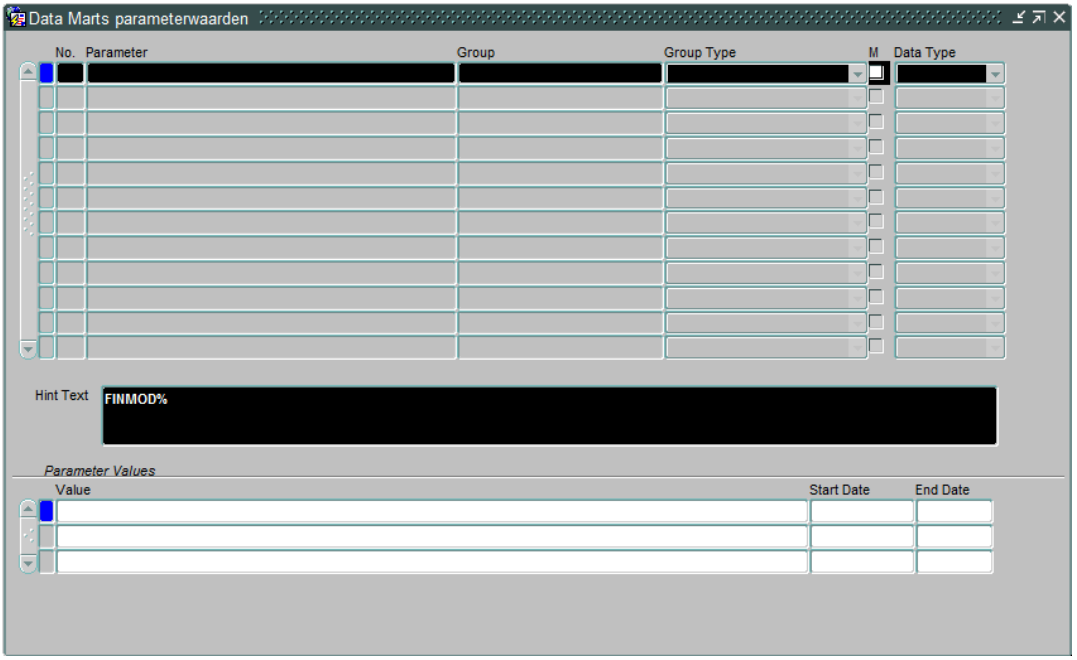
In OHI Back Office a window is supplied 'Data Marts Parameter Values' (ZRGO032F). All Data Mart parameter values can be adjusted here.

These parameters are customer-specific and the value of a number of these parameters **must** be adjusted here **before** Oracle Health Insurance Data Marts can be loaded. A further number may also be adjusted if desired.

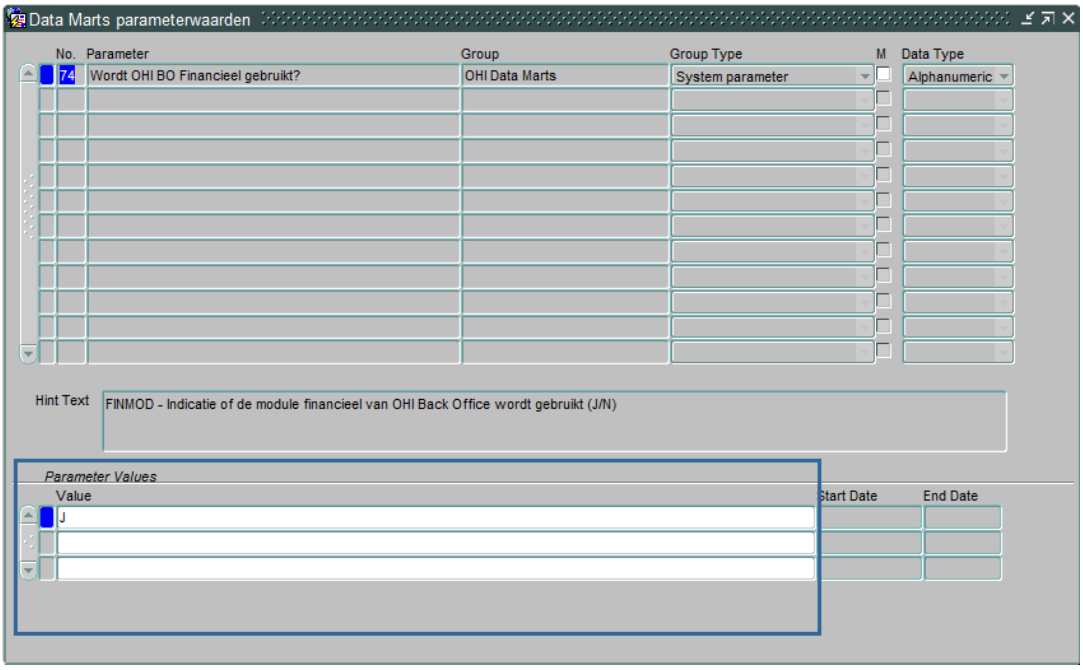
The parameters that **must** be adjusted to ensure correct operation of the loading process:

Parameter	Description
FINMOD	Indication (J/N [Yes/No]) of whether the financial module of OHI Back Office is used to load financial transactions. The default value is J [meaning yes], which means that Oracle Health Insurance Data Marts assumes that the financial data can be retrieved from OHI Back Office.
EOZNLS	The value that is given for this parameter depends on the NLS language setting. This determines, among other things, how a number is displayed. We distinguish between two different styles of notation. The following values indicate each style for the EOZNLS parameter. 1) Decimals separated by a comma Thousands separated by a period e.g.: 1.000.000,001 This style is used in the DUTCH NLS Language, among others. Value for EOZNLS: nls_numeric_characters = ',.' This is the default value for the parameter. 2) Thousands separated by a comma Decimals separated by a period e.g.: 1,000,000.001 This style is used in the AMERICAN NLS Language, among others. Value for EOZNLS: nls_numeric_characters = '.,'

Query for the parameter (e.q. FINMOD):



And change the value of the parameter:



There are a number of parameters that are not mandatory for the correct operation of the load run, but are required for functional population, namely:

Parameter	Description
DCTYP1	First type of third-party code
DCTYP2	Second type of third-party code
DCTYP3	Third type of third-party code
DCTYP4	Fourth type of third-party code
DCTYP5	Fifth type of third-party code

This shows the third-party codes that should be loaded in the members dimension as alternative identifying codes. The values that have to be entered here are the codes of the 'code type' as shown in the 'Code' window in the relation management subsystem of OHI Back Office. The default value is empty.

The other parameters concern descriptions of unknown values, for example, that **can** be adjusted if desired. In case of a non-Dutch environment it is advisable to change the value for the parameter ‘LOGLNG’ from the value ‘NL’ to ‘EN’, this will ensure that information displayed in logging files will be in English.

In addition a number of date system parameters have been added to the table DWH_SYS_PARAMETERS (which are not in the window) that are used to determine from which date a number of fact tables should be loaded.

Parameter	Description
DCEVDM	Date from for load run of Claim fact.
PREVDM	Date from for load run of Premiums fact.
VZEVDM	Date from for load run of Members fact.
ZVNVDM	Date from for load run of Care authorizations fact.
OHWVDM	Date from for load run of Work in Progress Claim fact.

These dates have an initial value of '01-01-1980', which means that this is the date from for the listing of data when loading for the first time. If there is older data that also needs to be loaded, then the relevant date has to be adjusted once in the table.

Finally, there is the parameter INDDCA that specifies whether the aggregation of claims in table DWH_DECLARATIES_AGG should be performed. This parameter should be set manually to 'J' [meaning yes] or 'N' [meaning no].

External files

Three external data files are defined within Oracle Health Insurance Data Marts. This relates to the following files:

- LEEFTIJD_CATEGORIEEN.dat:
  File with a number of age categories, e.g. youth or senior citizen.

```
LEEFTIJD (PK)            NUMBER
CAT_VIJF                 VARCHAR2(30)
CAT_TIEN                 VARCHAR2(30)
CAT_VIJFTIEN             VARCHAR2(30)
CAT_JEUGD                VARCHAR2(30)
CAT_PENSIOEN             VARCHAR2(30)
CAT_SPECIAAL             VARCHAR2(30)
CAT_LOGO                 VARCHAR2(30)
```

- POSTCODE_VERRIJKINGEN.dat:
  File with geographic data enhancements (e.g. disadvantaged neighborhood or DHV area code).

```
POSTCODE_NR (PK)         NUMBER
ACHTERSTANDSWIJK         VARCHAR2(1)
ADVISEUR_BUITENDIENST    VARCHAR2(50)
VESTIGINGSMANAGER        VARCHAR2(50)
RAYONCODE                VARCHAR2(50)
CONSUMENTEN_MARKTREGIO   VARCHAR2(100)
DHV_REGIOCODE            VARCHAR2(10)
DHV_REGIO_OMSCHRIJVING   VARCHAR2(100)
WGR_REGIOCODE            VARCHAR2(10)
WGR_REGIO_OMSCHRIJVING   VARCHAR2(100)
OAD_CODE                 VARCHAR2(10)
ZIP_CODE                 VARCHAR2(10)
WZV_REGIOCODE            VARCHAR2(10)
WZV_REGIO_OMSCHRIJVING   VARCHAR2(100)
```

- WBX_OBIEE_TRANSLATIONS.csv
  File with translations for Oracle Data Marts

```
METADATA_OBJECT          VARCHAR2(4000)
SESSION_VARIABLE (PK)    VARCHAR2(4000)
```

```
NEDERLANDS                     VARCHAR2(4000)
ENGELS                         VARCHAR2(4000)
SPAANS                         VARCHAR2(4000)
```

Templates of these files can be retrieved from iProjects Files (NL Oracle Health Insurance Public - OHI Releases (all products) - Release 2011.01 - Templates - Data Marts).

The file LEEFTIJD_CATEGORIEEN.dat must be filled for correct operation of the Oracle Health Insurance Data Marts load run. The file POSTCODE_VERRIJKINGEN.dat may be left empty.

The files should be placed in the $OZG_BASE directory of the relevant Oracle Health Insurance Data Marts environment on the database server. This is a manual process.

The content of these files can be changed with the exception of the key fields **(PK)**.

When adjusting the files, spaces should be used and not tabs. To check that any changes have been made correctly and that the files have been placed in the correct location, select queries can be performed on the external tables that are populated by the files.

WBX_OBIEE_TRANSLATIONS.csv is used for Oracle Data Marts. The translation for all metadata is described here. The default language is set to English (LANG_ID = ‘en’) which means that in WBX_OBIEE_TRANSLATIONS.csv there should always be a translation for the English language. To limit the languages to be used in Oracle Data Marts set the AllowedLanguages parameter in instanceconfig.xml (e.g. <AllowedLanguages>en,nl</AllowedLanguages> as per documentation *Oracle Fusion Middleware System Administrator’s Guide for Oracle Business Intellgence Enterprise Edition, Localizing Oracle Business Intelligence*). When no language has been chosen yet, the language defaults to the browser language. To adjust the language as per login, modify the url (eg http://Server_Name:port_number/analytics/saw.dll?Dashboard&Lang=nl). Once new translations are made available by modifying the WBX_OBIEE_TRANSLATIONS.csv file, the Oracle Business Intelligence server needs to be restarted.

File	Table
LEEFTIJD_CATEGORIEEN.dat	STG_LEEFTIJD_CATEGORIEEN_EXT
POSTCODE_VERRIJKINGEN.dat	STG_POSTCODE_VERRIJKINGEN_EXT

SET-UP OF ORACLE BUSINESS INTELLIGENCE ENTERPRISE EDITION ENVIRONMENT

Oracle Business Intelligence Enterprise Edition (OBI EE) is the best reporting tool to use.

SOFTWARE INSTALLATION

For installation of OBI EE, reference is made to the installation documentation of this product.

For reports, end users can make use of the presentation layer of OBI EE. Reports can be made with Oracle BI Answers and these can then be shared using the Oracle BI Interactive Dashboard component.

For the right versions of the software, see:

Certifications matrix 10.13.1.0

ORACLE BUSINESS INTELLIGENCE CUSTOMIZATIONS

Customization of OBI is supported, to support your company style. This includes but is not limited to logo, background color and font color, basically all HTML customization may be made. For detailed instructions on Customizing Oracle Business Intelligence see the White Paper.
(**HTTP://WWW.ORACLE.COM/TECHNETWORK/MIDDLEWARE/BI/CUSTOMIZING-ORACLE-BIEE-11G-176387.PDF**)

CREATING A NON PRIVILEGED USER: OBI_SELECT_USER

Based on the principle of "the least privileged" a user should be created with minimal privileges.
The process of creating such a user consists of creating a database user and granting privileges to that user. The priviliges needed are obd_rol_select, which contains only select privileges on the warehouse tables. The create session privilege is required to be able to logon to the database. Access to the directory obd_input is required for accessing the translations stored in an external table.

This is done in SQLPlus, instructions as per below.

Log on to the OHIBI database and create the OBI_SELECT_USER

SQLPlus> create user obi_select_user identified by <password>;

SQLPlus> grant obd_rol_select to obi_select_user;

SQLPlus> grant create session to obi_select_user;

SQLPlus> grant read on directory obd_input to obi_select_user;

When VPD policies are enabled privacy sensitive data will be masked (see `VPD Policies` for which tables/columns data will be maked).

INSTALLATION OF ORACLE HEALTH INSURANCE DATA MARTS REPOSITORY (RPD)

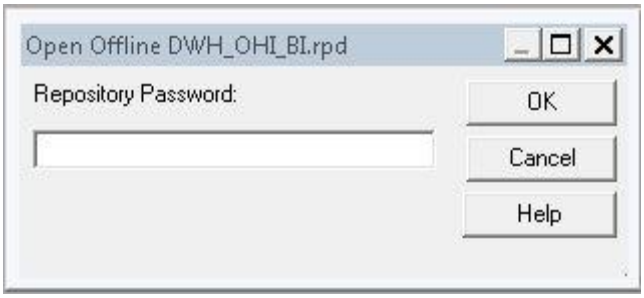
Oracle supplies an OBI EE repository as part of OHI Data Marts. This repository makes the OHI Data Marts database accessible. The repository can be installed on the Oracle BI Server.

This repository includes a subject area for each star schema as well as an over-arching subject area in which the entire data model has been made accessible.

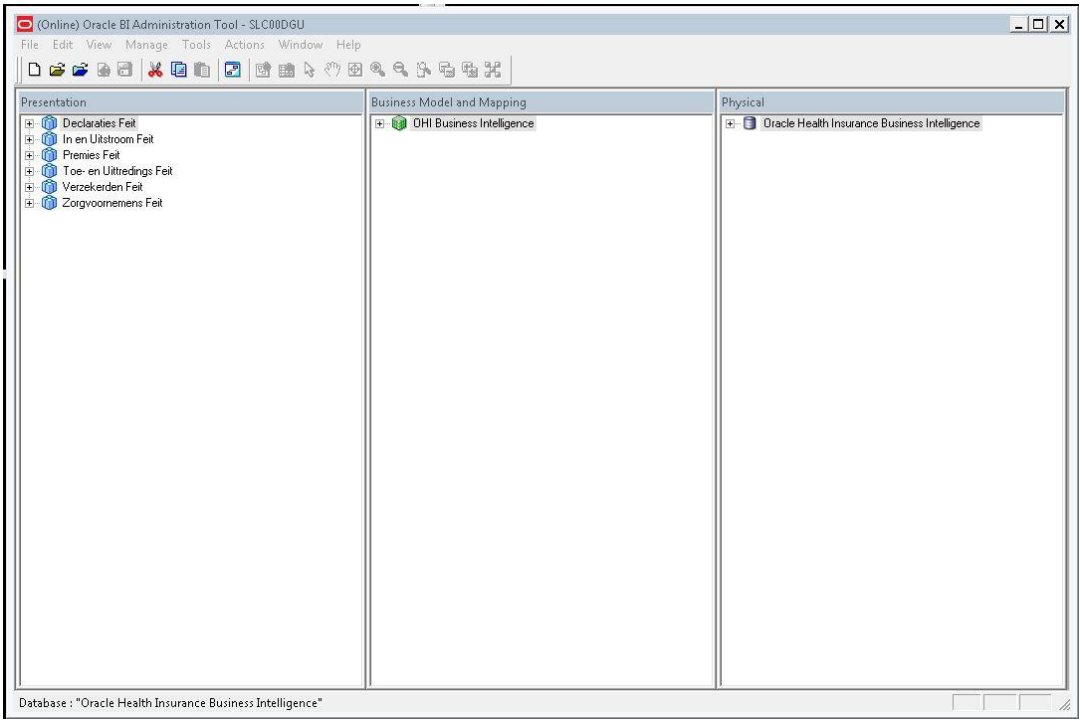
Before being able to use the DWH_OHI_BI.rpd file in your OBIEE environment, you will have to configure 3 connection pools using the Oracle BI Administration Tool.

Please follow the steps below to configure the 3 connection pools.

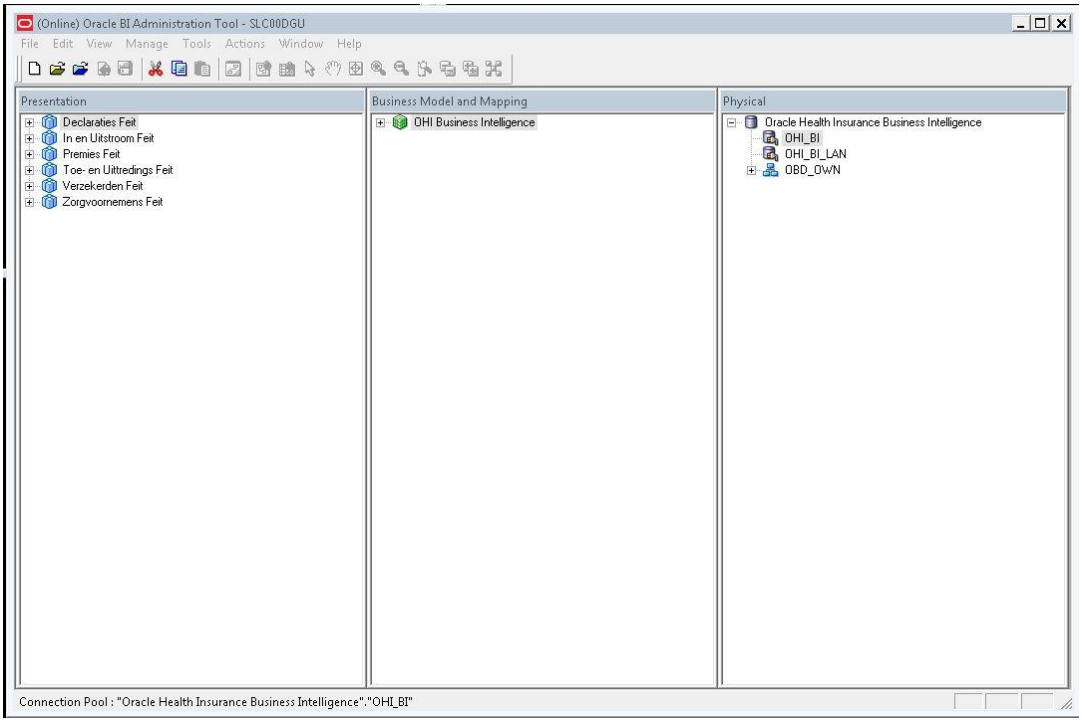
Open the DWH_OHI_BI.rpd offline using the Oracle BI Adminsitration Tool:



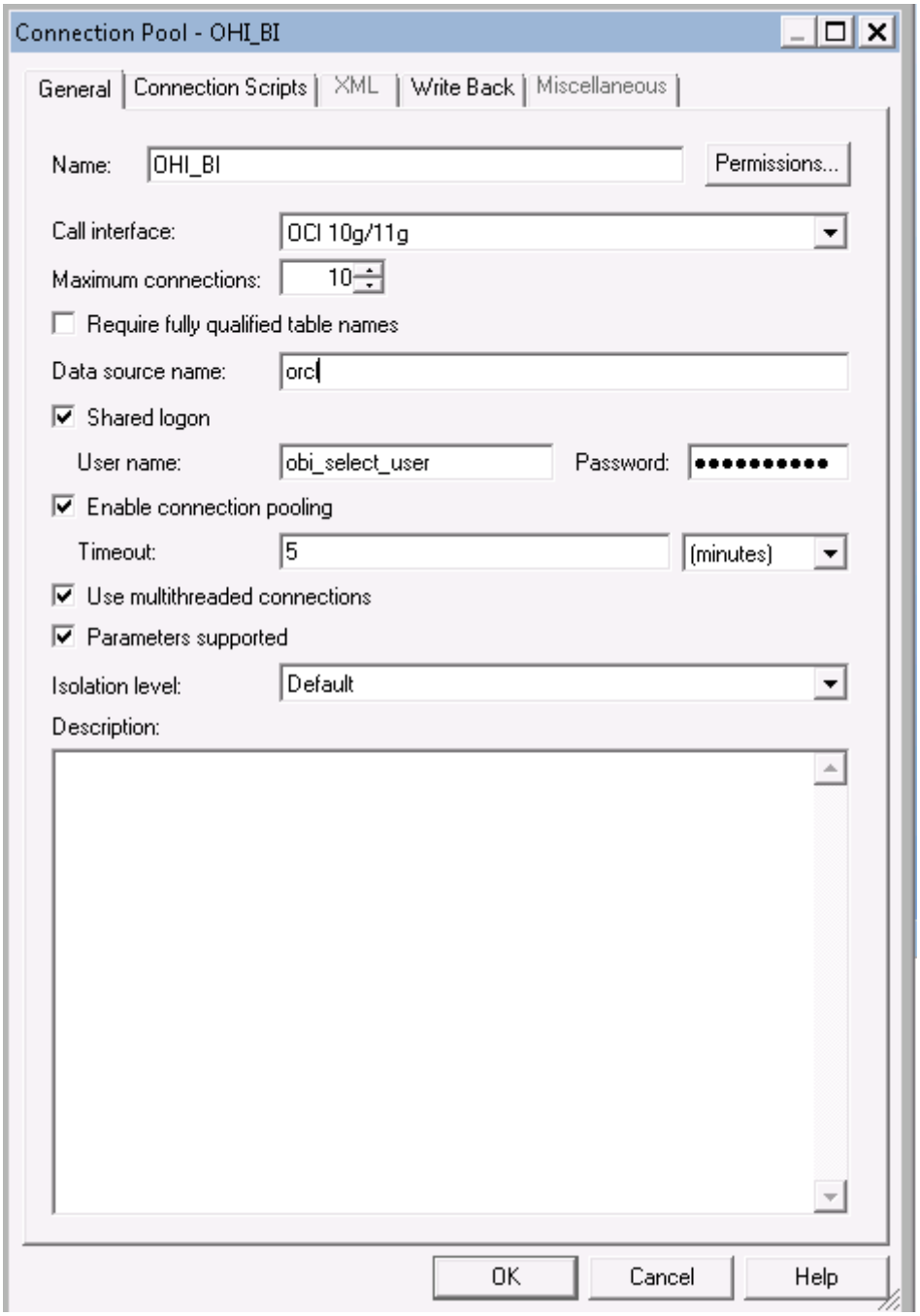
Enter adm1n1strator as the initial password.



Open the Oracle Health Insurance Data Marts node in the Physical layer.



Open the OHI_BI connection pool by double clicking OHI_BI.



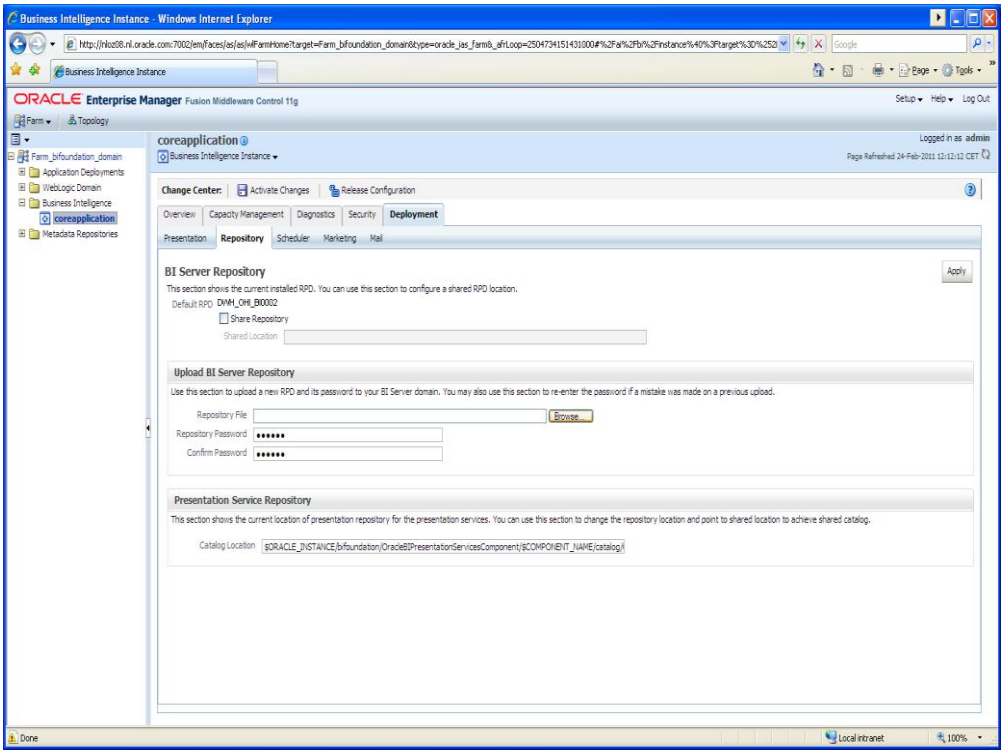
Change the data source name to your OHI Data Marts database.

The Connection pool OHI_BI_LAN is the connection pool used by translations of the repository to enable usage of OBI in your native language. This is not the same connection pool as OHI_BI due to the fact that the translation table may be stored on a different server as opposed to where the Data Marts reside. To configure the connection pool for OHI_BI_LAN, execute the above steps for the OHI_BI_LAN connection pool.

The connection pool for Usage Tracking is by default the repository used by OBI and should be a different Oracle schema as the Data Marts. With security in mind this would also typically be a different database as the database where the Data Marts reside. To configure the connection pool for Usage Tracking please execute the above steps for the OHI_BI_USAGE_TRACKING connection pool, and use the credentials of the OBI repository.

Save the DWH_OHI_BI.rpd.

After this you can import the DWH_OHI_BI.rpd in your OBI EE server environment using the Oracle Enterprise Manager (Fusion Middleware Control 11g) shown below.



In this window, under the heading "Upload BI Server Repository", choose the new DWH_OHI_BI.rpd file and enter the password. Enter the new password again for verification. The new repository will be imported and receives a new sequence number.

Note: this will cause any changes made to the repository to be removed.

CONFIGURE USAGE TRACKING

To configure usage tracking see the documentation in "*Oracle® Fusion Middleware System Administrator's Guide for Oracle Business Intelligence Enterprise Edition*", chapter 9 "*Managing Usage Tracking*".

As a reference below values are an example of Direct Insertion in $ORACLE_INSTANCE//config/OracleBIServerComponent/coreapplication_obis1/NQSConfig.INI

DIRECT_INSERT = YES;
PHYSICAL_TABLE_NAME = "Oracle Health Insurance Data Marts"."OBI_BIPLATFORM"."S_NQ_ACCT";
CONNECTION_POOL = "Oracle Health Insurance Data Marts"."OHI_BI_USAGE_TRACKING";
SUMMARY_STATISTICS_LOGGING = YES;
SUMMARY_ADVISOR_TABLE_NAME = "Oracle Health Insurance Data Marts"."OBI_BIPLATFORM"."S_NQ_SUMMARY_ADVISOR";

UPGRADING ORACLE BUSINESS INTELLIGENCE REPOSITORY

To upgrade the OBIEE repository, the procedure used, is refered to as a "Full Merge With a Common Parent with Binary Repositories". The documentation may be found in "Oracle Fusion Middleware Metadata Repository Builder's Guide for Oracle Business Intelligence Enterprise Edition".

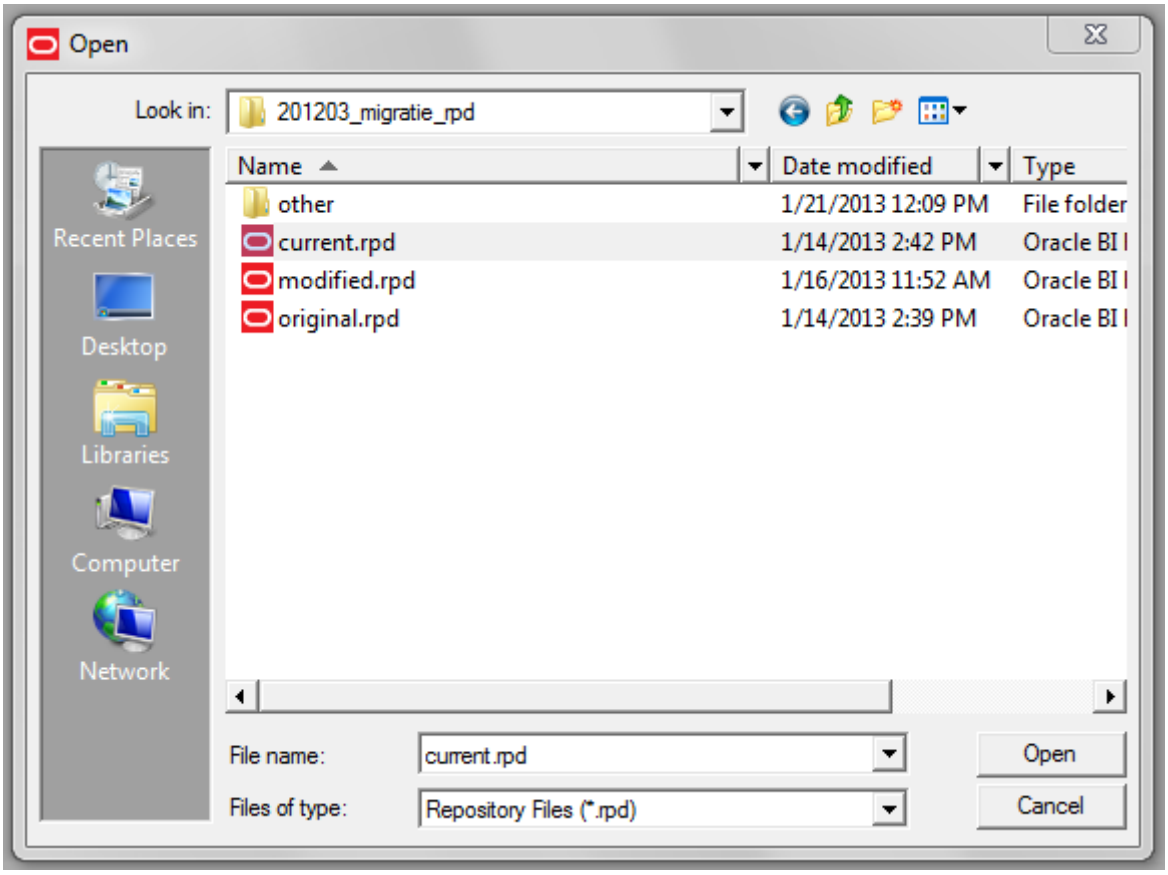
Definitions used in this topology:

- Orignal RPD: The rpd of the release which is migrated from.
  (This is the version provided in the previous release of OHI Data Marts and is unmodified by the Health Insurer, for instance of version 2012.03.01, in the example original.rpd)
- Modified RPD: The rpd of the release which is migrated to.
  (This is the version provided in the new release of OHI Data Marts and is unmodified by the Health Insurer, for instance of version 10.13.1.0.0, in the example modified.rpd)
- Current RPD: The rpd of the release of the customer where is migrated to
  (This is the client version modified i.e. the version of the Health Insurer of the current release for instance 2012.03.01, in the example current.rpd).

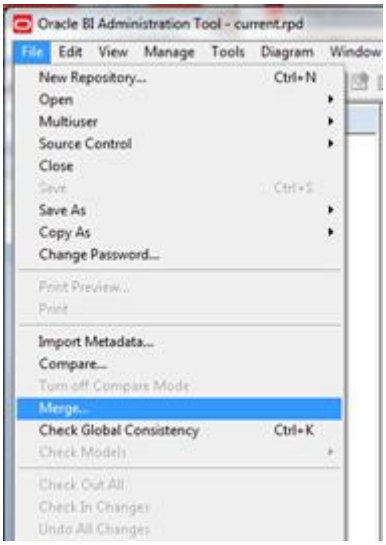
In short the above mentioned procedure needs to be followed.

First Download and Install the Oracle Business Intelligence Developer Client Tools Installer. To download the correct version, please consult the certification matrix
(From OHI BI release 10.13.1.0.0 and further the required version for the Client tool is 11.1.1.6.5).

1. Open the current.rpd.



2. File Merge

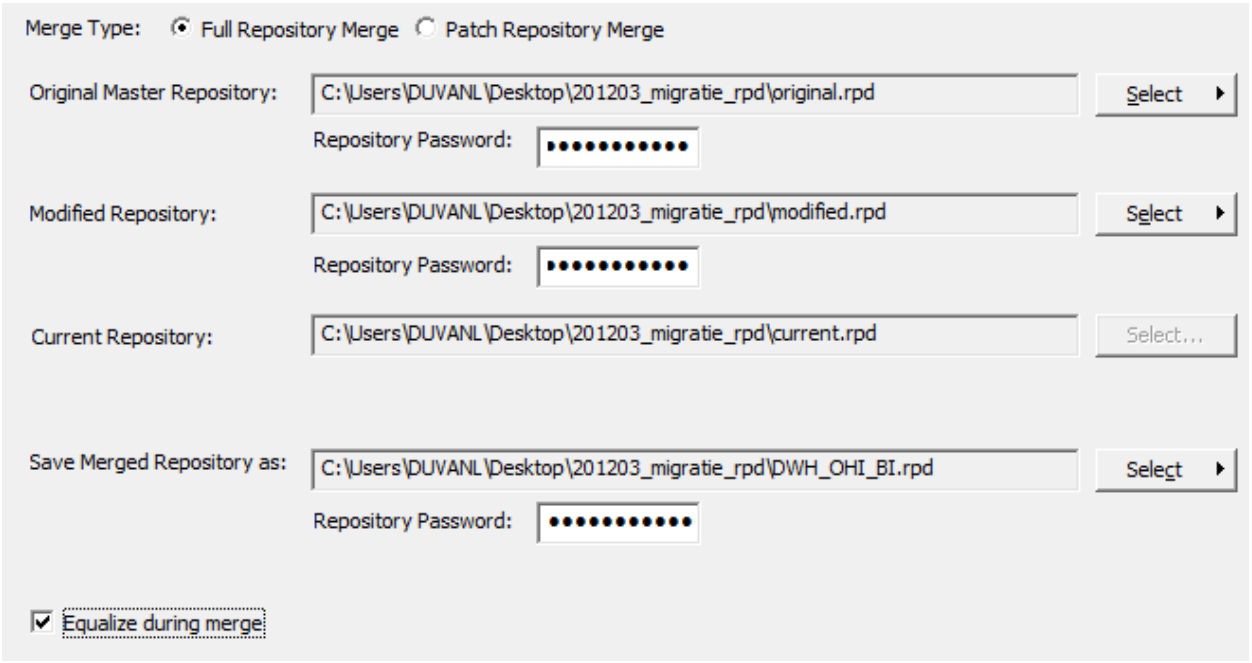


3. Tick equalize during merge

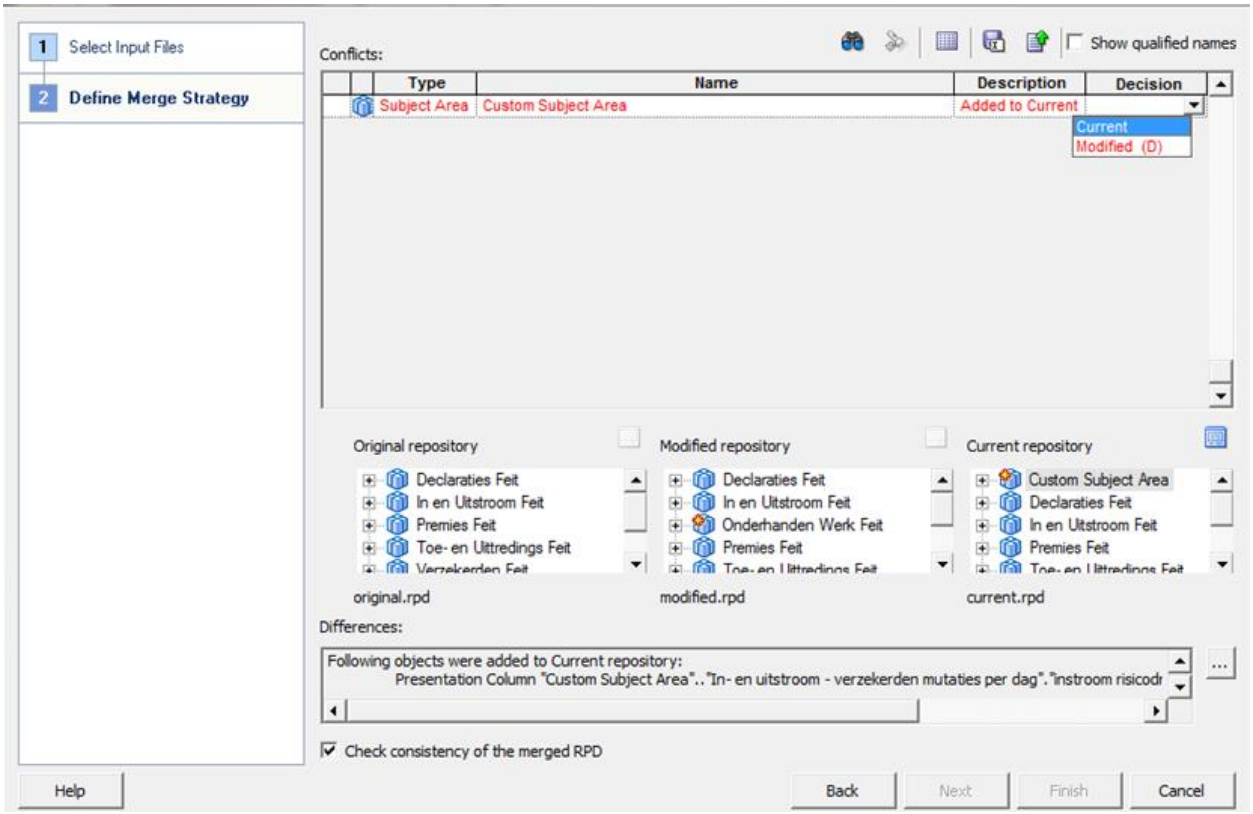
4. Open original and merge rpds

5. Type the passwords for original and modified repositories.

6. Save merged repository as: DWH_OHI_BI.rpd



7. Define Merge strategy:
   Tick Check Consistency of the merged RPD.

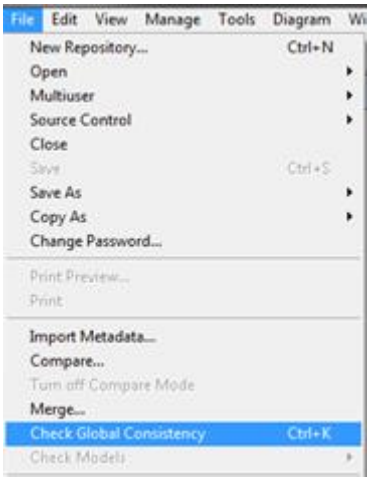


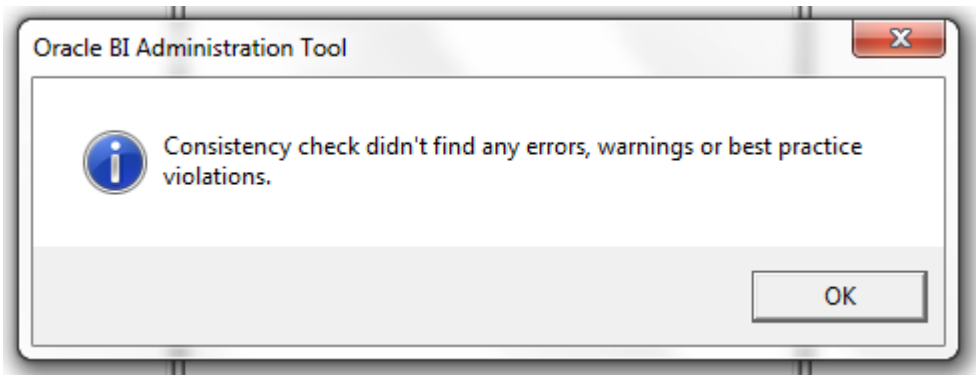
-- choose current to keep the changes in Current repository (Health Insurer RPD)

-- choose modified to keep the change In Modified repository (Oracle RPD) the (D) stands for delete, because this when choosing this it will actually remove the Custom Subject Area.

In the example current is chosen, to keep the Custom Subject Area in the newly to be generated PRD.

8. Review the results of the Consistency Check, and make sure there are no consistency errors.





9. Check the connection details of the connection pools; while merging the two repositories the connection details will be overwritten.
10. Upload the newly created rpd DWH_OHI_BI.rpd to the OBIEE repository see "Upload BI Server Repository".

PART II - APPLICATION MANAGEMENT

INTRODUCTION

This chapter describes aspects of both the technical and functional application management of Oracle Health Insurance Data Marts.

FUNCTIONAL MANAGEMENT

Logging and validation

Validation of the Oracle Health Insurance Data Marts load runs is an important part of the functional/technical management.

Currently, the log information concerning the run load runs can be found in multiple locations. **It is therefore not sufficient just to check the status of the script request in the batch scheduler.** It may be that the status of the script after running the load run is 'Complete', but that errors have in fact occurred. Therefore, the administrator should check the sources of log information described below.

Logging of loading

The following views are present in the obd_own schema to request information on the results of the load run:

WBX_LAADRUNS_VW

This view gives a complete overview of all load runs that have taken place. The following is shown for each load run:

Column	Description
LAADRUN_ID	Unique generated key
RELEASE_NR	Oracle Health Insurance Data Marts release number
SAV_ID_EXT	ID of script request ZRGOE01S.
SAV_ID_STG	ID of script request ZRGOS01S.
SAV_ID_DWH	ID of script request ZRGOD01S.
DECLARATIES_GELADEN	Indication (J/N [meaning Y/N]) of whether the claims fact is loaded in this run.
VERZEKERDEN_GELADEN	Indication (J/N [meaning Y/N]) of whether the policies fact is loaded in this run.
PREMIES_GELADEN	Indication (J/N [meaning Y/N]) of whether the premiums fact is loaded in this run.
TOE_EN_UITTREDINGEN_GELADEN	Indication (J/N [meaning Y/N]) of whether the In- and Out-fluxs fact is loaded in this run.
ZORGVOORNEMENS_GELADEN	Indication (J/N [meaning Y/N]) of whether the care authorizations fact is loaded in this run.
ABONNEMENTSHONORARIUM_GELADEN	Indication (J/N [meaning Y/N]) of whether the per capita agreement fact is loaded in this run.
VERBINTENISSEN_GELADEN	Indication (J/N [meaning Y/N]) of whether the provider relationships fact is loaded in this run.
ONDERHANDEN_WERK_GELADEN	Indication (J/N [meaning Y/N]) of whether the work in progress fact is loaded in this run.
ZORG_INKOOP_GELADEN	Indication (J/N [meaning Y/N]) of whether the procurement agreement fact is loaded in this run.
FINANCIEEL_GELADEN	Indication (J/N [meaning Y/N]) of whether the financial fact is loaded in this run.
DATUM_EXTRACTIE	Date of extraction of the source data
LAADPERIODE_DECLARATIES	Date used for this load as date to for selection of the source data for claims
LAADPERIODE_VERZEKERDEN	Date used for this load as date to for selection of the source data for policies.
LAADPERIODE_PREMIES	Date used for this load as date to for selection of the source data for premiums.
LAADPERIODE_ZORGVOORNEMENS	Date used for this load as date to for selection of the source data for care authorizations.

Column	Description
LAADPERIODE_VERBINTENISSEN	Date used for this load as date from for selection of the source data for provider relationships.
LAADPERIODE_ONDERHANDEN_WERK	Date used for this load as date to for selection of the source data for work in progress.
LAADPERIODE_FINANCIEEL	Date used for this load as date to for selection of the source data for financials.
CONTROLE_UITGEVOERD	Is the loaded data checked?
DOORLOOPTIJD_EXTRACTIE	Run time of the extraction phase
DOORLOOPTIJD_TRANFORMATIE	Run time of the transformation phase
DOORLOOPTIJD_LADEN	Run time of the load phase
DOORLOOPTIJD_TOTAAL	Total run time of the extraction + transformation + load phase
LAADRUN_ID	Unique generated key
RELEASE_NR	Oracle Health Insurance Data Marts release number

WBX_MAPPING_VW

In this view all mappings are shown that are run during a load run. The following information is available:

Column	Description
SAV_ID	The ID of the script request from the OHI Back Office batch scheduler
FASE	Phase in which the load run is found
MAPPING_NAAM	Name of the mapping
MAPPING_GESTART	Time when the mapping was started
AANTAL_VERWERKTE_RIJEN	Number of processed rows
AANTAL_FOUTEN	Number of errors arisen
AANTAL_MINUTEN	Number of minutes the mapping took

WBX_LAADRUN_ERRORS_VW

In this view all errors are shown that have occurred during a load run. This concerns the technical error message that may arise during a mapping. The following information is available:

Column	Description
SAV_ID	The ID of the script request from the OHI Back Office batch scheduler
FASE	Phase in which the load run is found
MAPPING_NAAM	Name of the mapping
TIJDSTIP_FOUT	Time when the error occurred
FOUT_MELDING	Which error occurred

WBX logging

Since there are also a number of non-OWB packages that run during the load run for which no logging exists in the OWB audit logging described above, two logging tables have been added in which logging is also written for the non-OWB packages.

This concerns the tables WBX_LOG_EVENTS and WBX_LOG_MESSAGES (master – detail).

In WBX_LOG_EVENTS the following log data is saved:

Column	Comments
SAV_ID	The ID of the script request from the OHI Back Office batch scheduler
AUDIT_ID	Audit ID of the load run, refers to the ID of the table stg_sys_audit.
SCRIPTNAAM	The code of the script request from the OHI Back Office batch scheduler
STARTTIJD	Start time of the script request
EINDTIJD	End time of the script request
GELADEN_SCHEMAS	The star schemas that have been loaded with this script request
EINDSTATUS	Final status of the script request (Start, Error, Complete)

In WBX_LOG_MESSAGES the following log data is saved:

Column	Comments

Column	Comments
WB_RT_AUDIT_ID	The OWB runtime audit ID (only for mappings generated by OWB)
SAV_ID	The ID of the script request, the FK column to WBX_LOG_EVENTS.
OBJECT_NAAM	The name of the object (package, procedure, ...) which is being logged.
OPMERKINGEN	Potential remarks (step numbers in the case of partitioning)
STARTTIJD	Start time of the object
EINDTIJD	End time of the object

Results of the compatibility checks

The views below provide detailed information on the compatibility checks performed between OHI Back Office and Oracle Health Insurance Data Marts.

WBX_CTR_DECLARATIES_VW

This view shows the results of compatibility checks performed on the claim fact.

Column	Description
SAV_ID	The ID of the script request from the OHI Back Office batch scheduler
CTR_ID	Identification of the check result
CODE	Code of the check
OMSCHRIJVING	Description of the check
DCR_NR	Claim number
VOLGNR	Sequence number of the claim line
DATUM_ACCOORD	The date on which the claim line was approved
PAKKET	The product offered by the coverage
VEL_ID	Identification of the journal entry
GRG_NR	General ledger account number
GBF_NR	General ledger unit number
RUK_NR	Category number
DATUM_IMPORT	Date imported of the liability
RISICODRAGER	The relationship number of the insurance company bearing the risk
BETAALMAAND	The month in which the payment took place
SOORT_BEDRAG	The amount type of the columns below
BEDRAG_BO	Total amount of the claim in Oracle Back Office
BEDRAG_BI	Total amount of the claim in Oracle Health Insurance Data Marts

WBX_CTR_DECL_PRESTATIE_DTLS_VW

This view shows the results of compatibility checks performed on the claim fact.

Column	Description
SAV_ID	The ID of the script request from the OHI Back Office batch scheduler
CTR_ID	Identification of the check result
CODE	Code of the check
OMSCHRIJVING	Description of the check
DCR_NR	Claim number
VOLGNR	Sequence number of the claim line
DATUM_ACCOORD	The date on which the claim line was approved
SOORT	The number type of the columns below
AANTAL_BO	Total number of performed activities/time spent in Oracle Back Office
AANTAL_DM	Total number of performed activities/time spent in Oracle Health Insurance Data Marts

WBX_CTR_PREMIES_VW

This view shows the results of the compatibility check on the premium fact.

Column	Description
SAV_ID	The ID of the script request from the OHI Back Office batch scheduler
CTR_ID	Identification of the check result

Column	Description
CODE	Code of the check
OMSCHRIJVING	Description of the check
DATUM_VA	Start date of the check period
DATUM_TM	End date of the check period
PTL_ID	The unique ID of the premium time line in OHI Back Office
REL_NR	The party number of the member
DATUM_PROLONGATIE	The month for which the renewal was performed
RISICODRAGER	The number of risk bearer
PAKKET	The code of the product
DEKKINGSMAAND	The month for which the coverage applies
SOORT_BEDRAG	Description of the check amount
BEDRAG_BO	The monthly amount of the premium including potential discounts and surcharges in OHI Back Office
BEDRAG_BI	The monthly amount of the premium including potential discounts and surcharges in Oracle Health Insurance Data Marts

WBX_CTR_VERZEKERDEN_VW

This view shows the results of the compatibility check on the policies fact. For all branded product combinations of a member it is checked that this is present in both OHI Back Office and Oracle Health Insurance Data Marts.

Column	Description
SAV_ID	The ID of the script request from the OHI Back Office batch scheduler
CTR_ID	Identification of the check result
CODE	Code of the check
OMSCHRIJVING	Description of the check
PEILDATUM	End date of the check period
CLI_REL_NR	The member which is a member of the policy
MERK_CODE	A unique identifying code for the brand
PAKKET_CODE	The product that is offered
PREMIE_CONSTRUCTIE_CODE	The premium structure that is offered in the product
DEKKING_CONSTRUCTIE_CODE	The unique code of the coverage structure unit
EIGEN_RISICO_CONSTRUCTIE_CODE	The unique code of the yearly deductible structure unit
EIGEN_RISICO_HOOGTE_CODE	The code by which the yearly deductible level is identified
ZORGPLICHT_CODE	The code of the contracted care
AANTAL_BI	Number of memberships in Oracle Health Insurance Data Marts
AANTAL_BO	Number of memberships in OHI Back Office

WBX_CTR_ONDERHANDEN_WERK_VW

This view shows the results of the compatibility checks on the Work in Progress Claims fact.

Column	Description
SAV_ID	The ID of the script request from the OHI Back Office batch scheduler
CTR_ID	Identification of the check result
CODE	Code of the check
OMSCHRIJVING	Description of the check
DATUM_VA	Start date of the check period
DATUM_TM	End date of the check period
DCR_NR	Claim number
VOLGNR	Sequence number of the claim line
SOORT_BEDRAG	Description of the check amount
BEDRAG_BO	Number or amount of the Work in Progress Claim in OHI Back Office
BEDRAG_BI	Number or amount of the Work in Progress Claim in Oracle Health Insurance Data Marts

WBX_CTR_ZORG_VOORNEMENS_VW

This view shows the results of the compatibility checks on the care authorizations fact.

Column	Description
SAV_ID	The ID of the script request from the OHI Back Office batch scheduler
CTR_ID	Identification of the check result
CODE	Code of the check
OMSCHRIJVING	Description of the check
DATUM_VA	Start date of the check period
DATUM_TM	End date of the check period
ZVN_NR	Care authorization identification number
VOLGNR	Sequence number of the period within the care authorization
SOORT_BEDRAG	Description of the check amount
BEDRAG_BO	Number or amount of the care authorization in OHI Back Office
BEDRAG_BI	Number or amount of the care authorization in Oracle Health Insurance Data Marts

WBX_CTR_FINANCIEEL_VW

This view shows the results of the compatibility checks on the financial fact.

Column	Description
SAV_ID	The ID of the script request from the OHI Back Office batch scheduler
CTR_ID	Identification of the check result
CODE	Code of the check
OMSCHRIJVING	Description of the check
DATUM_VA	Start date of the check period
DATUM_TM	End date of the check period
FACTUURGATEGORIE	Financial category
DATUM_AFGEHANDELD	Date completed
SOORT_BEDRAG	Description of the checked amount
BEDRAG_BO	Amount of the payable/receivable in OHI Back Office
BEDRAG_BI	Amount of the payable/receivable in Oracle Health Insurance Data Marts

Logging load scripts

The output of the scripts started through the OHI Back Office application is saved in .out files. These files show how the load run ran, including run times and potential errors. These scripts can be found on the OHI Back Office application server under $OZG_BASE/out/<user>.

<user>: user used to log in to the batch scheduler to start the load run.

Authorization

User access

It is recommended to create a separate account for each user of Oracle Health Insurance Data Marts. This is particularly convenient from a security standpoint.

This account must be created in the Oracle Health Insurance Data Marts database. CREATE SESSION rights must be assigned to the account at database level, as well as the database role OBD_ROL_SELECT. The database role OBD_ROL_SELECT has select rights on all relevant DWH tables and views.

Creating an account in the database and assigning the correct rights/roles can be done in many ways, for example with the following statement in SQL*Plus:

```
create user username identified by password;
grant create session to username;
grant obd_rol_select to username;
```

External tables

The Oracle Health Insurance Data Marts load run makes use of external tables. These are files on the server that are treated as tables by the database. These external tables reside on the Oracle Health Insurance Data Marts database server in the directory referred to by the Unix variable $OZG_ADMIN. The input and output (log and bad files) end up in the directory referred to by the Unix variable $TMP.

Because the database for the external tables has to have a reference to these directories, these directory objects are created in the Oracle Health Insurance Data Marts database.

These files contain data that can be adjusted by the functional administrator, after which these adjustments are made to the Data Warehouse in the next load run.

Configuration of generic structure for claim properties

Introduction

It is possible to register a set of claim properties in OHI Back Office, which can be used for a specific claim type (depending on the type of procedure). When a claim property is required for use in OHI Data Marts it can be added by means of window 'Claim Flex Fields' (ZRGO030F).

Before functionality was used to add the claim property into a generic structure, claim properties were fixed attributes in the data warehouse (fixed properties which were not configurable).

Examples

Example 1
Move claim properties that are already present in the old fixed structure to the generic structure.

The following claim properties of the procedure claim group ‘Tandheelkunde’ are available as fixed columns in OHI Data Marts:

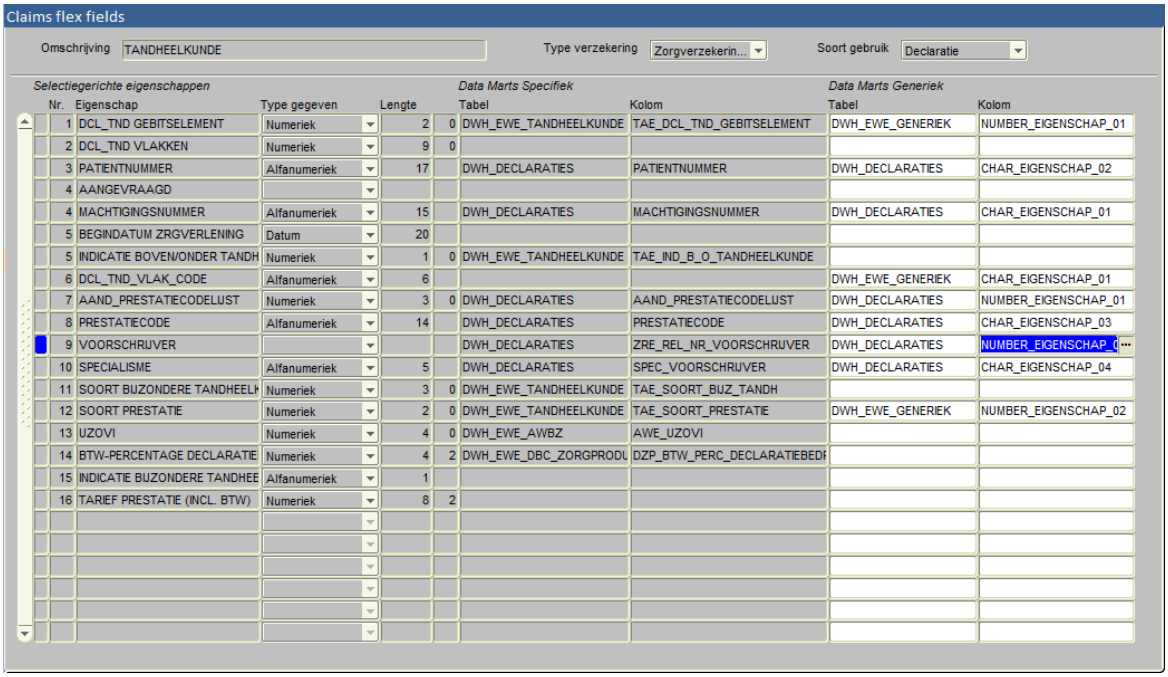
Claim Property	BI table	BI column
Aand. prestatiecodelijst	DWH_DECLARATIES	AAND_PRESTATIECODELIJST
Gebitselement	DWH_EWE_TANDHEELKUNDE	TAE_DCL_TND_GEBITSELEMENT
Vlakcode	DWH_EWE_TANDHEELKUNDE	TAE_DCL_TND_VLAK_CODE
Machtigingsnummer	DWH_DECLARATIES	MACHTIGINGSNUMMER
Patientnummer	DWH_DECLARATIES	PATIENTNUMMER
Prestatiecode	DWH_DECLARATIES	PRESTATIECODE
Soort prestatie	DWH_EWE_TANDHEELKUNDE	TAE_SOORT_PRESTATIE
Specialisme	DWH_DECLARATIES	SPEC_VOORSCHRIJVER
Voorschrijver	DWH_DECLARATIES	ZRE_REL_NR_VOORSCHRIJVER

In this example these attributes are moved into generic claim property columns.

In order to load these claim properties into the generic structure, the mapping to the generic structure must be added.The claim properties can be loaded into the following generic attributes:

Claim Property	Table	Generic Attribute
Aand. prestatiecodelijst	DWH_DECLARATIES	NUMBER_EIGENSCHAP_01
Gebitselement	DWH_EWE_GENERIEK	NUMBER_EIGENSCHAP_01
Vlakcode	DWH_EWE_GENERIEK	CHAR_EIGENSCHAP_01
Machtigingsnummer	DWH_DECLARATIES	CHAR_EIGENSCHAP_01
Patientnummer	DWH_DECLARATIES	CHAR_EIGENSCHAP_02
Prestatiecode	DWH_DECLARATIES	CHAR_EIGENSCHAP_03
Soort prestatie	DWH_EWE_GENERIEK	NUMBER_EIGENSCHAP_02
Specialisme	DWH_DECLARATIES	CHAR_EIGENSCHAP_04
Voorschrijver	DWH_DECLARATIES	NUMBER_EIGENSCHAP_02

In this case the window 'Claims Flex Fields' (ZRGO030F) should look like this:



'Omschrijving' contains the procedure claim group as it is defined in OHI Back Office; in this case this is ‘TANDHEELKUNDE’.

'Data Marts Specifiek, Tabel' contains the location of the old fixed structure. In this example a few properties are part of DWH_DECLARATIES and several others are part of DWH_EWE_TANDHEELKUNDE.

'Data Marts Specifiek, Column' contains the column name in the old structure.

The next two columns provide the mapping to the new generic structure. 'Data Marts Generiek, Tabel' contains the table were the claim property should be loaded and the column 'Data Marts Generiek, Column' contains the attribute where the claim property is loaded.

The value of 'Omschrijving' is loaded into DWH_DECLARATIES (column dwh_declaraties.gebruikgroep). The combination of the generic column and the column dwh_declaraties.gebruikgroep determines the functional meaning of the column. In the abovementioned example, the column dwh_declaraties.char_eigenschap_01 for DWH_DECLARATIES with dwh_declaraties.gebruikgroep is equal to ‘TANDHEELKUNDE’, and contains the claim property MACHTIGINGSNUMMER.

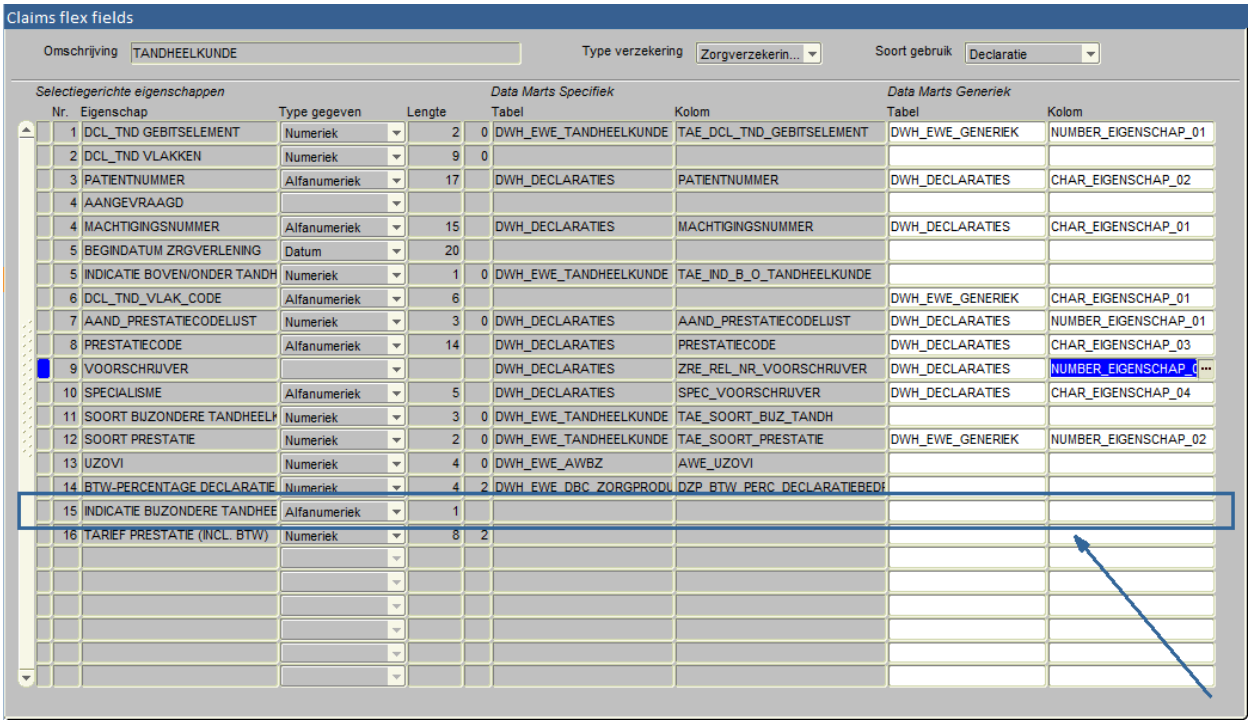
Example 2:

A new claim property ‘INDICATIE_BIJZONDERE_TANDHEELKUNDE’ is introduced in OHI Back Office for procedure claim group ‘TANDHEELKUNDE’, this claim property should be added to OHI Data Marts. The claim property can have two values in OHI Back Office: ‘Y’ or ‘N’.

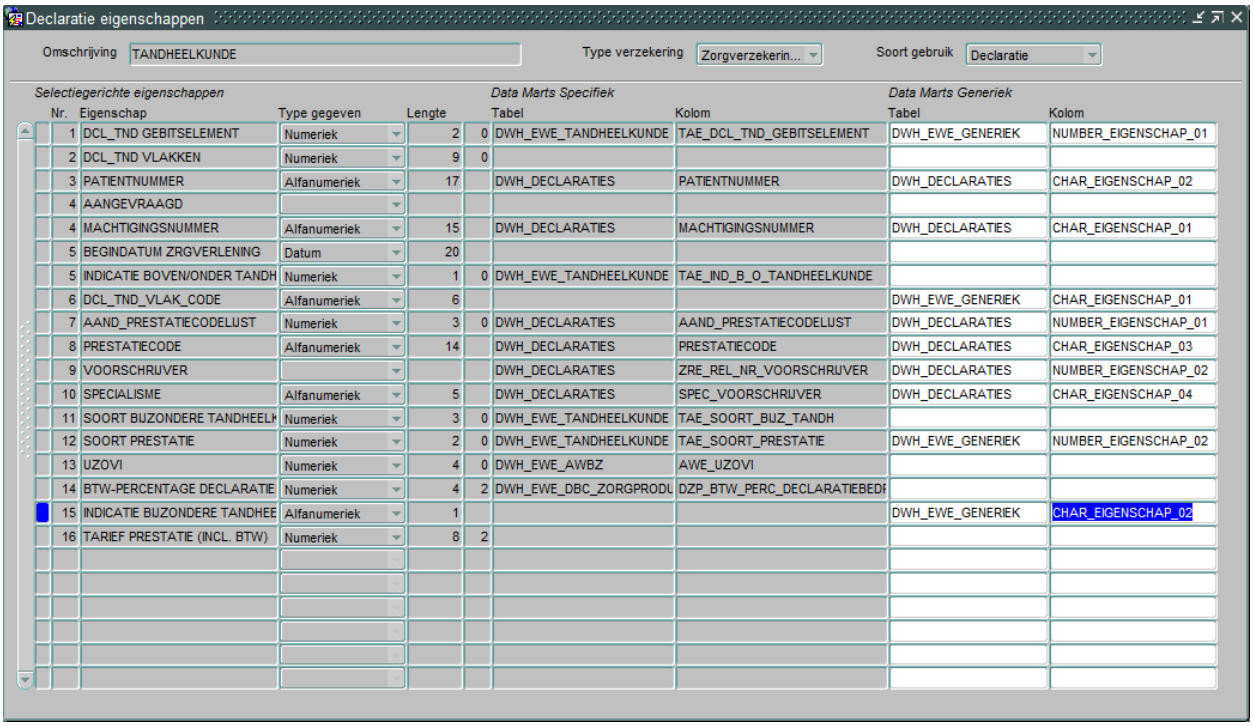
In order to add this claim property, in the window 'Claim Flex Fields' (ZRGO030F) the new property must be altered and a column should be chosen where the claim property ‘INDICATIE_BIJZONDERE_TANDHEELKUNDE’ is loaded into OHI Data Marts.

The first thing to decide is whether this property should be loaded into the dimension table (DWH_EWE_GENERIEK), or into the fact table (DWH_DECLARATIES). When a claim property does not have many different values, it is recommended to save it in the dimension table for storage optimization. In this case there are only two possible values (‘Y’ or ‘N’); therefore this property should be added to the dimension table DWH_EWE_GENERIEK.

The property is a character, in example 1 we already added some properties for the procedure claim group ‘TANDHEELKUNDE; One character claim property is already used for this Procedure Claim Group, namely ‘DCL_TND_VLAK_CODE’. This one is mapped to column CHAR_EIGENSCHAP_01 in table DWH_EWE_GENERIEK. Therefore the new claim property can be added to CHAR_EIGENSCHAP_02, this attribute is not yet used for this procedure claim group. This should be changed in window 'Claim Flex Fields' (ZRGO030F):



The complete window will now look as follows:



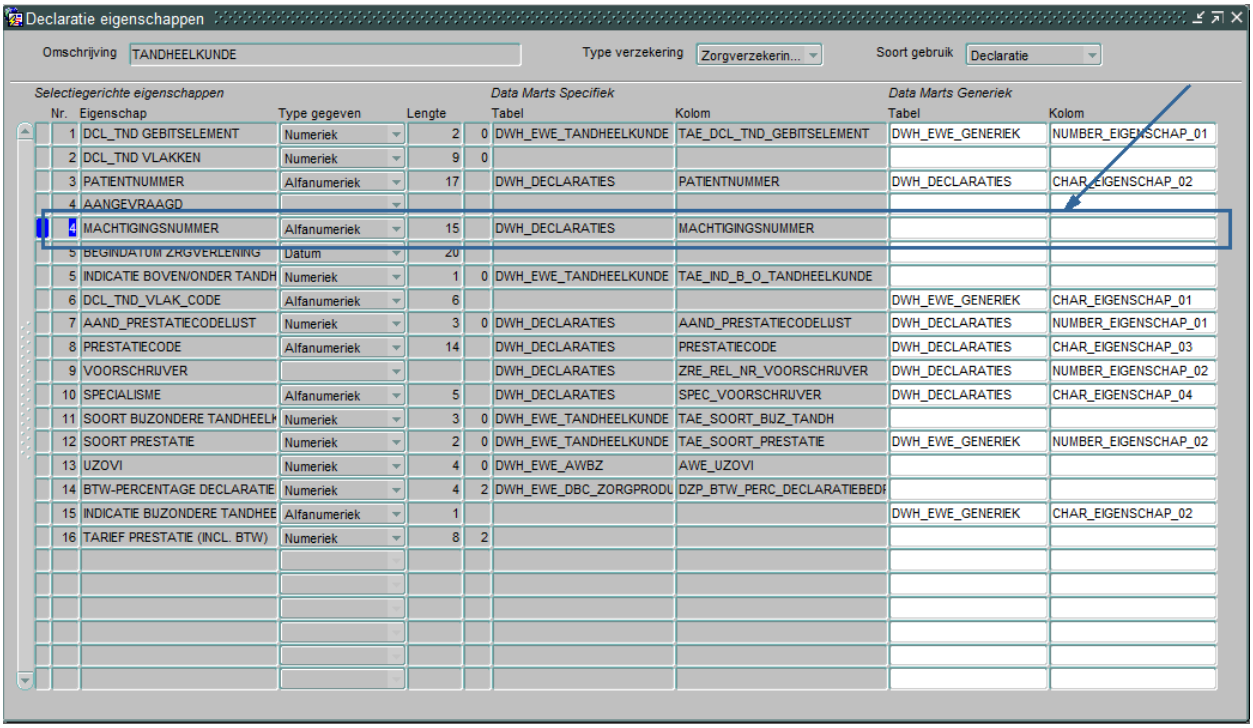
Note that the columns 'Tabel' and 'Kolom' of 'Data Marts Specifiek' are not set up for INDICATIE_BIJZONDERE_TANDHEELKUNDE, this is because this claim property does not exists in the old fixed structure (claim property is added after release 2012.01).

In the situation where both fixed and generic structures are set up, the new records are only loaded in the generic structure; this is due to storage optimization. An exception to this are claim properties that are used for dimension keys, these claim properties are also loaded in the old structure if a definition is available for the generic structure.

Example 3:

Claim property 'MACHTIGINGSNUMMER' for Procedure Claim Group 'TANDHEELKUNDE' should not be loaded in the generic structure but only in the old fixed structure (table: DWH_DECLARATIES, attribute: MACHTIGINGSNUMMER).

In this example, the old fixed attribute for ‘MACHTIGINGSNUMMER’ should be used. The claim property must not be mapped to a generic attribute. Once a property is mapped to a generic attribute, this can not be undone!



Note that only 'Tabel' and 'Kolom' of 'Data Marts Specifiek' are set up for MACHTIGINGSNUMMER.

Also note that this is only relevant for claim properties that were already available in the old fixed structure in OHI Data Marts.

When to place a claim property in a dimension table

There are two places where a claim property can be added in OHI Data Marts: fact table DWH_DECLARATIES and dimension table DWH_EWE_GENERIEK.

It is important to make a good decision where to place a claim property in the generic structure. A claim property should be placed in DWH_DECLARATIES if there are many different potential values for the claim property. For instance the claim property PATIENTNUMMER contains the patient number, this is different for every patient and therefore there are many potential values in OHI Back Office for this claim property. As a result of this it should be placed in DWH_DECLARATIES. If this is placed in DWH_EWE_GENERIEK, the dimension will be very large. A new dimension record will be created for almost every fact record, which results in very insufficient storage optimization.

It can be generally stated that when a claim property has more than a few dozen different values, that it should be placed in the fact table.

Generate views to create a functional overlay over the generic structure

It is possible to generate a view per Procedure Claim Group that contains a fixed set of DWH_DECLARATIES attributes and a dynamic created set of claim properties. In DWH_DECLARATIES and DWH_EWE_GENERIEK there is a generic naming convention (e.g. CHAR_EIGENSCHAP_01) for claim properties. With the view generator it is possible to create a view that contains all claim properties of a Procedure Claim Group, the attributes will obtain the functional name of the claim property instead of the generic name. The view can be used to represent the data with column names that have a functional meaning.

The view can be generated with a procedure that is available in the package WBX_VIEW_GENERATOR. The procedure is called WBX_VIEW_GENERATOR.GENERATE_DECLARATIE_VIEW. This procedure must be executed under the OBD_OWN schema on the OHI Data Marts database. It is mandatory to give a value for the parameter ‘P_GEBRUIKGROEP’. This parameter should contain the exact name of the Procedure Claim Group. For example, the view for the Procedure Claim Group ‘TANDHEELKUNDE’ can be generated with the following statement:

```
exec wbx_view_generator.generate_declaratie_view('TANDHEELKUNDE');
```

This will generate a view named DWH_DCE_TANDHEELKUNDE_VW that can subsequently be used to integrate information into the reporting environment.

Add claim properties from generic structure to OBI EE repository

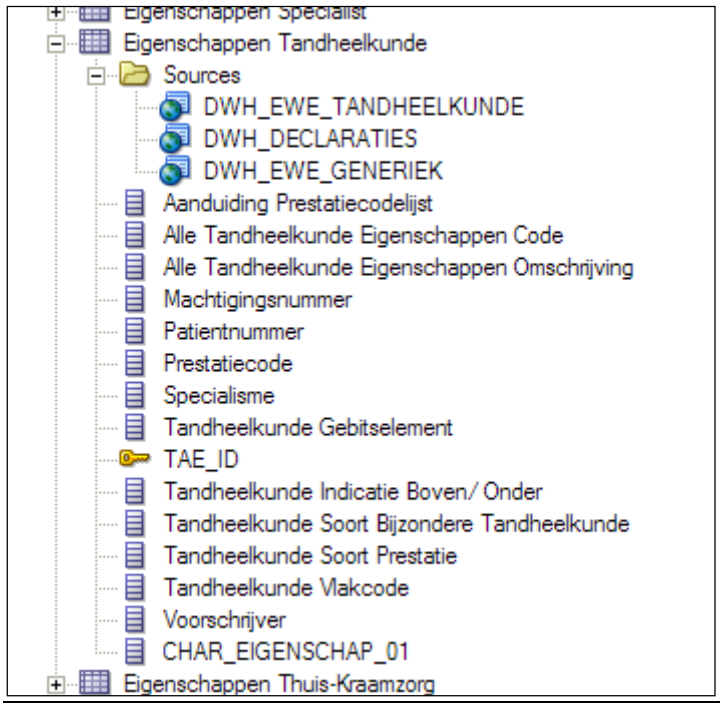
This paragraph describes how to add generically mapped claim properties to the OBI EE repository business model.

All generic claim property attributes are already added to the physical layer of the OBI EE repository.

How a claim property attribute can be added to the business model layer and presentation layer in OBI EE is described in the following example:

In Procedure Claim Group 'Tandheelkunde' the claim property 'EXAMPLE123' is added. This column is mapped to 'CHAR_EIGENSCHAP_01' in DWH_EWE_GENERIEK. The claim property 'EXAMPLE123' should be added to the logical table 'Eigenschappen Tandheelkunde' and should be named 'Example 123'.

1. Add the generic column to the business model
   Drag the attribute 'CHAR_EIGENSCHAP_01' from the Physical Table Source 'DWH_EWE_GENERIEK' to the Logical Table 'Eigenschappen Tandheelkunde'. Once this has been done, DWH_EWE_GENERIEK is automatically added as a Logical Table Source and CHAR_EIGENSCHAP_01 is added to the logical table, represented as follows:



2. Rename the column to the functional meaning
   Change the name of CHAR_EIGENSCHAP_01 to 'Example 123' in the logical table

3. Add the associated Claim Procedure Group
   Due to the fact that generic claim property attributes can have a differenct functional meaning per Claim Procedure Group, it is needed to add information that CHAR_EIGENSCHAP_01 only means 'Example 123' as the Claim Procedure Group is equal to 'TANDHEELKUNDE'. Open the Logical Table Source 'DWH_EWE_GENERIEK'.

   After opening the Logical Table Source 'DWH_EWE_GENERIEK', open the 'General' tab. In pane 'Map to these tables:' the physical table source 'DWH_EWE_GENERIEK' is already there. The attribute 'GEBRUIKGROEP' from 'DWH_DECLARATIES' is also needed, therefore add

   'DWH_DECLARATIES' as physical table by means of selecting the 'Add' button: 

   Add 'DWH_DECLARATIES' as a source

Now open the ‘Column Mapping’ tab. ‘Example 123’ is mapped to CHAR_EIGENSCHAP_01. This is correct, but it only should be mapped as the Claim Property Group is ‘TANDHEELKUNDE’. Therefore we need to change the expression. Open the expression editor for ‘CHAR_EIGENSCHAP_01’ by means of using this button: 

The editor window is shown. CHAR_EIGENSCHAP_01 should be used only if GEBRUIKGROEP is equal to ‘TANDHEELKUNDE’. This can be achieved by means of using this expression:

```
Case when  "Oracle Health Insurance Business Intelligence"."".."OBD_OWN"."DWH_DECLARATIES"."GEBRUIKGROEP" = 'TANDHEELKUNDE'
      then  "Oracle Health Insurance Business Intelligence"."".."OBD_OWN"."DWH_EWE_GENERIEK"."CHAR_EIGENSCHAP_01"
      else  null
end
```

Add the new attribute to the presentation layer
Add ‘Example 123’ to the Presentation table ‘- Eigenschappen Tandheelkunde’

Check in the changes, save the repository and the claim property will now be available for end-users.

Configuration of generic structure for dimension properties

Introduction

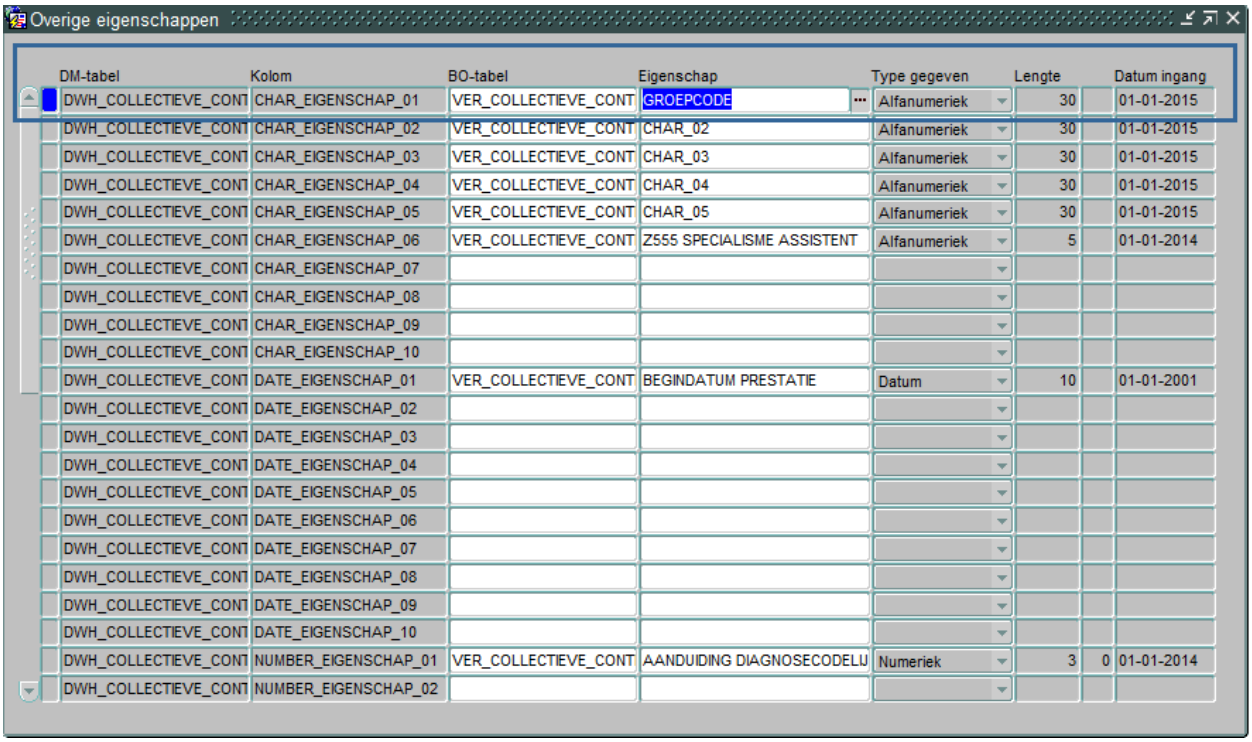
It is possible to register a set of dimension properties in OHI Back Office. When a dimension property is required for use in OHI Data Marts it can be added by window 'Other Flex Fields' (ZRGO031F).

Examples

Example
A new dimension property 'GROEPCODE' is introduced in OHI Back Office for group contracts, this dimension property should be added to OHI Data Marts.

In order to add this dimension property, th window 'Other Flex Fields' (ZRGO031F) should be used. Query on 'DWH_COLLECTIEVE_CONTRACTEN' in field 'DM-tabel'. The new property must be added and a column should be defined where the dimension property 'GROEPCODE' is loaded into OHI Data Marts.

The property is a character and will be mapped to column CHAR_EIGENSCHAP_01 in table DWH_COLLECTIEVE_CONTRACTEN. The following line should be altered:



DM-tabel	Kolom	BO-tabel	Eigenschap	Type gegeven	Lengte		Datum ingang
DWH_COLLECTIEVE_CONT	CHAR_EIGENSCHAP_01	VER_COLLECTIEVE_CONT	GROEPCODE	Alfanumeriek	30		01-01-2015
DWH_COLLECTIEVE_CONT	CHAR_EIGENSCHAP_02	VER_COLLECTIEVE_CONT	CHAR_02	Alfanumeriek	30		01-01-2015
DWH_COLLECTIEVE_CONT	CHAR_EIGENSCHAP_03	VER_COLLECTIEVE_CONT	CHAR_03	Alfanumeriek	30		01-01-2015
DWH_COLLECTIEVE_CONT	CHAR_EIGENSCHAP_04	VER_COLLECTIEVE_CONT	CHAR_04	Alfanumeriek	30		01-01-2015
DWH_COLLECTIEVE_CONT	CHAR_EIGENSCHAP_05	VER_COLLECTIEVE_CONT	CHAR_05	Alfanumeriek	30		01-01-2015
DWH_COLLECTIEVE_CONT	CHAR_EIGENSCHAP_06	VER_COLLECTIEVE_CONT	Z555 SPECIALISME ASSISTENT	Alfanumeriek	5		01-01-2014
DWH_COLLECTIEVE_CONT	CHAR_EIGENSCHAP_07						
DWH_COLLECTIEVE_CONT	CHAR_EIGENSCHAP_08						
DWH_COLLECTIEVE_CONT	CHAR_EIGENSCHAP_09						
DWH_COLLECTIEVE_CONT	CHAR_EIGENSCHAP_10						
DWH_COLLECTIEVE_CONT	DATE_EIGENSCHAP_01	VER_COLLECTIEVE_CONT	BEGINDATUM PRESTATIE	Datum	10		01-01-2001
DWH_COLLECTIEVE_CONT	DATE_EIGENSCHAP_02						
DWH_COLLECTIEVE_CONT	DATE_EIGENSCHAP_03						
DWH_COLLECTIEVE_CONT	DATE_EIGENSCHAP_04						
DWH_COLLECTIEVE_CONT	DATE_EIGENSCHAP_05						
DWH_COLLECTIEVE_CONT	DATE_EIGENSCHAP_06						
DWH_COLLECTIEVE_CONT	DATE_EIGENSCHAP_07						
DWH_COLLECTIEVE_CONT	DATE_EIGENSCHAP_08						
DWH_COLLECTIEVE_CONT	DATE_EIGENSCHAP_09						
DWH_COLLECTIEVE_CONT	DATE_EIGENSCHAP_10						
DWH_COLLECTIEVE_CONT	NUMBER_EIGENSCHAP_01	VER_COLLECTIEVE_CONT	AANDUIDING DIAGNOSECODELIJ	Numeriek	3	0	01-01-2014
DWH_COLLECTIEVE_CONT	NUMBER_EIGENSCHAP_02						

Generate views to create a functional overlay over the generic structure

It is possible to generate a view per dimension that contains all attributes of the dimension. For the generic dimension properties there is a generic naming convention (e.g. CHAR_EIGENSCHAP_01). With the view generator it is possible to create a view that contains all dimension properties, the attributes will obtain the functional name of the dimension property (as defined in 'Other Flex Fields' (ZRGO031F)) instead of the generic name. The view can be used to represent the data with column names that have a functional meaning.

The view can be generated with a procedure that is available in the package WBX_VIEW_GENERATOR. The procedure is called WBX_VIEW_GENERATOR.GENERATE_DIMENSIE_VIEW. This procedure must be executed under the OBD_OWN schema on the OHI Data Marts database. It is mandatory to give a value for the parameter 'P_DIMENSIE_TABEL'. This parameter should contain the exact name of the dimension table. For example, the view for the group contract dimension 'DWH_COLLECTIEVE_CONTRACTEN' can be generated with the following statement:

```
exec wbx_view_generator.generate_dimensie_view('DWH_COLLECTIEVE_CONTRACTEN');
```

This view can subsequently be used to integrate information into the reporting environment.

Add dimension properties from generic structure to OBI EE repository

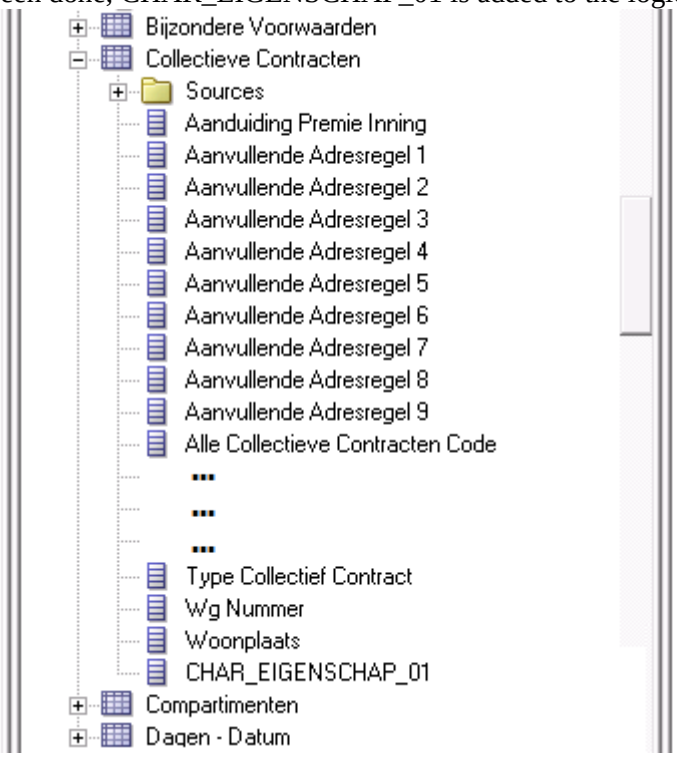
This paragraph describes how to add generically mapped dimension properties to the OBI EE repository business model.

All generic dimension property attributes are already added to the physical layer of the OBI EE repository.

How a dimension property attribute can be added to the business model layer and presentation layer in OBI EE is described in the following example:

In the Group Contract dimension the property 'GROEPCODE' is added. This column is mapped to 'CHAR_EIGENSCHAP_01' in DWH_COLLECTIEVE_CONTRACTEN. The dimension property 'GROEPCODE' should be added to the logical table 'Collectieve Contracten' and should be named 'Groep Code'.

4. Add the generic column to the business model
   Drag the attribute 'CHAR_EIGENSCHAP_01' from the Physical Table Source 'DWH_COLLECTIEVE_CONTRACTEN' to the Logical Table 'Collectieve Contracten'. Once this has been done, CHAR_EIGENSCHAP_01 is added to the logical table, represented as follows:



5. Name the column to the functional meaning
   Change the name of CHAR_EIGENSCHAP_01 to 'Groep Code' in the logical table

6. Add the new attribute to the presentation layer
   Add 'Groep Code' to the Presentation table 'Collectieve Contracten'

   Check in the changes, save the repository and the claim property will now be available for end-users.

PART III - TECHNICAL MANAGEMENT

CLEANING UP OWB AUDIT DATA

As described earlier, OWB generates audit information while running the OWB mappings. This information is stored in the run time tables of the Oracle Health Insurance Data Marts repository schema OWBSYS (WB_RT_% tables). These tables are located in the OWBSYS tablespace. Every time that it's loaded this tablespace grows.

To avoid this table space becoming too large it is possible to delete (part of) the audit data. A package is available for this in the OWBSYS schema. This is the package ***WB_RT_API_PURGE*** that contains a number of 'purge' procedures that can remove data in a variety of ways:

`purge_execution`	`All audit data is cleaned`
`purge_execution (exec_id)`	`Audit data with rte_id = exec_id is cleaned`
`purge_execution (start_date, end_date)`	`All data from the period between the start and end date is cleaned`

This audit data forms an important source of information for how a load run has run and for resolving problems. It is therefore important to consider carefully whether the information is genuinely no longer needed before running this procedure.

This data can also be removed using the script purge_audit_template.sql in the directory $ORACLE_HOME/owb/rtp/sql/.

NEW RELEASES OF ORACLE HEALTH INSURANCE DATA MARTS

When new releases of Oracle Health Insurance Data Marts are brought out new versions of this documentation will also be supplied via iProjects files.

The Oracle Health Insurance installation menu OHIPATCH must be used for the installation of new Oracle Health Insurance Data Marts releases or patches. For operation of the installation menu: see document ‘PCM02101.pdf’ (Oracle Health Insurance Installation of Releases).

REORGANIZATION OF TABLES

It is recommended to regularly (depending on the load frequency, e.g. once per quarter/half year) reorganize the Data Warehouse facts tables and indexes on facts tables.

For the partitioned facts tables:

- DWH_AFGEWEZEN_DECLARATIES
- DWH_DECL_PRESTATIE_DETAILS
- DWH_DECLARATIES
- DWH_DECLARATIES_AGG
- DWH_DEELNAME_MUTATIES
- DWH_ONDERHANDEN_WERK
- DWH_PREMIES
- DWH_VERBINTENISSEN
- DWH_VERPLICHTINGEN
- DWH_VERZEKERDEN

- DWH_VORDERINGEN
- DWH_ZORG_VOORNEMENS

this can be done using:

- ALTER TABLE [table name] MOVE PARTITION [partition name]
- ALTER INDEX [index name] REBUILD PARTITION [partition name]

For the non-partitioned facts table:

- DWH_OPENSTAANDE_VERPLICHTINGEN
- DWH_OPENSTAANDE_VORDERINGEN
- DWH_TOE_UITTREDINGEN

this can be done using:

- ALTER TABLE [table name] MOVE
- ALTER INDEX [index name] REBUILD

COMPRESSION OF PARTITIONED FACT TABLES.

As of Oracle Health Insurance Data Marts version 2011.03 it is possible to compress the partitioned tables. By compressing large fact tables, a large amount of disk space can be saved.

Step '850 - Partition/compress tables' in OHIPATCH.pl is available for this. This choice is the same for OHI Back Office and Oracle Health Insurance Data Marts, however the following submenu choices are only shown for Oracle Health Insurance Data Marts.

```
INFO : ==========================================
INFO : = Redefine a table:                       =
INFO : = O - Compress Advanced                   =
INFO : = Q - Compress for Query (Exadata only)   =
INFO : = A - Compress for Archive (Exadata only) =
INFO : = U - Uncompress  (revert compression)    =
INFO : ==========================================
INFO : Which table redefinition would you like to execute (A, A, Q, U)?
```

If one of the compress actions (O, Q or A) is chosen, then you see a list of tables that can be compressed. By typing in the table name the choice is confirmed and the (sub)partitions of the table are compressed, including local partitioned indexes. For large fact tables this may take a long time. After compression, all local partitioned indexes are no longer usable and have to be rebuilt. Option '870 - Rebuild unusable indexes' in OHIPATCH.pl is available for this. Rebuilding these 'unusable' indexes can also take a long time if the indexes concern large facts tables.

If 'U'ncompress (revert compression) is chosen, then you see a list of tables that can be uncompressed. The rest of the procedure is the same as for compressing tables. After uncompressing tables, the local partitioned tables must also be rebuilt.

Note: Ensure that there is enough disk space available for compression and uncompression.

OWB RUNTIME SERVICE

As of Oracle Health Insurance Data Marts version 2010.01, use is made of OWB11g. As a consequence, as well as the Oracle Health Insurance Data Marts OBD_OWN schema, an OWB Runtime repository schema OWBSYS and a workspace owner OHI_BI_WS_OWN are also available for the administration of the installation and execution of OWB mappings (see Server installation).

To enable performance of this administration, the repository makes use of the OWB Runtime Service. This is a process on the server that is started by the OS user oracle when starting the Oracle Health Insurance Data Marts database.

There are a number of scripts available relating to the management of this runtime service (in $ORACLE_HOME/owb/rtp/swl) and these can be run in SQLPlus under the OWBSYS account:

show_service.sql	This script shows the current status of the service.
start_service.sql	This script starts the service.
stop_service.sql	This script stops the service.
service_doctor.sql	This script can be used to perform a diagnosis of the service if it will not start.
reset_repository.sql	Sets the password of the repository owner (OWBSYS) and initializes a number of repository-specific values.

See the header of these scripts for a more detailed explanation of their use.

PART IV - APPENDICES

APPENDIX A: CLONING ORACLE HEALTH INSURANCE DATA MARTS ENVIRONMENTS

From a management standpoint, it is sometimes necessary to make a copy of an Oracle Health Insurance Data Marts environment and place it in another environment, for example to make a production environment available on a test environment.

An Oracle Health Insurance Data Marts environment consists of the repository schema OWBSYS, the OWB workspace owner OHI_BI_WS_OWN and the Oracle Health Insurance Data Marts schema OBD_OWN.

Seeing as environment-specific information is stored in the repository, a number of things have to be done after the transfer to adjust this connection information for the new environment to ensure that everything continues to work correctly.

This environment-specific information includes:

- Connection information for the repository
- Connection information relating to the registered locations

For a description of the cloning of an Oracle Health Insurance Data Marts environment there is a page available on My Oracle Support. See:

How To Update Warehouse Builder After A Database Cloning [ID 434272.1]

This also refers to a Java based tool for correcting the locations used by Oracle Health Insurance Data Marts:

How To Use The OWB 11.2.0.3 Java Tool oracle.wh.util.locationhelper.LocationTool for Failover [ID 1362745.1]

APPENDIX B: USE OF WBX_LAADRUN_ERRORS_VW AND WBX_MAPPING_VW VIEWS.

A number of extra steps are required to use views WBX_LAADRUN_ERRORS_VW and WBX_MAPPING_VW. This is because, in contrast to older OWB versions, from OWB 11gR2 makes use of workspaces. The WBX_LAADRUN_ERRORS_VW and WBX_MAPPING_VW views calculate OWB public views in the OWBSYS schema that is subdivided into workspaces. Before the view is calculated it must be specified for which workspace the view should be used. This is done as follows:

Extra grants must be assigned from OWBSYS to OBD_OWN:

grant select on wb_rt_audit to obd_own with grant option;
grant select on wb_rt_errors to obd_own with grant option;

Users who want to calculate the view should be set up as workspace users (of workspace OHI_BI_WS) using the OWB Repository assistant (reposinst.sh).

Therefore before the view is used, the following command has to be run:

Execute owbsys.wb_workspace_management.set_workspace('OHI_BI_WS','OHI_BI_WS_OWN');

This can potentially be solved using a LOGON trigger:

```
create or replace trigger gebruiker_logon_trigger
   after logon on gebruiker.schema
begin
owbsys.wb_workspace_management.set_workspace('OHI_BI_WS','OHI_BI_WS_OWN');
end;
```

For this, an explicit grant has to first be assigned to 'user' from OWBSYS:

grant execute on wb_workspace_management to user;

In the example above replace 'user' with the user that wishes to use the named views.

APPENDIX C: EXADATA USAGE

When migrating to Exadata bear in mind the following migration path.

1. Please refer to chapter C: "*Converting to Oracle RAC and Oracle Rac One Node from Single Instance Oracle Database.", "Oracle Real Application Clusters Installation Guide 11g Release 2".*
2. Please refer to chapter 8: "*Performing Oracle ASM Data Migration with RMAN*", "*Oracle Automatic Storage Management Administrator's Guide*."

After migration steps

Check to see if owbsys or obd_own accounts are locked

Sqlplus obd_own/<PASSWORD>

select username,account_status,lock_date,expiry_date from dba_users where username in ('OWBSYS','OBD_OWN','OWBRT_SYS','OHI_BI_WS_OWN');

If the owbsys repository need to be reset run: $ORACLE_HOME/owb/rtp/sql/reset_repository.sql)

For example:
-- reset home
-- all homes on exadata should be located here:
update wb_rt_service_nodes
set server_side_home = '/u01/app/oracle/product/11.2.0.3/dbhome_1';
--
commit
$ORACLE_HOME/owb/UnifiedRepos/reset_owbcc_home.sql
</u01/app/oracle/product/11.2.0.3/dbhome_1>

$ORACLE_HOME/owb/rtp/sql/reset_repository.sql
<Password for OWBSYS>

Check to see if locations are registered

```
BEGIN
  sys_mapping_util_pck.set_workspace;
END;
/

SELECT object_name
FROM user_objects
WHERE object_type = 'PACKAGE BODY'
AND SUBSTR(object_name,1,3) IN ('DWH','STG')  -- Destination
AND SUBSTR(object_name,9,3) IN ('DWH','STG','SRC','EXT') -- Source
MINUS
SELECT wo.object_name FROM owbsys.wb_rt_warehouse_objects wo ORDER BY 1;
/
```

(should not show any rows, if any redo migration, do not unregister locations)

Check to see if OWB is running

select sid,serial#,username,status,server,machine,module from gv$session where module = 'OWB_CCS';

If there is a module OWB_CCS then shutdown owb by running:

Sqlplus OWBSYS/<PASSWORD>

@$ORACLE_HOME/ owb/rtp/sql/stop_service.sql

In release 101201 for Open Beleid the proper execute rights should be granted to
Exadata DataMachine machine (755)
$ORACLE_HOME/owb/bin/unix/OMBPlus.sh

$ORACLE_HOME/owb/bin/unix/setowbenv.sh
$ORACLE_HOME/owb/bin/unix/ombplus
If this is not set correctly patching will fail with a permission denied error

3. Make sure the DB link SRC_OPENZORG.WORLD point to the correct Open Zorg environment.

4. Grant permissions as per administrator reference on the Open Zorg Database.

```
Sqlplus system/password@BO
grant create session to obd_select_user;
Grant alter session to obd_select_user;
Grant ozg_rol_select to obd_select_user;
Grant select, insert, delete on ozg_owner.geb#obd_declaraties to obd_select_user;
Grant execute on ozg_owner.alg_tab_pck  to obd_select_user;
Grant execute on ozg_owner.fin_fpm_vars_pck to obd_select_user;
Grant select on v_$database to obd_select_user;
Grant execute on ozg_owner.geb_odr_pck to obd_select_user;
```

5. Make sure the following patchset is installed on the Exadata Database Machine: 16568042
Note 1: that this patchset is installed in 2012.01, however if the physical machine changes, the software of new and old machines need to be identical.
Note 2: please note that no SQL statements of the patch need to be applied: they should be already applied in 2012.01 (ie only run Opatch apply)

6. Make sure java and xmldb are installed.

7. For multi node RAC: Apply the steps described in “Migrating OWB from Single Instance to **multi node RAC** “
For single instance: Apply the steps described in “OWB 11gR2 post-cloning process for OHI Data Marts”

8. Gather schema statistics for OBD_OWN.

9. Make sure the tnsnames.ora entries on application server are pointing to the correct environment for Open Beleid as well as Open Zorg.

10. Recompile schema OBD_OWN, after which make sure no invalid objects are found in schema OBD_OWN

```
begin
dbms_utility.compile_schema( schema  => user , compile_all   =>TRUE,  reuse_settings =>TRUE);
end;

select count(*) from user_objects where status != 'VALID';
```

11. Create wallet entries on Application server for 3 users, and display mkstore is similar to below on the application server.
mkstore –wrl . -listCredential
<db> batch
<db>_batch batch
<db>_install obd_own
Also make sure there is a tnsnames entry for
<db>
<db>_batch
<db>_install
To check for multi instance rac node (setup 1, non preferred setup):
mkstore –wrl . –listCredential
oton1 batch
oton1_batch batch
oton1_install obd_own

where oton1 is the service_name which is used for extraction

sqlplus /@oton1
sqlplus /@oton1_batch
sqlplus /@oton1_install
should all be able to connect

To check for multi instance rac node (setup 2, preferred setup) or single instance:

mkstore –wrl . –listCredential
oton batch
oton_batch batch
oton_install obd_own

where oton is the service_name which is used for extraction

sqlplus /@oton
sqlplus /@oton_batch
sqlplus /@oton_install
should all be able to connect

12. Make sure the input files of the External tables, are in the OBD_INPUT location, if not place them in OBD_INPUT directory and make sure oracle has read, write permissions on that directory.

13. Make sure there is a batch scheduler running on the application server.

Note For **Multi node Rac**: To transfer the service to another node:
srvctl modify service -d db_unique_name -s service_name -i old_instance_name -t new_instance_name [-f]
also please note that you might need a batch scheduler on the other node, and tnsnames entry on the second node of the first service to be able to connect to the relocated service.

Note for Exadate: the use of a IORM plan is highly recommendable, so is setting the limit clause for a consistent performance experience see MOS “Configuring Exadata I/O Resource Manager for Common Scenarios [ID 1363188.1]”

The database parameter parallel_degree_policy = auto takes care of maximum utilization of resources. Resource Management is about maximum consumption of resources and acquiring the maximum resources. If there are for instance 2 databases A and B, and the total available resources is 100. Then maximum utilization of resources for A means utilizing 100 of that resources; but for B then there are no resources left.

The manual distribution of resource, for instance A can consume 80% of the resources and B 20% of the resources, is an example of IOResource Management, and should make sure B can also aquire their maximum of resources of 20%.

In ideal situation IOResources Management is assigned and utilized to a maximum.

Therefore an IOResource Management Plan is recommended to prevent a case where patching an OHIBI installation takes significcant amount of time, since for example all resources are taken by A, acquiring resources failed.

When statements run in parallel but they could in potential have a higher degree of parallism, maximum utilization of resources failed.

The first situation if more concerning than the latter, therefore an IOResourcemanagement plan is highly recommended. Setting the parallel_degree_policy = auto is optional, but only supported for the ETL user OBD_OWN and not for the OBD_SELECT_USER, and should be implemented with an after logon trigger as shown below. Reason for this is that statement queing will occur.

```
create or replace trigger trg_parallel_etl after logon on database
begin
if user = 'OBD_OWN' then
execute immediate 'alter session set parallel_degree_policy=AUTO';
execute immediate 'alter session enable parallel dml';
end if;
end;
/
```

For DOP to work properly IO Calibration is needed. This can be achieved by executing the one time only process `DBMS_RESOURCE_MANAGER.CALIBRATE_IO, failure to do so would lead to the message "` *automatic DOP: skipped because of IO calibrate statistics are missing" in the explain plan. The example below is derived from the documentation*:

http://docs.oracle.com/cd/E11882_01/appdev.112/e10577/d_resmgr.htm#CJGHGFEA

Example of using I/O Calibration procedure

```
SET SERVEROUTPUT ON
DECLARE
  lat  INTEGER;
 iops INTEGER;
 mbps INTEGER;
BEGIN
-- DBMS_RESOURCE_MANAGER.CALIBRATE_IO (<DISKS>, <MAX_LATENCY>, iops, mbps, lat);
DBMS_RESOURCE_MANAGER.CALIBRATE_IO (2, 10, iops, mbps, lat);
   DBMS_OUTPUT.PUT_LINE ('max_iops = ' || iops);
   DBMS_OUTPUT.PUT_LINE ('latency  = ' || lat);
   DBMS_OUTPUT.PUT_LINE ('max_mbps = ' || mbps);
end;
/
While running
```

APPENDIX D: OWB 11GR2 POST-CLONING PROCESS FOR OHI DATA MARTS

The post-cloning process means the steps to be executed after the files of a shutdown of an Oracle Health Insurance Data Marts database have been copied and started up again as another database.

This document describes the post-cloning process of an Oracle Health Insurance Data Marts (a.k.a. OHI Data Marts) environment running on an Oracle 12c Release 1 database and Oracle Warehouse Builder 11g Release 2 (11.2.0.4).

It is presumed that the database and owb software for the database to be cloned is in the same Oracle Home as the database that is used for cloning. If this is not the case than the following steps need to be executed first:

– connect as sysdba

– execute reset_owbcc_home.sql in $OWB_HOME/UnifiedRepos
when asked for the control center home, enter the full path to the $OWB_HOME (minus the owb directory)
for instance /ozg/app/oracle/product/11.2.0.4/db_1
note: in some instances when running @start_service.sql later on to (re)start the CC service this may result in OWB not being able to find the rtrepos.properties file. In that case please run reset_owbcc_home.sql again but now add the owb directory to the path (i.e. /ozg/app/oracle/product/11.2.0.4/db_1/owb). After this start_service will be able to start normally. When on a rac node, make sure to copy rtrepos.propereties to all other nodes.

The post-clone steps:

1) . ozg_init.env <sid of the OHI Data Marts database>

2) . ozg_init.env DB11G2

On Exadata or in absence of ozg_init.env make sure ORACLE_HOME ENVIRONMENT_NAME en PATH environmental variables are set.

3) Login as user OWBSYS and perform the following commands to reset the repository password and stopping the control center

```
@$OWB_HOME/rtp/sql/reset_repository.sql <password OWBSYS user)
@$OWB_HOME/rtp/sql/stop_service.sql
```

4) Install the sources for the package WB_RT_RESET_LOCATION

```
@$OWB_HOME/rtasst/wb_rt_reset_location.pls
@$OWB_HOME/rtasst/wb_rt_reset_location.plb
```

5) Set the context for the OWB repository

```
begin
  wb_workspace_management.set_workspace( p_ws_name => 'OHI_BI_WS'
                                       , p_owner   => 'OHI_BI_WS_OWN'
                                       );
end;
/
```

6) Update the connection details for the Default Control Center. Change the connection details HOST, PORT and SERVICE_NAME to the appropriate values for the OHI Data Marts database

```
update wb_rt_service_nodes
set connect_spec = '<HOST>:<PORT>:<SERVICE_NAME>'
;
commit;
```

9) Update the location details for OHI Data Marts. Change the connection details HOST, PORT and SERVICE_NAME to the appropriate values for the OHI Data Marts database. The command below needs to be peformed for the following

locations (by changing the value for the parameter LOCATION within command listed below) LOC_DWH, LOC_STG, LOC_WBX and LOC_SRC_EXT_TABLES.

```
begin
  wb_rt_reset_location.fixRTLocation( '<LOCATION>'
                                    , '<HOST>'
                                    , '<PORT>'
                                    , '<SERVICE_NAME>'
                                    , '12'
                                    , 'OBD_OWN'
                                    , 'OBD_OWN'
                                    , '<Password of OBD_OWN>'
                                    );
end;
/
```

7) Update the location details for OHI Back Office. Change the connection details HOST, PORT and SERVICE_NAME to the appropriate values for the OHI Back Office database.

```
begin
  wb_rt_reset_location.fixRTLocation( 'LOC_SRC_OPENZORG'
                                    , '<HOST>'
                                    , '<PORT>'
                                    , '<SERVICE_NAME>'
                                    , '12'
                                    , 'OBD_SELECT_USER'
                                    , 'OBD_SELECT_USER'
                                    , '<Password of OBD_SELECT_USER>'
                                    );
end;
/
```

8) Start the Contol Center service. This may fail the first time.

```
@start_service.sql

Role set.
Available
Database managed service using nloz08:1527:oacc and home
/ozg/app/oracle/product/11.2.0.3/db_1
PL/SQL procedure successfully completed.
```

Please execute again after a minute and the service will start. Make absolutely sure that the service is DATABASE MANAGED (as opposed to CLIENT MANAGED). If this is not the case straight away, stop and start the service again.

9) Check the state of the Control Center

```
@service_doctor.sql

Role set.
>>>>>> There are errors in one or more PL/SQL packages and functions
Platform properties have been loaded correctly
Platform location has been seeded correctly
NLS messages have been loaded correctly
The platform service is available
Service script is accessible to the database server
Connection information stored within the repository is correct
PL/SQL procedure successfully completed.
```

10) Correct the OBD_INPUT directory object in the database (login SYS as SYSDBA).

```
SQL> drop directory obd_input;
Directory dropped.
SQL> create directory obd_input as '/ozg/app/oracle/product/Zorg/oacc';
Directory created.
SQL> grant read, write on directory obd_input to public;
Grant succeeded.
```

11) Create or change the private database link SRC_OPENZORG (as OBD_OWN) to the correct Back Office source. The account OBD_SELECT_USER should be used as the CONNECT TO user

```
create database link SRC_OPENZORG
connect to OBD_SELECT_USER
using '<OHI Back Office Service Name>';
```

APPENDIX E: MIGRATING OWB FROM SINGLE INSTANCE TO MULTI NODE RAC

Steps for OWB for RAC Environment

1. Create Unique Service Names

   srvctl add service -d rovac -s rovac1 -r rovac1
   srvctl add service -d rovac -s rovac2 -r rovac2

2. Start the service

   srvctl start service -d rovac -s rovac1
   srvctl start service -d rovac -s rovac2

3. Check the Status of added service's

   srvctl status service -d rovac

   Service rovac1 is running on instance(s) rovac1
   Service rovac2 is running on instance(s) rovac2

4. Complete post Clone steps as mentioned in the previous chapter. Please check the instructions mentioned below.

   - Use SCAN Address as the Host name for the Locations (SLC03FNR-S-SCAN.US.ORACLE.COM)
   - While running @UpdateControlCenter.sql, make sure to use the following parameters.

Enter Workspace Name:	OHI_BI_WS
Enter Workspace User Name:	OHI_BI_WS_OWN
Enter Control Center Name:	DEFAULT_CONTROL_CENTER
Host:	SLC03FNR.US.ORACLE.COM **(Make sure to use host name)**
Port:	1521
Service Name:	ROVAC1 **(Make sure you use ROVAC1 and not ROVAC for Unique Service Name for Each Node)**
New Net Service Name:	

   - When running update control center use ROVAC1:

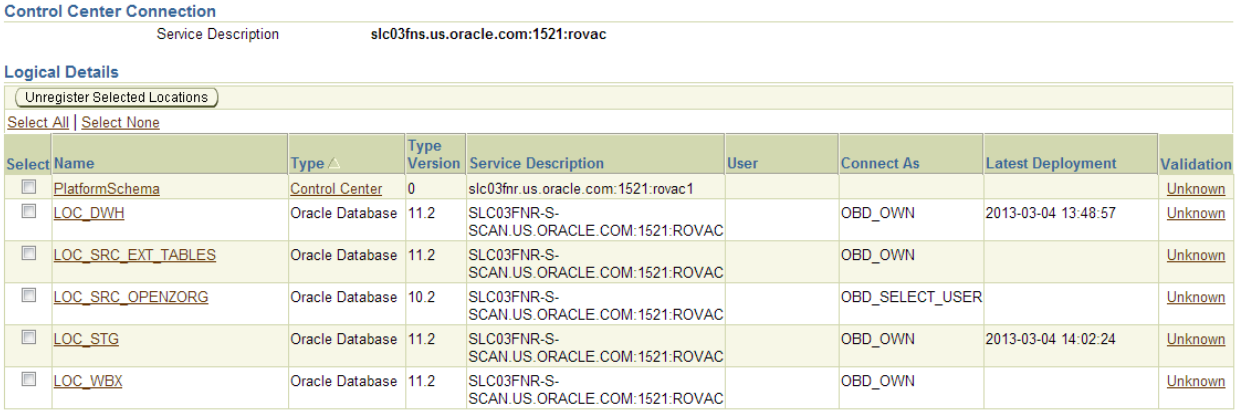
   ```
   update wb_rt_service_nodes
   set connect_spec = 'slc03fnr.oracle.com:1521:rovac1';

   commit;
   ```

   - Since service nodes need to be unique the service name for the node will be different than the service of the location, therefore the following error may be ignored.

     Error: Registered connection properties (SLC03FNR.ORACLE.COM:1521:ROVAC) are NOT the same as the connection properties for the Control Center (slc03fnr.us.oracle.com:1521:rovac1)

   - Location Report Should look as following.



Control Center Connection

Service Description: **slc03fns.us.oracle.com:1521:rovac**

Logical Details

Unregister Selected Locations

Select All | Select None

Select	Name	Type	Type Version	Service Description	User	Connect As	Latest Deployment	Validation
☐	PlatformSchema	Control Center	0	slc03fnr.us.oracle.com:1521:rovac1				Unknown
☐	LOC_DWH	Oracle Database	11.2	SLC03FNR-S-SCAN.US.ORACLE.COM:1521:ROVAC		OBD_OWN	2013-03-04 13:48:57	Unknown
☐	LOC_SRC_EXT_TABLES	Oracle Database	11.2	SLC03FNR-S-SCAN.US.ORACLE.COM:1521:ROVAC		OBD_OWN		Unknown
☐	LOC_SRC_OPENZORG	Oracle Database	10.2	SLC03FNR-S-SCAN.US.ORACLE.COM:1521:ROVAC		OBD_SELECT_USER		Unknown
☐	LOC_STG	Oracle Database	11.2	SLC03FNR-S-SCAN.US.ORACLE.COM:1521:ROVAC		OBD_OWN	2013-03-04 14:02:24	Unknown
☐	LOC_WBX	Oracle Database	11.2	SLC03FNR-S-SCAN.US.ORACLE.COM:1521:ROVAC		OBD_OWN		Unknown

5. How to Register the Node's for RAC

   No need of registering the first node. It will automatically get registered as 1st Node.

   We only need to register the 2nd Node.

   scp /u01/app/oracle/product/11.2.0/dbhome_1/owb/bin/admin/rtrepos.properties to 2nd Node ---> Before Registering this file should be copied to 2nd Node.

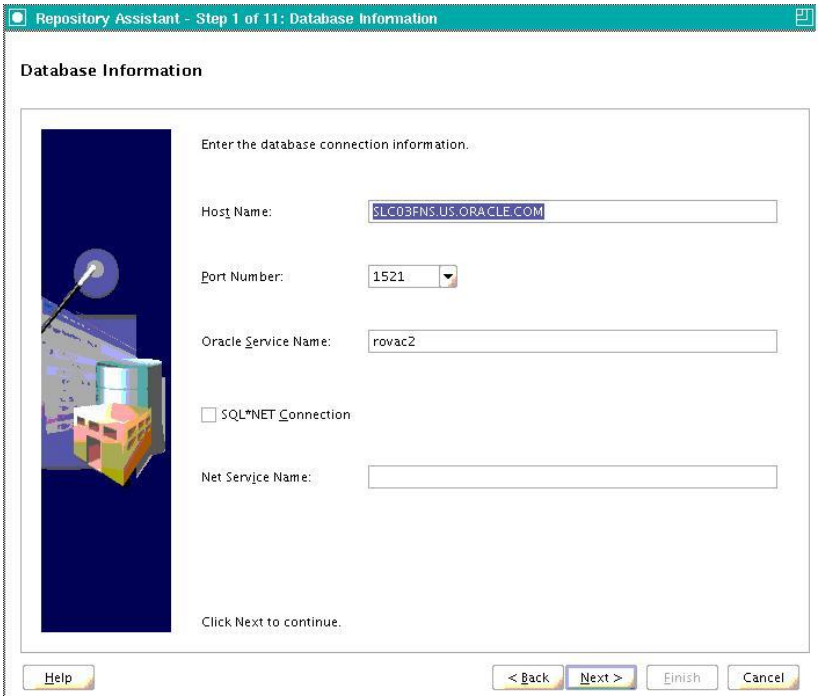
   Register Node 2 for RAC Services.
   Login to Node 2 and perform the following steps.

   . ozg_init.env rovac2
   . ozg_init.env DB11G2
   set the Appropriate DISPLAY according to the VNC Session (eg: export DISPLAY=:10.0) if vnc running on port 10

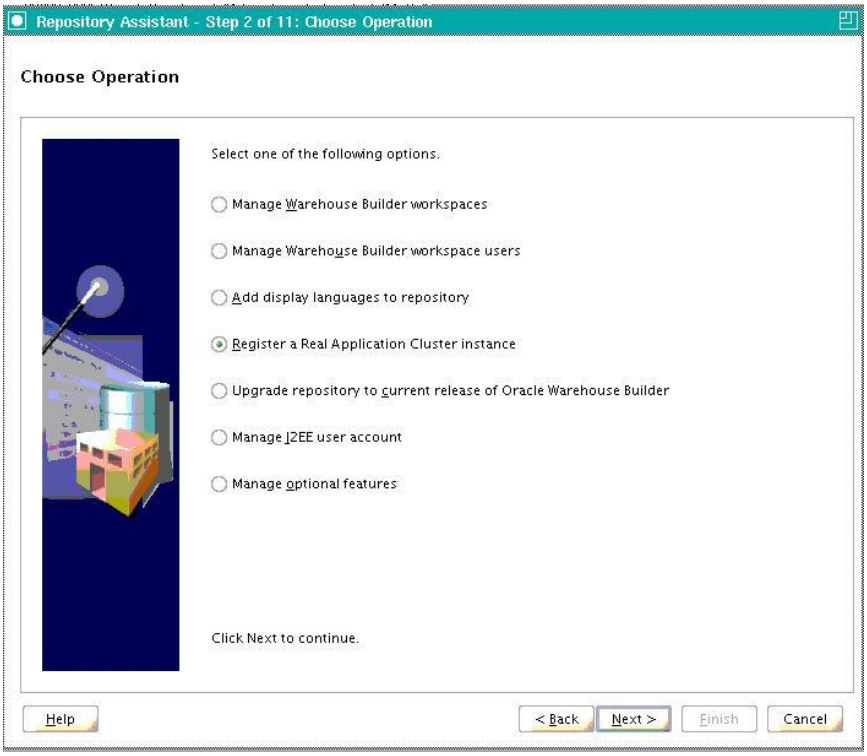
   On Exadata above steps would be:
   export ORACLE_HOME=/u01/app/oracle/product/11.2.0.3/dbhome_1
   export ENVIRONMENT_NAME=<SID>
   export PATH=$ORACLE_HOME/bin:$PATH
   export DISPLAY=:<displaynumber>

   cd $ORACLE_HOME/owb/bin/unix
   ./reposinst.sh

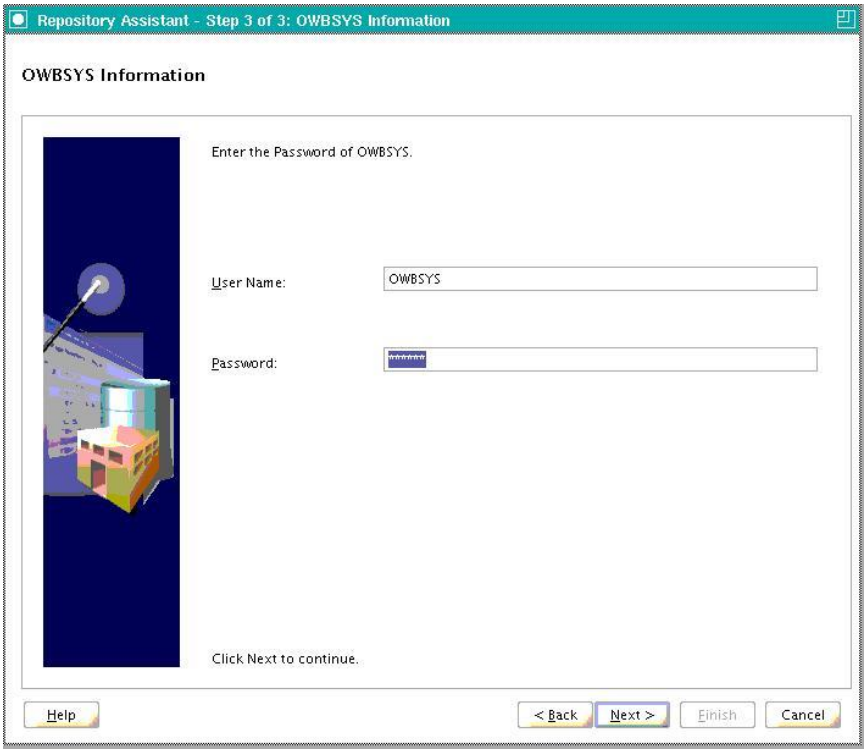
   Enter the Hostname as Node2 Host name (SLC03FNS.US.ORACLE.COM)
   Port 1521
   Oracle Service Name = rovac2 (2nd Node Unique Service Name)

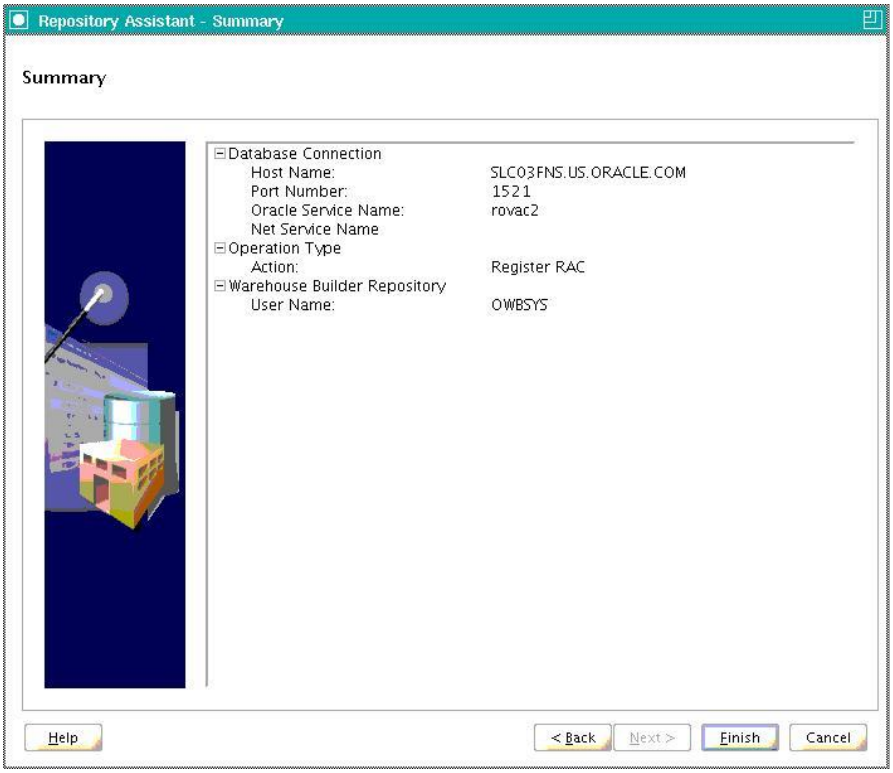


Select Register a Real Application Cluster Instance Option and click on Next



Enter the owbsys password and click on Next





Click on Finish.

A Message will appear that it has successfully registered the Node.

6. Finally the obd_own user needs permssion on the WB$_IV_CONTROL_CENTERS view for OBDPATCH.pl to succeed.

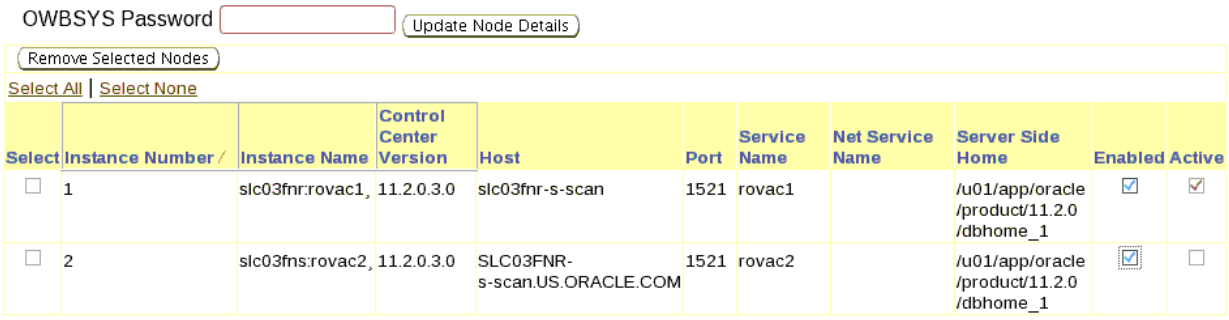
SQL> grant select on WB$_IV_CONTROL_CENTERS to OBD_OWN with grant option;

To Verify the Node is Registered do the following.
Start the Repository Browser Listener:

./startOwbbInst.sh

Login to Repository Browser Listener
https://slc03fnr.us.oracle.com:8999/owbb/RABLogin.uix

Click on Service Node Report.. Following should be visible.



Select	Instance Number	Instance Name	Control Center Version	Host	Port	Service Name	Net Service Name	Server Side Home	Enabled	Active
☐	1	slc03fnr:rovac1	11.2.0.3.0	slc03fnr-s-scan	1521	rovac1		/u01/app/oracle/product/11.2.0/dbhome_1	☑	☑
☐	2	slc03fns:rovac2	11.2.0.3.0	SLC03FNR-s-scan.US.ORACLE.COM	1521	rovac2		/u01/app/oracle/product/11.2.0/dbhome_1	☑	☐

The first time you login the 2nd node will not be enabled. Whether to enable node 2 depends on the setup you choose. Please refer to the next paragraph for the details; the default situation is 2 nodes enabled.

This screen shows that Unique Service names are used for each instance, ROVAC1 and ROVAC2.

How To Verify if OWB is Installed Correctly on RAC and Exadata (Doc ID 455999.1)

1. UNIQUE Service Names

   srvctl status service -d rovac

   Service rovac1 is running on instance(s) rovac1
   Service rovac2 is running on instance(s) rovac2

2. LISTENER Configuration
   We are using SCAN listener, so tnsnames.ora file has been updated as follows.

```
ROVAC1 =
  (DESCRIPTION =
    (ADDRESS LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = slc03fnr-vip.us.oracle.com)(PORT = 1521))
    )
    (CONNECT_DATA =
      (SERVICE_NAME = rovac1)
    )
  )

ROVAC2 =
  (DESCRIPTION =
    (ADDRESS LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = slc03fns-vip.us.oracle.com)(PORT = 1521))
    )
    (CONNECT_DATA =
      (SERVICE_NAME = rovac2)
    )
  )
```

3. To get the instance and status per node, execute the following command on one of the nodes:

```
select inst_id
,      instance_number
,      instance_name
,      database_status
```

```
,       status
,       host_name
from    gv$instance
;

   INST_ID INSTANCE_NUMBER INSTANCE_NAME    DATABASE_STATUS   STATUS       HOST_NAME
---------- --------------- ---------------- ----------------- ------------ -------------
         1 1               rovac1           ACTIVE            OPEN         slc03fnr
         2 2               rovac2           ACTIVE            OPEN         slc03fns
```

4. To get the service_name for all nodes, execute the following command on one of the nodes:

```
select inst_id
,       name
from    gv$services
;

INST_ID    NAME
---------- ----------------------------------------------------------------
         1 rovac1
         2 rovac2
```

5. If the service_names have been assigned to the instances, use following query to verify this.

```
select s.inst_id
,       instance_number
,       instance_name
,       name service_name
,       host_name
from    gv$services s
,       gv$instance i
where   s.inst_id = i.inst_id
;

 INST_ID   INSTANCE_NUMBER INSTANCE_NAME    SERVICE_NAME    HOST_NAME
---------- --------------- ---------------- --------------- ---------------
         1 1               rovac1           rovac1          slc03fnr
         2 2               rovac2           rovac2          slc03fns
```

6. How the OWB configuration should look like

   For OWB 11.2 queries are as follows:

```
select * from owbsys.owbrtps;

KEY VALUE
---------------------------------- ----------------------------------------
11.2.0.3.0[2] /u01/app/oracle/product/11.2.0/dbhome 1
11.2.0.3.0[1] /u01/app/oracle/product/11.2.0/dbhome_1
```

7. Check Service Node Report under OWB Browser Listener
   https://slc03fnr.us.oracle.com:8999/owbb/RABLogin.uix

   1 slc03fnr:rovac1 11.2.0.3.0 SLC03FNR.US.ORACLE.COM 1521 ROVAC1
   /u01/app/oracle/product/11.2.0/dbhome_1
   2 slc03fns:rovac2 11.2.0.3.0 SLC03FNS.US.ORACLE.COM 1521 ROVAC2
   /u01/app/oracle/product/11.2.0/dbhome_1

8. Enable all RAC nodes

```
select schema_user, what,
to_char(last_date, 'DD-MON-YYYY HH24:MI:SS') last_date,
to_char(next_date, 'DD-MON-YYYY HH24:MI:SS') next_date
from user_jobs
where what like '%check_service%'


SCHEMA_USER WHAT                                 LAST_DATE           NEXT_DATE
----------- ------------------------------------ ------------------- -------------------
OWBSYS      wb rti service job.check service(1);  09-OKT-2012 09:27:54 09-OKT-2012 09:33:54
OWBSYS      wb rti service job.check service(26); 09-OKT-2012 09:27:54 09-OKT-2012 09:33:54
```

```
begin
  wb_rti_service_job.check_service(1);
  wb_rti_service_job.check_service(26);
end;
/
```

9. Locations

   Locations that point to the RAC should be using Net Service Names. If host:port:service Locations are used, this would cause mappings to connect to a specific RAC node during mapping execution. A crash of such a node would stop the mapping. This can be prevented by using Net Service Names Locations.

Reference MOS Note ID's used to configure.

1. How to Install a Warehouse Builder Repository on a RAC (Doc ID 459961.1)
2. How To Verify if OWB is Installed Correctly on RAC and Exadata (Doc ID 455999.1)
3. How to Regenerate the rtrepos.properties File (Doc ID 473151.1)
4. Option To Register Node is Missing when Configuring OWB 11g On RAC (Doc ID 791362.1)

OWB RAC RATIONALE

With regard to RAC there are two configuration modes or rather setups possible. In Setup 1 there is one RAC node exclusively for ETL, and one exclusively for “Query”. Setup 2 has two nodes, both for query and for ETL.
In setup 1 ETL was performed on one node, and node 2 was queried constantly by two “Query” sessions over a database link. The databaselink connected to node 2 only.

In setup 2 ETL was performed on both nodes and both nodes were queried constantly by two “Query” sessions over a database link. The database link connected to the service.

Multiple tests were performed to define optimum: Setup 1 or Setup 2. AWR snapshots were taken during the tests. Below is a snippet from the AWR report from Setup 1

Global Cache Load Profile

	Per Second	Per Transaction
Global Cache blocks received:	1.14	17.69
Global Cache blocks served:	1.58	24.40
GCS/GES messages received:	130.18	2,013.60
GCS/GES messages sent:	72.16	1,116.16
DBWR Fusion writes:	0.05	0.72
Estd Interconnect traffic (KB)	83.06	

Below is a snippet from the AWR report from Setup 2

Global Cache Load Profile

	Per Second	Per Transaction
Global Cache blocks received:	0.39	4.87
Global Cache blocks served:	1.07	13.45
GCS/GES messages received:	75.68	950.75
GCS/GES messages sent:	42.95	539.55
DBWR Fusion writes:	0.03	0.39
Estd Interconnect traffic (KB)	46.50	

From this can be deducted that there is far more communication between the nodes in Setup 1 than in Setup 2. Alhough this might look strange since Partition Exchange Loading is used (PEL), this is actually not strange since PEL only is used in the fact tables and not in dimension tables.

Extraction Transformation and Loading Times:
Setup 1 approx: 5hrs 20 min
Setup 2 approx: 4hrs 53 min

From the tests is also concluded that Setup 2 performed ETL quicker than Setup 1, although the load times did not differ too much. Therefore preferred setup is setup 2, ETL and Query on both nodes.

OWB RAC SETUP 1

This setup describes pinning the ETL to node 1, reason for this is that other queries can be directed to node 2.

This Oracle Warehouse Builder setup for OHI Data Marts consists of the following;
OHI Back Office - on multi node rac node 1 and node 2
OHI Data Marts - on multi node rac, with OWB configured to run ETL on node1, node 2 is for custom ETL and query (EUL).

Setup 1 relies on an application server with wallet and tnsnames entry rovac1; sqlplus connects to the first node. In the Runtime Audit Browser only one node is enabled for ETL.

Service Nodes

OWBSYS Password [ ] Update Node Details

Remove Selected Nodes

Select All | Select None

Select	Instance Number	Instance Name	Control Center Version	Host	Port	Service Name	Net Service Name	Server Side Home	Enabled	Active
☐	1	slc03fnr:rovac1	11.2.0.3.0	slc03fnr-s-scan	1521	rovac1		/u01/app/oracle/product/11.2.0/dbhome_1	☑	☑
☐	2	slc03fns:rovac2	11.2.0.3.0	SLC03FNR-s-scan.US.ORACLE.COM	1521	rovac2		/u01/app/oracle/product/11.2.0/dbhome_1	☐	☐

Queries are bound to the instance by an after logon trigger or database parameter:
parallel_force_local = true

```
CREATE OR REPLACE TRIGGER trg_parallel_force_local AFTER logon ON database
  DECLARE
    l_osuser             VARCHAR2(1000);
    l_application_server VARCHAR2(1000);
  BEGIN
    SELECT lower(sys_context('USERENV', 'OS_USER')) l_osuser,
      lower(sys_context('USERENV', 'HOST')) l_application_server
    INTO l_osuser,
      l_application_server
    FROM DUAL;
    IF USER = 'BATCH' AND l_osuser = 'batch'
       AND l_application_server = 'slc03fnr'
    THEN --- change l_application_server to the name of your application server
      execute immediate 'ALTER session SET parallel_force_local = true';
    END IF;
  END;
  /
```

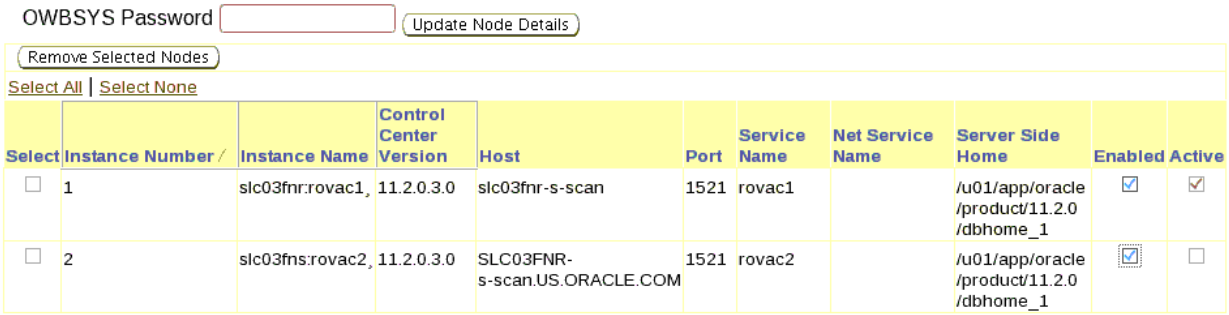
The wallet entries on the application server are directed to the first instance (node 1), and “Scriptaanvragen” are supplied with the ENVIRONMENT_NAME of the first node.

OWB RAC SETUP 2

This setup describes ETL that can run on both nodes, this is the preferred setup.

OHI Back Office - on multi node rac
OHI Data Marts - on multi node rac with OWB to configured ETL to run on both nodes, and custom ETL / query (EUL) on two nodes.

In the Runtime Audit Browser two nodes are enabled for ETL.



Select	Instance Number	Instance Name	Control Center Version	Host	Port	Service Name	Net Service Name	Server Side Home	Enabled	Active
☐	1	slc03fnr:rovac1	11.2.0.3.0	slc03fnr-s-scan	1521	rovac1		/u01/app/oracle/product/11.2.0/dbhome_1	☑	☑
☐	2	slc03fns:rovac2	11.2.0.3.0	SLC03FNR-s-scan.US.ORACLE.COM	1521	rovac2		/u01/app/oracle/product/11.2.0/dbhome_1	☑	☐

Queries are not bound to the instance by either a logon trigger or database parameter:
parallel_force_local = false

```
CREATE OR REPLACE TRIGGER trg_parallel_force_local AFTER logon ON database
  DECLARE
    l_osuser             VARCHAR2(1000);
    l_application_server VARCHAR2(1000);
  BEGIN
    SELECT lower(sys_context('USERENV', 'OS_USER')) l_osuser,
      lower(sys_context('USERENV', 'HOST')) l_application_server
    INTO l_osuser,
      l_application_server
    FROM DUAL;
    IF USER = 'BATCH' AND l_osuser = 'batch' AND l_application_server = 'slc03fnr'
    THEN --- change l_application_server to the name of your application server
      execute immediate 'ALTER session SET parallel_force_local = false;
    END IF;
  END;
  /
```

The wallet entries on the application server are directed to the service, not to a specific node.

RESTRICTED MODE

While patching an OHIBI version, it is recommended to have the database in restricted mode. For details about restriced mode see the Oracle Database Administrator Guide, for example part E25494-03, Altering Database Availability, Restricting Access to an Open Database.
To restrict access to an instance use the following command:

SQL> alter system enable restricted session;

Please note that if on a RAC cluster all the instances should be in restricted modes.
If however the database uses a dynamic listener, then the OHIBI database cannot be in restricted mode.
In such a case it is the responsiblity of the DBA to make sure there are no users logged in at the time of patching a release. To make sure there are no users logged in at the moment of patching and during the patch process make sure there are no active sessions from existing users by quering gv$session. Also make sure no user accounts can login to the database, by either informing your users, or locking accounts.
As an example two examples SQL are provided here:

SQL> select sid,serial#,username,osuser,machine from gv$session where osuser != 'oracle';
SQL> alter user <username> account lock;

To allow access to the instance use the following command;

SQL> alter system disable restricted session;

CHAINED ROWS

When installing or patching a release, chained rows will be detected if present in user_tables. To remove the chained rows do the following in SQLPlus.

```
SQL> alter table <TABLE_NAME> move;
SQL> alter table <TABLE_NAME> compute statistics;
```

After which the indexes are unusable and need to be rebuild with step 870 of the OHIPATCH.pl script.

APPENDIX F: ACTIVE DATA GUARD

OHI Data Marts supports an environment within which Active Data Guard is being used. OHI Data Marts has been tested and certified against a physical standby database in read-only mode.

The process of creating a Physical Standby database is described in detail in chapter “3 Creating a Physical Standby Database” of the document “Oracle® Data Guard Concepts and Administration, 11g Release 2 (11.2)”, part number E41134-02.

More information on managing Archive Redo Logs can be read in chapter 13 “Managing Archived Redo Logs” of the document “Oracle® Database Administrator's Guide, 11g Release 2 (11.2)”, part number E25494-04.

APPENDIX G: ORACLE ACCESS MANAGER / ORACLE INDENTITY MANAGER

Oracle Business Intelligence Enterprise Edition (OBIEE) is supported in combination with Oracle Access Manager and Oracle Identity Manager. For the supported version for each of the products please review the certification matrix available on the page 'Oracle Fusion Middleware Supported System Configurations' (http://www.oracle.com/technetwork/middleware/ias/downloads/fusion-certification-100350.html).

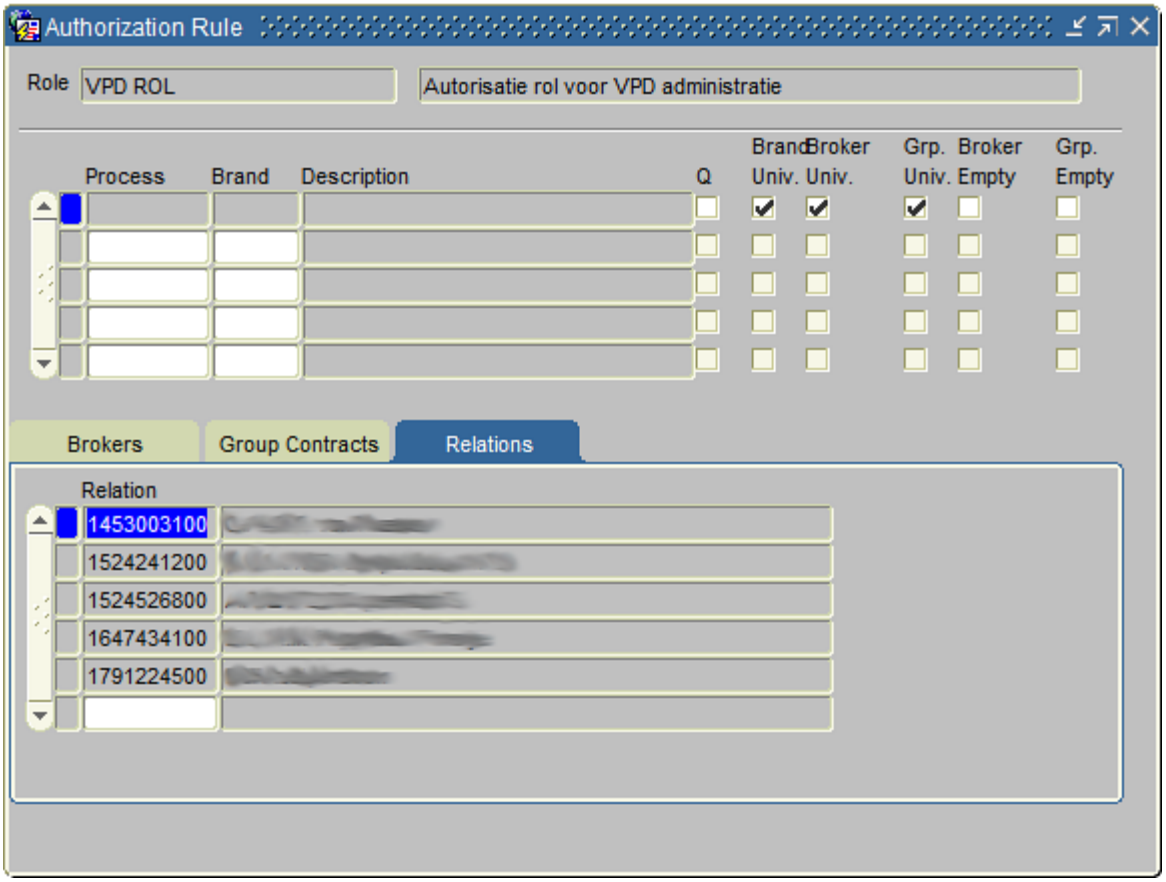
To setup OBIEE in combination with Oracle Access Manager 11g please review the Oracle Student Learning Installation and Deployment Guide, part number E20664-04. This guide provides detailed information on the installation and configuration of Oracle Access Manager 11g, as well as information on how to setup OAM as the SSO solution for OBIEE.

For detailed information on the Oracle Identity Managent Suite, please review the document Oracle Fusion Middleware Integration Overview for Oracle Identity Management Suite 11g Release 1 (11.1.1), part number E15477-03.

APPENDIX H: VIRTUAL PRIVATE DATABASE

Introduction

To make use of the feature Virtual Private Database (VPD) for masking privacy sensitive information one can specify Authorization Rules within OHI Back Office using the 'Authorization Rule'-form (ZRG7093F). These roles can be assigned to users/groups.



These rules are used within both OHI Back Office and OHI Data Marts to mask privacy sensitive information.

When logging into OBIEE the user-account used for OBIEE will be used to determine the authorization as is defined in OHI Back Office for the same user.

OBIEE Users

Within Weblogic Console users are to be defined with the exact same user name as they have been defined within OHI Back Office. These user need to belong at minimum to the group 'BIConsumers'.
See '[Introduction to Security in Oracle Business Intelligence]' for more information the Oracle Business Intelligence security model.

When a user logs in to the OBIEE environment a procedure is called to identify the functional user (i.e. the username used to log into OBIEE). This user will then be used to evaluate the authorization rules against.

When there is a need to also apply VPD outside of the OBIEE (i.e. for users directly connecting to the database themselves using a database account) a LOGON trigger should be defined to ensure that the VPD

policies also know for what user the authorization rules should be evaluated. The following is an example for a LOGON trigger.

```
create or replace trigger dwh_vpd_init_user
after logon
on database
enable
declare
  l_vpd_user stg_gebruiker_rollen.fun_oracle_user%type;
begin
  l_vpd_user := dwh_vpd_pck.set_user( USER );
end;
/
```

Make sure the user has been granted the OBD_ROL_SELECT and has not been granted '`exempt access policy`'.

Refreshing Authorizationrules

To be able to make use of VPD within the OHI Data Marts environment the authorization rules batch `ZRGO023S` (Refreshing Authorizationrules) needs to be executed. This batch load information about the authorization rules defined within OHI Back Office into the OHI Data Marts environment.
This batch can also to be scheduled using the menu options 'Data Marts' ➔ 'Refreshing Authorizationrules'.

VPD Policies

For the following OHI Data Marts tables/columns VPD policies are defined

Data Marts table	Privacy sensitive columns
DWH_ADRESSEN	ADS_AANVULLENDE_ADRESREGEL_1, ADS_AANVULLENDE_ADRESREGEL_2, ADS_AANVULLENDE_ADRESREGEL_3, ADS_AANVULLENDE_ADRESREGEL_4, ADS_AANVULLENDE_ADRESREGEL_5, ADS_AANVULLENDE_ADRESREGEL_6, ADS_AANVULLENDE_ADRESREGEL_7, ADS_AANVULLENDE_ADRESREGEL_8, ADS_AANVULLENDE_ADRESREGEL_9, ADS_HUISNUMMER, ADS_HUIS_NR, ADS_HUIS_NR_TOEVOEGING, ADS_LAND_CODE, ADS_LAND_OMSCHRIJVING, ADS_POSTCODE, ADS_POSTCODE_LETTER, ADS_POSTCODE_NR, ADS_STRAATNAAM, ADS_WOONPLAATS
DWH_COLLECTIEVE_CONTRACTEN	CCT_AANVULLENDE_ADRESREGEL, CCT_AANVULLENDE_ADRESREGEL_2, CCT_AANVULLENDE_ADRESREGEL_3, CCT_AANVULLENDE_ADRESREGEL_4, CCT_AANVULLENDE_ADRESREGEL_5, CCT_AANVULLENDE_ADRESREGEL_6, CCT_AANVULLENDE_ADRESREGEL_7, CCT_AANVULLENDE_ADRESREGEL_8, CCT_AANVULLENDE_ADRESREGEL_9, CCT_HUISNUMMER, CCT_HUIS_NR, CCT_HUIS_NR_TOEVOEGING, CCT_LAND_CODE, CCT_LAND_OMSCHRIJVING, CCT_NAAM, CCT_OMSCHRIJVING, CCT_POSTCODE, CCT_POSTCODE_LETTER, CCT_POSTCODE_NR, CCT_STRAATNAAM, CCT_WOONPLAATS
DWH_CREDITEUREN	CDR_AANVULLENDE_ADRESREGEL, CDR_AANVULLENDE_ADRESREGEL_2, CDR_AANVULLENDE_ADRESREGEL_3, CDR_AANVULLENDE_ADRESREGEL_4, CDR_AANVULLENDE_ADRESREGEL_5, CDR_AANVULLENDE_ADRESREGEL_6, CDR_AANVULLENDE_ADRESREGEL_7, CDR_AANVULLENDE_ADRESREGEL_8, CDR_AANVULLENDE_ADRESREGEL_9, CDR_HUISNUMMER, CDR_HUIS_NR, CDR_HUIS_NR_TOEVOEGING, CDR_LAND_CODE, CDR_LAND_OMSCHRIJVING, CDR_NAAM, CDR_OMSCHRIJVING, CDR_POSTCODE, CDR_POSTCODE_LETTER, CDR_POSTCODE_NR, CDR_P_AANVULLENDE_ADRESREGEL, CDR_P_AANVULLENDE_ADRESREGEL_2, CDR_P_AANVULLENDE_ADRESREGEL_3, CDR_P_AANVULLENDE_ADRESREGEL_4, CDR_P_AANVULLENDE_ADRESREGEL_5, CDR_P_AANVULLENDE_ADRESREGEL_6, CDR_P_AANVULLENDE_ADRESREGEL_7, CDR_P_AANVULLENDE_ADRESREGEL_8, CDR_P_AANVULLENDE_ADRESREGEL_9, CDR_P_HUISNUMMER, CDR_P_HUIS_NR, CDR_P_HUIS_NR_TOEVOEGING, CDR_P_LAND_CODE, CDR_P_LAND_OMSCHRIJVING, CDR_P_POSTCODE, CDR_P_POSTCODE_LETTER, CDR_P_POSTCODE_NR, CDR_P_STRAATNAAM, CDR_P_WOONPLAATS, CDR_STRAATNAAM, CDR_WOONPLAATS

Data Marts table	Privacy sensitive columns
DWH_DEBITEUREN	DBR_AANVULLENDE_ADRESREGEL, DBR_AANVULLENDE_ADRESREGEL_2, DBR_AANVULLENDE_ADRESREGEL_3, DBR_AANVULLENDE_ADRESREGEL_4, DBR_AANVULLENDE_ADRESREGEL_5, DBR_AANVULLENDE_ADRESREGEL_6, DBR_AANVULLENDE_ADRESREGEL_7, DBR_AANVULLENDE_ADRESREGEL_8, DBR_AANVULLENDE_ADRESREGEL_9, DBR_HUISNUMMER, DBR_HUIS_NR, DBR_HUIS_NR_TOEVOEGING, DBR_LAND_CODE, DBR_LAND_OMSCHRIJVING, DBR_NAAM, DBR_OMSCHRIJVING, DBR_POSTCODE, DBR_POSTCODE_LETTER, DBR_POSTCODE_NR, DBR_P_AANVULLENDE_ADRESREGEL, DBR_P_AANVULLENDE_ADRESREGEL_2, DBR_P_AANVULLENDE_ADRESREGEL_3, DBR_P_AANVULLENDE_ADRESREGEL_4, DBR_P_AANVULLENDE_ADRESREGEL_5, DBR_P_AANVULLENDE_ADRESREGEL_6, DBR_P_AANVULLENDE_ADRESREGEL_7, DBR_P_AANVULLENDE_ADRESREGEL_8, DBR_P_AANVULLENDE_ADRESREGEL_9, DBR_P_HUISNUMMER, DBR_P_HUIS_NR, DBR_P_HUIS_NR_TOEVOEGING, DBR_P_LAND_CODE, DBR_P_LAND_OMSCHRIJVING, DBR_P_POSTCODE, DBR_P_POSTCODE_LETTER, DBR_P_POSTCODE_NR, DBR_P_STRAATNAAM, DBR_P_WOONPLAATS, DBR_STRAATNAAM, DBR_WOONPLAATS
DWH_DECLARANTEN	DCT_AANVULLENDE_ADRESREGEL, DCT_AANVULLENDE_ADRESREGEL_2, DCT_AANVULLENDE_ADRESREGEL_3, DCT_AANVULLENDE_ADRESREGEL_4, DCT_AANVULLENDE_ADRESREGEL_5, DCT_AANVULLENDE_ADRESREGEL_6, DCT_AANVULLENDE_ADRESREGEL_7, DCT_AANVULLENDE_ADRESREGEL_8, DCT_AANVULLENDE_ADRESREGEL_9, DCT_HUISNUMMER, DCT_HUIS_NR, DCT_HUIS_NR_TOEVOEGING, DCT_IND_UNIVERSEEL, DCT_LAND_CODE, DCT_LAND_OMSCHRIJVING, DCT_OMSCHRIJVING, DCT_POSTCODE, DCT_POSTCODE_LETTER, DCT_POSTCODE_NR, DCT_P_AANVULLENDE_ADRESREGEL, DCT_P_AANVULLENDE_ADRESREGEL_2, DCT_P_AANVULLENDE_ADRESREGEL_3, DCT_P_AANVULLENDE_ADRESREGEL_4, DCT_P_AANVULLENDE_ADRESREGEL_5, DCT_P_AANVULLENDE_ADRESREGEL_6, DCT_P_AANVULLENDE_ADRESREGEL_7, DCT_P_AANVULLENDE_ADRESREGEL_8, DCT_P_AANVULLENDE_ADRESREGEL_9, DCT_P_HUISNUMMER, DCT_P_HUIS_NR, DCT_P_HUIS_NR_TOEVOEGING, DCT_P_LAND_CODE, DCT_P_LAND_OMSCHRIJVING, DCT_P_POSTCODE, DCT_P_POSTCODE_LETTER, DCT_P_POSTCODE_NR, DCT_P_STRAATNAAM, DCT_P_WOONPLAATS, DCT_STRAATNAAM, DCT_WOONPLAATS
DWH_DECLARATIES	AW_BSN_GEINDICEERDE_CLIENT, AW_BSN_PARTNER, AW_BSN_VERZEKERDE, AW_RELATIECODERING, HUISNUMMER, LANDCODE_BRON, PATIENTNUMMER, POSTCODE_BRON, POSTCODE_LETTER_BRON, POSTCODE_NR_BRON, REKENINGGEGEVEN, RT_LANDCODE, WOONPLAATS_BRON
DWH_PREMIES	LAN_CODE_BRON, POSTCODE_BRON, POSTCODE_LETTER_BRON, POSTCODE_NR_BRON, WOONPLAATS_BRON
DWH_TUSSENPERSONEN	TUP_AANVULLENDE_ADRESREGEL,TUP_AANVULLENDE_ADRESREGEL_2,TUP_AANVULLENDE_ADRESREGEL_3,TUP_AANVULLENDE_ADRESREGEL_4,TUP_AANVULLENDE_ADRESREGEL_5,TUP_AANVULLENDE_ADRESREGEL_6,TUP_AANVULLENDE_ADRESREGEL_7,TUP_AANVULLENDE_ADRESREGEL_8,TUP_AANVULLENDE_ADRESREGEL_9,TUP_HUISNUMMER,TUP_HUIS_NR,TUP_HUIS_NR_TOEVOEGING,TUP_IND_ABZ_BERICHT,TUP_LAND_CODE,TUP_LAND_OMSCHRIJVING,TUP_OMSCHRIJVING,TUP_POSTCODE,TUP_POSTCODE_LETTER,TUP_POSTCODE_NR,TUP_STRAATNAAM,TUP_WOONPLAATS
DWH_VERZEKERDEN	EXCASSOREKENING_DECLARATIES, HOEDANIGHEID, HUISNUMMER, INCASSOREKENING_DECLARATIES, LAN_CODE_BRON, PCE_LETTER_BRON, POSTCODE_BRON, REKENINGGEGEVEN, WOONPLAATS_BRON, WTZ_GROEP_CODE
DWH_ZORG_CLIENTEN	ZCL_AANVULLENDE_ADRESREGEL, ZCL_AANVULLENDE_ADRESREGEL_2, ZCL_AANVULLENDE_ADRESREGEL_3, ZCL_AANVULLENDE_ADRESREGEL_4, ZCL_AANVULLENDE_ADRESREGEL_5, ZCL_AANVULLENDE_ADRESREGEL_6, ZCL_AANVULLENDE_ADRESREGEL_7, ZCL_AANVULLENDE_ADRESREGEL_8, ZCL_AANVULLENDE_ADRESREGEL_9, ZCL_ACHTERNAAM, ZCL_A_NUMMER, ZCL_EMAIL_ADRES, ZCL_HUISNUMMER, ZCL_HUIS_NR, ZCL_HUIS_NR_TOEVOEGING, ZCL_LAND_CODE, ZCL_LAND_OMSCHRIJVING, ZCL_NAAM_NAAMAANDUIDING, ZCL_NAT_CODE, ZCL_NAT_OMS, ZCL_PAR_ACHTERNAAM, ZCL_PAR_VOORVOEGSEL, ZCL_POSTCODE, ZCL_POSTCODE_LETTER, ZCL_POSTCODE_NR, ZCL_P_AANVULLENDE_ADRESREGEL, ZCL_P_AANVULLENDE_ADRESREGEL_2, ZCL_P_AANVULLENDE_ADRESREGEL_3, ZCL_P_AANVULLENDE_ADRESREGEL_4, ZCL_P_AANVULLENDE_ADRESREGEL_5, ZCL_P_AANVULLENDE_ADRESREGEL_6, ZCL_P_AANVULLENDE_ADRESREGEL_7, ZCL_P_AANVULLENDE_ADRESREGEL_8, ZCL_P_AANVULLENDE_ADRESREGEL_9, ZCL_P_HUISNUMMER, ZCL_P_HUIS_NR, ZCL_P_HUIS_NR_TOEVOEGING, ZCL_P_LAND_CODE, ZCL_P_LAND_OMSCHRIJVING, ZCL_P_POSTCODE, ZCL_P_POSTCODE_LETTER, ZCL_P_POSTCODE_NR, ZCL_P_STRAATNAAM, ZCL_P_WOONPLAATS, ZCL_SOFI_NR, ZCL_STRAATNAAM, ZCL_TELEFOON_NR, ZCL_TITULATUUR, ZCL_VOORLETTERS, ZCL_VOORNAMEN, ZCL_VOORVOEGSELS, ZCL_WOONPLAATS, ZCL_ZCT_DERDEN_CODERING_1, ZCL_ZCT_DERDEN_CODERING_2, ZCL_ZCT_DERDEN_CODERING_3, ZCL_ZCT_DERDEN_CODERING_4, ZCL_ZCT_DERDEN_CODERING_5

Data Marts table	Privacy sensitive columns
DWH_ZORG_RELATIES	KOE_CODE, KOE_OMSCHRIJVING, ZRE_AANVULLENDE_ADRESREGEL, ZRE_AANVULLENDE_ADRESREGEL_2, ZRE_AANVULLENDE_ADRESREGEL_3, ZRE_AANVULLENDE_ADRESREGEL_4, ZRE_AANVULLENDE_ADRESREGEL_5, ZRE_AANVULLENDE_ADRESREGEL_6, ZRE_AANVULLENDE_ADRESREGEL_7, ZRE_AANVULLENDE_ADRESREGEL_8, ZRE_AANVULLENDE_ADRESREGEL_9, ZRE_CTG_NR, ZRE_EXTERN_KENMERK, ZRE_HAGRO_OMSCHRIJVING, ZRE_HUISNUMMER, ZRE_HUIS_NR, ZRE_HUIS_NR_TOEVOEGING, ZRE_LAND_CODE, ZRE_LAND_OMSCHRIJVING, ZRE_POSTCODE, ZRE_POSTCODE_LETTER, ZRE_POSTCODE_NR, ZRE_P_AANVULLENDE_ADRESREGEL, ZRE_P_AANVULLENDE_ADRESREGEL_2, ZRE_P_AANVULLENDE_ADRESREGEL_3, ZRE_P_AANVULLENDE_ADRESREGEL_4, ZRE_P_AANVULLENDE_ADRESREGEL_5, ZRE_P_AANVULLENDE_ADRESREGEL_6, ZRE_P_AANVULLENDE_ADRESREGEL_7, ZRE_P_AANVULLENDE_ADRESREGEL_8, ZRE_P_AANVULLENDE_ADRESREGEL_9, ZRE_P_HUISNUMMER, ZRE_P_HUIS_NR, ZRE_P_HUIS_NR_TOEVOEGING, ZRE_P_LAND_CODE, ZRE_P_LAND_OMSCHRIJVING, ZRE_P_POSTCODE, ZRE_P_POSTCODE_LETTER, ZRE_P_POSTCODE_NR, ZRE_P_STRAATNAAM, ZRE_P_WOONPLAATS, ZRE_REL_OMSCHRIJVING, ZRE_STRAATNAAM, ZRE_WOONPLAATS, ZRE_ZCL_CODE, ZRE_ZCL_OMSCHRIJVING, ZRE_ZKR_CODE
DWH_ZORG_WERKGEVERS	WGR_AANVULLENDE_ADRESREGEL, WGR_AANVULLENDE_ADRESREGEL_2, WGR_AANVULLENDE_ADRESREGEL_3, WGR_AANVULLENDE_ADRESREGEL_4, WGR_AANVULLENDE_ADRESREGEL_5, WGR_AANVULLENDE_ADRESREGEL_6, WGR_AANVULLENDE_ADRESREGEL_7, WGR_AANVULLENDE_ADRESREGEL_8, WGR_AANVULLENDE_ADRESREGEL_9, WGR_HUISNUMMER, WGR_HUIS_NR, WGR_HUIS_NR_TOEVOEGING, WGR_LAND_CODE, WGR_LAND_OMSCHRIJVING, WGR_OMSCHRIJVING, WGR_POSTCODE, WGR_POSTCODE_LETTER, WGR_POSTCODE_NR, WGR_STRAATNAAM, WGR_TELEFOON_NR, WGR_WOONPLAATS

Using step 880 (Enable/disable VPD) of OHIPATCH.pl the VPD policies can be enabled (Policy group 'OHI_RELATIONS', policy 'REL_ANONIEM').

Adding custom policies

If there is a need to add additional VPD policies that are not offered within OHI Data Marts, one could create additional policies keeping in mind the following rules

- Do not add custom policies to the policy group 'OHI_RELATIONS' and/or policy name 'REL_ANONIEM'. These are used and reserved for OHI Data Marts
- Custom policies should not negatively impact performance
- Custom policies are only allowed to mask columns and should never remove entire rows (i.e. always specify what columns should be masked with the table or tables for which the custom policy is defined using the `sec_relevant_cols` option). Hiding/removing rows will result incomplete data when reporting on OHI Data Marts.
- Only SELECT policies are allowed
- Do not apply masking on columns that are most likely to be used within predicates (i.e. foreign key relation columns)

Example of adding a custom policy for DWH_MERKEN

In this example a custom policy is created for the dimension DWH_MERKEN within which data for the columns `AOR_OMSCHRIJVING`, `MRK_OMSCHRIJVING`, `AGB_OMSCHRIJVING` and `ZVR_NAAM` will be masked (for all users) when data related to `FINPR`, `PRE` or `ZFWCD` is selected.

Please consult the Oracle Database Security Guide for additional information on the topic 'Using Oracle Virtual Private Database to Control Data Access' at https://docs.oracle.com/database/121/DBSEG/vpd.htm#DBSEG007.

Creating new policy group (CUSTOM_VPD_GROUP)

```
begin
  dbms_rls.enable_grouped_policy
  ( object_name => 'DWH_MERKEN'
  , group_name  => 'CUSTOM_VPD_GROUP'
  , policy_name => 'CUSTOM_VPD_POLICY'
  , enable      => true
  );
end;
/
```

Creating new policy (CUSTOM_VPD_POLICY) within policy group

Thefollwing statement creates a SELECT policy for the table DWH_MERKEN for which data for the columns `AOR_OMSCHRIJVING`, `MRK_OMSCHRIJVING`, `AGB_OMSCHRIJVING` and `ZVR_NAAM` is to be masked.

The PL/SQL package/function DWH_CUSTOM_VPD_PCK.ADD_PREDICATE_SEL is specified as the function which generates the predicate for this policy.

```
begin
  dbms_rls.add_grouped_policy
  ( object_name           => 'DWH_MERKEN'
  , policy_group          => 'CUSTOM_VPD_GROUP'
  , policy_name           => 'CUSTOM_VPD_POLICY'
  , policy_function       => 'DWH_CUSTOM_VPD_PCK.ADD_PREDICATE_SEL'
  , statement_types       => 'SELECT'
  , enable                => false
  , sec_relevant_cols     =>
'AOR_OMSCHRIJVING,MRK_OMSCHRIJVING,AGB_OMSCHRIJVING,ZVR_NAAM'
  , sec_relevant_cols_opt => dbms_rls.ALL_ROWS
  );
end;
/
```

PL/SQL package

The PL/SQL package that is linked to the VPD policy will be executed for every row/column returned from a query that (in this example) uses the table DWH_MERKEN, by calling the function `add_predicate_sel`.

The function `add_predicate_sel` first checks for which table it is being called, this way this package can be used for more then just the DWH_MERKEN table, but could also be used for other tables containing information related to DWH_MERKEN. If this function returns condition which evaluates to **False** access is granted and the data will be masked.

The function `check_mrk` performs the actual check whether or not access is allowed.

```
create or replace package dwh_custom_vpd_pck
is

  function add_predicate_sel
  ( pi_schema_name in varchar2
  , pi_object_name in varchar2
  ) return varchar2
  ;

end dwh_custom_vpd_pck;
/

create or replace package body dwh_custom_vpd_pck
is
```

```
  function check_mrk
  ( pi_mrk_code in dwh_merken.mrk_code%type
  ) return varchar2
  is
    l_retval varchar2(1) := 'N'; -- By default no access
  begin
    if pi_mrk_code in ('FINPR', 'PRE', 'ZFWCD')
    then
      l_retval := 'J'; -- Allow access
    end if;
    return l_retval;
  end;

  function add_predicate_sel
  ( pi_schema_name in varchar2
  , pi_object_name in varchar2
  ) return varchar2
  is
    l_retval    varchar2(4000) := ' 1=1 ';
    l_tablename varchar2(60);
  begin
    l_tablename := upper(pi_object_name);
    if l_tablename = 'DWH_MERKEN'
    then
      l_retval := ' (dwh_custom_vpd_pck.check_mrk(MRK_CODE) = ''J'') ';
    end if;
    return l_retval;
  end add_predicate_sel;

end dwh_custom_vpd_pck;
/
```

Ensure that the role OBD_ROL_SELECT has been granted execute privileges on the package DWH_CUSTOM_VPD_PCK and a public synonym has been created.

```
grant execute on obd_own.dwh_custom_vpd_pck to OBD_ROL_SELECT;
create public synonym dwh_custom_vpd_pck for obd_own.dwh_custom_vpd_pck;
```

Enabling the policy

By default when adding a policy the policy is not enabled. Using the following statement the policy on DWH_MERKEN is activated.

```
begin
  dbms_rls.enable_grouped_policy
  ( object_name => 'DWH_MERKEN'
  , group_name  => 'CUSTOM_VPD_GROUP'
  , policy_name => 'CUSTOM_VPD_POLICY'
  , enable      => true
  );
end;
/
```

For this moment on any query on DWH_MERKEN each for within DWH_MERKEN will be evaluated and data will be masked accordingly.