ORACLE
INSURANCE

Oracle Health Insurance Back Office

Release Installation

version 2.74

Part number: G49637-01

March 25, 2026

CHANGE HISTORY

Release	Version	Changes
10.12.2.0.0	2.17	Added change history paragraph.
10.12.3.0.0	2.18	Added appendices C&D about database parameters and privileges checkups
10.13.1.0.0	2.19	Changed appendix C about database parameter checkups
10.13.2.0.0	2.20	Changed appendix C because checks on database instance parameters have been changed: • The checks on other non OHI Back Office prescribed parameters now result in a warning message when they are set instead of a fatal message. • The check on _OPTIMIZER_JOIN_FACTORIZATION being set on false is now only applicable to OHI Back Office. • Changed check on PARALLEL_MAX_SERVERS as parallel execution activation is now supported during runtime use of the OHI Back Office application. For that reason now also checks on PARALLEL_DEGREE_POLICY, PARALLEL_ADAPTIVE_MULTI_USER, PARALLEL_FORCE_LOCAL and PARALLEL_MIN_TIME_THRESHOLD are introduced. For more regarding these last settings and their usage please consult the Installation, Configuration and DBA manual. • A number of checks now results in a lower severity. This documentation has been adapted to comply with this lower severity.
10.13.2.0.0	2.21	• Checks on Back Office instance parameters PARALLEL_DEGREE_POLICY, PARALLEL_ADAPTIVE_MULTI_USER, PARALLEL_FORCE_LOCAL were not described in the previous version despite the comment mentioning it. There is no check on PARALLEL_MIN_TIME_THRESHOLD. • For OHI Data Marts the check on PARALLEL_THREADS_PER_CPU is specified. • The instructions for reading in a Designer export dump file have changed. • The completion instruction now explicitly reminds you of redeploying the .ear files used for the web services.
10.13.3.0.0	2.22	• Enforce grant security in step 120 of OZGPATCH • Configuration file parameter ozg_120_revoke_in_automode • Grant OZG_ROL_BATCH to OZG_OWNER for administrative purposes • Added message to generic Back Office messages in appendix C • Removed check on _OPTIMIZER_JOIN_FACTORIZATION, added check on OPTIMIZER_USE_SQL_PLAN_BASELINES in instance parameter checks.
10.13.3.0.0	2.23	• Replaced reference to DB11G2 by DB11204 as OHI Back Office release 10.13.3 uses database version 11.2.0.4. • Corrected import command for using datapump, impdp instead of impdb.
10.13.3.0.0	2.24	• Corrected text on check for workarea_size_policy.
10.14.1.0.0	2.25	• Changed DBMS_LOCK check, granted now to user instead of PUBLIC • Added checks for OHI Dynamic PL/SQL user account
10.14.1.0.0	2.26	• Added description on functionality to suppress errors during installation • Added description on optional external alert command • Updated references to releases with the new release naming layout • Added parameter for parallel forms / reports compilation to speed up installation
10.14.2.0.0	2.27	• Added parameter to force client compilation during automode
10.15.1.0.0	2.28	• Changed parameter check on OPTIMIZER_MODE • Removed parameter OPTIMIZER_INDEX_COST_ADJ • Removed references to Oracle Reports and reports account • $ORACLE_SID replaced $TWO_TASK • Removed ozdefine.sql (not used anymore) • Changed references from 11g Oracle Home to 12 Oracle Home • Implemented consequences of files moved from $OZG_ADMIN to $OZG_BASE/utils and rename of OZGPATCH to OHIPATCH • Added check on PGA_AGGREGATE_LIMIT, OPTIMIZER_ADAPTIVE_FEATURES, REMOTE_DEPENDENCIES_MODE • Changed check on OPTIMIZER_USE_SQL_PLAN_BASELINES from ERROR to INFO message. • Removed check on RESOURCE_MANAGER_PLAN.
10.15.3.0.0	2.29	• Added advise for installation menu step 800. • Check for _common_data_view_enabled added. • Changed description for checks on PCT_FREE. • Changed description for adjusting SORT_AREA_SIZE and HASH_AREA_SIZE • Moved check on WORKAREA_SIZE_POLICY to generic checks • Renamed OHI Business Intelligence to OHI Data Marts

Release	Version	Changes
10.16.1.0.0	2.30	• WORK_AREA_SIZE is now checked to be always set at AUTO • For PARALLEL_MAX_SERVERS during runtime use of OHI Back Office a warning is now actually given when it is higher than CPU_COUNT times PARALLEL_THREADS_PER_CPU • When a parameter is set at the PDB level a warning is given this is against OHI advise • During OHI BO installations it is checked a minimum value of 100Mb for HASH_AREA_SIZE and 200Mb for SORT_AREA_SIZE is set. • Some additional instructions are added for when you run step 120 with answer Yes for enforcing grant security.
10.16.2.0.0	2.31	• Added installation menu step 880. • Added privileges for OHI_VIEW_OWNER. • Changed check for PGA_AGGREGATE_LIMIT.
10.17.1.0.0	2.33	• Removed references to release folder /admin. • Added advise for patching patch ranges in automode due to postponing step 120 in this situation.
10.17.1.3.0	2.34	• Added system priv. check ALTER ANY SEQUENCE to Appendix E.
10.17.1.4.0	2.35	• Removed requirement to set DB initialization parameters only at CDB level from Appendix D.
10.17.2.0.0	2.36	• Grant security will be enforced by default; removed installation option ozg_120_revoke_in_automode • Added installation option ozg_120_verbose_grants_in_automode • Added SELECT ANY DICTIONARY as mandatory privilege for OHI Back Office dynamic PL/SQL user in appendix E. • Replaced installation menu step 140 by more generic step 890.
10.17.2.2.0	2.37	• Added chapter 9 about patchreleases that allow webservices to remain online. • Added “END :” tag message and evaluation of this tag during OHIPATCH “error scan” processing.
10.18.1.30.0	2.38	• Removed Removed ALTER ANY SEQUENCE system privilege. • Adapted version checks to check on database 12.2 instead of 12.1 component versions. • Check on OPTIMIZER_FEATURES_ENABLE adjusted to 12.2. • Removed check on OPTIMIZER_ADAPTIVE_FEATURES (is deprecated, should not be set). Added checks on OPTIMIZER_ADAPTIVE_PLANS and OPTIMIZER_ADAPTIVE_STATISTICS. • Changed check on _COMMON_DATA_VIEW_ENABLED (no longer used as work around). • Added check on AWR_PDB_AUTOFLUSH_ENABLED. • NLS_LENGTH_SEMANTICS should be set to BYTE again instead of CHAR. • Added description for installing Designer exports.of TC releases. • Added additional instructions for how to interpret step 890 • Added mandatory privilege ADMINISTER DATABASE TRIGGER and ANALYZE ANY for Data Marts
10.18.1.2.0	2.39	• Checks on PGA_AGGREGATE_TARGET and PGA_AGGREGATE_LIMIT are changed from FATAL to ERROR.
10.18.1.3.0	2.40	Added description of new checks: • Check if private grants on SYS and XDB objects have been issued to OHI BO owner, other OHI accounts and OHI_ROLE_ALL. • Check if OHI BO roles have been granted illegally to other OHI roles or OHI BO accounts. • Check if OHI BO users (functionarissen) and owners of customizations own objects with names used by OHI BO. • Check if obsolete BATCH user and roles have been removed. • Removed description for installing Designer exports.
10.18.1.4.0	2.41	• Changed installation menu step 850 regarding compression and partitioning functionality.
10.18.2.0.0	2.42	• Removed references to OHI Self Service where possible, changed incorrect references to database 12.1 related environment variables.
10.18.2.2.0	2.43	• Option 855 (drop partitions) for OHI Data Marts added. • Privilege on SYS.UTL_RECOMP added.
10.18.2.3.0	2.44	• Option 870 was not described despite being present for already a long time
10.19.1.0.0	2.45	• No changes. Republished with different part nr.
10.19.1.3.0	2.46	• Removed check on ALG_PERL_JOB.
10.19.1.4.0	2.47	• Added some additional information regarding applying partitioning as this missed some details which became clear when adding some more tables in this release where the partitioning is useful for.
10.19.2.0.0	2.48	• Update user requirements paragraph.
10.20.1.0.0	2.49	• No changes. Republished for year 2020.
10.20.3.0.0	2.50	• For PGA_AGGREGATE_LIMIT the checked value has been changed • Implemented effects of DB 19c being required • Other textual changes and improvements

Release	Version	Changes
10.20.4.0.0	2.51	• Added Initual Capacity setting for data source when web services are kept online during patching
10.20.5.0.0	2.52	• Adapted the last three Appendix chapters for the impact of offering the option for creating an OHI specific Database Vault realm.
10.20.6.0.0	2.53	• Appendix with Database Vault checks has been slightly adapted.
10.20.7.0.0	2.54	• Appendix with Database Vault checks has been slightly adapted. • Updated instructions for initial release installation
10.20.8.0.0	2.55	• Small correction in description of Appendix with Database Vault checks • INFO message for OPTIMIZER_USE_SQL_PLAN_BASELINES has been corrected
10.21.1.0.0	2.56	• Added check on _OPTIMIZER_USE_STATS_ON_CONVENTIONAL_DML • New part number
10.21.4.0.0	2.57	• Changed default time taken for step 820 to 0 hours. Added clarification for $OZG_BASE/conf/ohipatch.conf and defaults. • Added Chapter Command Line mode
10.21.6.0.0	2.58	• Compression options changed. Updated Command Line parameter values for step 850. Changed Appendix A
10.21.8.0.0	2.59	• OHIPATCH option 130 removed; EI media will be installed during database installation (step 110)
10.22.1.0.0	2.60	• No changes, republished with new part number.
10.22.2.0.0	2.61	• Extra copy for HSL sources in step 220 and 240 added.
10.22.7.0.0	2.62	• Added initial step to check for out-of-order installation.
10.23.1.0.0	2.63	• No changes, republished with a new part number.
10.23.8.0.0	2.64	• Updates for OBD_SELECT_USER; OBD_SELECT_ROLE removed.
10.24.1.0.0	2.65	• Republished with new part number. • Added check on _optimizer_vector_transformation • Updated APPENDIX C SUPPRESSING KNOWN INSTALLATION ERRORS FOR AUTOMATIC INSTALLATION for installing interim patches out of order
10.24.2.0.0	2.66	• Modified Wallet entry for application owner account
10.24.3.0.0	2.67	• It is now mandatory that the OHI BO object owner accounts are locked during regular operation.
10.24.5.0.0	2.68	• Rewrote Backup Requirements • Step 120: all grants and revoke will be logged; removed installation options ozg_120_verbose_grants_in_automode and rtr_pw
10.24.8.0.0	2.69	• Added SYSDATE_AT_DBTIMEZONE to appendix D
10.25.1.0.0	2.70	• For PGA_AGGREGATE_LIMIT the checked value has been changed • Republished with new part number.
10.25.2.0.0	2.71	• Compression and Partitioning: candidate tables specified
10.25.6.0.0	2.72	• Database option JAVAVM and related privileges are no longer needed. • Partitioning only an index is not supported Appendix_A_partition_compression
10.26.1.0.0	2.73	• Updated Database Vault check • New part number.
10.26.3.0.0	2.74	• Updated references to the document “OHI BO and DM - known installation issues.docx” and the announcements on Sharepoint. • Added description for version check after (re)deployment of a new .ear file. • Replaced ozg_init.env by oraset

CONTENTS

1. INTRODUCTION

This document contains the guidelines, for installation of a release for Oracle Health Insurance customers.

The document provides a technical description concerning the location of releases, how to install releases and how to handle errors during installation.

All the files of a (new) release are stored in a directory that has a name equal to the release name.

In this document the placeholder <release> is used for the name of a release. This placeholder has to be replaced by the release name. E.g. When you are installing a release with the release name '10.15.1.0.0' and the instructions contain a directory <release> then the directory with the name 10.15.1.0.0 is meant.

2. BASIC ASSUMPTIONS

The following basic assumptions apply to the installation of releases.

DELIVERY

OHI releases for OHI Back Office and OHI Data Marts are delivered via My Oracle Support (MOS) through section (tab) “Patches & Updates”.

NAMING OF THE RELEASE

TYPES OF RELEASES

There are 2 release types:

Release type	Description
Major release	Regular release with corrective and adaptive functionality. A major release will be delivered every 6 weeks.
Interim patch (release)	Bug fix for a certain problem. Does not contain adaptive functionality.

N.B.: Major releases are NOT ***cumulative***: Each major release requires the previous major release to be installed first. A major release does contain interim patches released since the previous major release, if they were ready in time to make it into the build.

Example

The OHI Back Office environment has version 10.20.1.0.0.
You want to upgrade OHI Back Office to release 10.20.3.0.3.
To do this, you have to install the following major release and interim patches:

1. Major release 10.20.2.0.0.
   This major release contains new functional changes and fixes, and 10.20.1.0.X interim patches as delivered up to the moment the major release was created.
2. Major release 10.20.3.0.0.
   This major release contains new functional changes and fixes, and 10.20.2.0.X interim patches as delivered up to the moment the major release was created.
3. Interim patch 10.20.3.0.1
4. Interim patch 10.20.3.0.2
5. Interim patch 10.20.3.0.3

ORACLE HEALTH INSURANCE RELEASE CALENDAR

The Oracle Health Insurance Release calendar contains the dates for releasing major releases and the end of support dates for major releases. The release calender is published on Beehive Online.

DEPENDENCY BETWEEN MAJOR AND INTERIM PATCHES

An interim patch can only be installed on the associated major release.

Example

Patch release **10.20.3.0**.1 can only be installed on release **10.20.3.0**.0.

RELEASE PER ENVIRONMENT

There is only one Oracle Health Insurance major release active per "environment" (including patch sets and interim patches concerned), e.g.:

- Production 10.20.1.0.X
- Acceptance 10.20.2.0.X

RELEASE NAMING

Release naming is done using the following coding conventions (named positions are divided by a *dot*):

Position	Description
1-2	Product suite version, i.e. '10'
4-5	Year in which the release was/is going to be released
7	Sequence number of the major release
9	Always zero
11-13	Interim patch release sequence number (0 means it does not concern an interim patch)

N.B.: The latest 4 characters of a release name may contain 'TC' for the first two characters followed by a 2 digit number. An example: 10.14.2.0.TC33. Such release names are reserved for delivering a 'Test release'. They are mainly used internally by OHI Development but may occasionally be delivered to customers, typically after a mutual agreement to use it for investigation and testing purposes.

It is very important to be aware of the nature of such a release and the consequences of installing such a Test release on an OHI environment, as the contents of such a release are volatile and not tracked by OHI Development. Installing such a release means the environment it is installed on can only used for testing purposes and will have to be destroyed or restored to a point before the installation of the Test release. at some later moment, as its state is unclear. Multiple versions of a TC release may have been created and there is *no way to get back to a supported configuration* other than by reverting to a copy of an environment without a TC release.

INSTALLATION SEQUENCE

The following basic assumptions apply to the installation sequence of patch releases:

MAJOR RELEASE DEPENDENCY

Installation of Oracle Health Insurance releases must be done in order of *dependency*. This means that the major release is installed first and subsequently the interim patches. It is not possible to install an interim patch when the associated major release has not yet been installed.

Example

Before *interim patch* **10.20.3.0.1** can be installed, *major* release **10.20.3.0.0** has to be installed.

SEQUENCE NUMBER DETERMINES INSTALLATION ORDER

Interim patches should be installed in the order of their sequence numbers.

Example 1 – major releases

10.20.**1**.0.0 must be installed before 10.20.**2**.0.0.

10.20.**2**.0.0 must be installed before 10.20.**3**.0.0.

Example 2 – interim patch releases

10.20.1.0.1 must be installed before 10.20.1.0.2.

10.20.1.0.2 must be installed before 10.20.1.0.3.

MANDATORY INSTALLATION

Interim patches

When an interim patch is installed, all previous interim patches have to be installed on the same major release. Exceptions are only allowed after a customer request has been approved by OHI Support. Without this explicit approval, you should NEVER skip patches or change the order. OHI Support needs to evaluate the dependencies between patches first..

Starting with release 10.22.7.0.1, the OHI patch program will check if the pre-requisite (patch-) release has been installed, before starting the installation.

3. INSTALLATION OF A RELEASE

REQUIREMENTS

During the installation of an Oracle Health Insurance major release no other sessions should be active, performing actions on the Oracle Health Insurance objects in the database. The installation operates on these objects and therefore requires exclusive access to install in a controlled way.

For a subset of interim patch releases 'online' patching is supported, meaning web services with their database sessions may remain active. This is described in more detail later on in this document.

To prevent other active session it is necessary the following conditions be met:

RESTRICTED SESSIONS

At the time of installation of a release (major or interim patch), only restricted sessions are allowed; this ensures that normal users will not have access to the database.
The Oracle Health Insurance owner account, under which the database installation is performed, therefore has the system privilege of a RESTRICTED SESSION.

The status can be obtained in 2 ways:

1. The database has to be started in restricted mode (`STARTUP RESTRICT`).
2. When it is not possible to bring down the database first, ensure that no user sessions are active and that only restricted sessions are permitted (`ALTER SYSTEM ENABLE RESTRICTED SESSION` followed by ending orkilling the non-restricted sessions. For RAC environments this should be executed for all instances but experience has shown it is better to shutdown all instances except one and put the remaining one in restricted session mode; this speeds up the AWR snapshots that are created at specific installation 'landmarks').

When the database needs to become available again, change into non-restricted mode (with a normal startup or `ALTER SYSTEM DISABLE RESTRICTED SESSION`).

BATCH SCHEDULER & OHI BACKGROUND JOBS

Before a release (meaning major or interim patch) is installed, the OHI Back Office batch scheduler has to be stopped through the official stop script, whether it is connected or not. This is needed to bring down the OHI background jobs also, to prevent any interference with the release installation.

OHI 'OBJECT OWNER' ACCOUNTS AND 'USER' ACCOUNTS

No Oracle Health Insurance users, interface sessions (SVL, HSL, PSL service accounts), JMS queue account(s) or query tool users, etc. should be active. The only active session should be the installation session of the Oracle Health Insurance owner account.

By using the restricted session mode any use of OHI objects normally should be prevented. An exception is when using the functionality to install patches which allow web services to remain online, see the 'Online Patching' paragraph.

OHI BO doesn't allow for the OHI object owner accounts (the table owner with a name that may differ per OHI customer, but usually OZG_OWNER, the OHI_VIEW_OWNER and the OHI_DPS_USER account) to be unlocked during regular operation. Only during patching should these accounts be unlocked.

BACKUP REQUIREMENTS (ARCHIVELOG AND NOLOGGING)

OHI Back Office and OHI Data Marts each have specific backup requirements.

This paragraph discusses the requirements related to OHI installation activities.

OHI Back Office

The OHI Back Office production environment needs to run in ARCHIVELOG mode. This ensures that the data can be restored after an incident, by restoring a backup (full or incremental). To restore the data created after the backup was made, a point-in-time recovery can be executed by the application of any (un)archived redo logs.

This method avoids data loss, but only if either:

- No data manipulation actions are executed with the NOLOGGING clause. Actions executed with the NOLOGGING clause are not logged in the redo logs and will result in data loss if a recovery is needed.
- The FORCELOGGING parameter is set to YES. This overrules the NOLOGGING clause.

OHI installations may execute data manipulation actions with the NOLOGGING clause (e.g., conversions) to reduce the installation time. If you set the FORCELOGGING parameter to NO to take advantage of the possible time savings during installation of a major release, you need to make a backup BEFORE and AFTER the installation activities to make sure the data can be restored.

OHI Back Office does not allow installing an interim release with the FORCELOGGING parameter set to NO. The time needed for the backups before and after the installation will not outweigh the installation time saved by setting FORCELOGGING to NO.

OHI Data Marts

An OHI Data Marts environment normally runs in NOARCHIVELOG mode to speed up ETL processing. If a recovery is needed the last full backup needs to be restored. It is therefore advisable for an OHI Data Marts environment to make a *full backup* ***before*** and ***after*** the installation.

The NOLOGGING functionality is used for OHI Data Marts for each installation.

Any data in OHI Data Marts that is lost by recovering the last full backup can be re-created by repeating the ETL jobs.

NOLOGGING explained

When NOLOGGING database functionality is used during an installation, modifications are not logged completely. This makes it impossible to use a restore of a backup that was made before the start of the installation and the subsequent application of any (un)archived redo logs to come to a consistent/usable situation as it existed during **or after** the release installation.

If you need a recovery action during the installation, you can restore of a backup that was made before the start of the installation and re-start the installation. If you need a recovery action after the application has been released to the users, you cannot recover the user actions with just the backup that was made before the start of the installation and the redo log files. You will need a backup that was made after the last NOLOGGING activity (in practice: after the installation was completed).

To summarize:

In case NOLOGGING is used, you will have to make a *full back* up *before and after* the installation, to be able to restore data from the backup, if necessary. A restore based on the backup made before the installation will not be successful, even when you run in ARCHIVELOG mode.

To allow the use of the NOLOGGING option, the database (or the tablespaces we discuss below) should not be running in FORCE LOGGING mode. They usually run with FORCELOGGING = NO, unless a standby database is used.

When FORCELOGGING = YES, the installation may be delayed considerably, but the requirement of a backup before and after the installation is not strictly necessary. The main reason to make these backups is to prevent long restore times, if a restore is needed for some reason.

IMPACT ON CHANGE LOGGING

When a new release is installed database objects may be changed, dropped, disabled, etc. This may impact OHI change logging (modification logging) functionality which might impact custom functionality based on this. For more information please read the white paper 'OHI Back Office - Modification Logging', which is present in docs.oracle.com.

SPECIFIC REQUIREMENTS FOR INSTALLATION OF MAJOR RELEASE

A number of specific requirements apply for the installation of an Oracle Health Insurance *major* release on an Oracle Health Insurance environment.

SPACE MANAGEMENT

Additional space needed

It is important to have a large amount of temporary sorting space available in the standard temporary tablespace. There also has to be sufficient undo space to process long-term operations in 1x. Finally, additional space might be needed for the increased size of tables and indexes or storage of temporary objects.

For the temporary tablespace applies that a minimum of 2 times the (unfragmented) space of the largest index is required.
The following query determines the required MBs:

```
select round((max(bytes)/1024/1024))*2 "Required #MB temp
space"
from   dba_segments
where  segment_type = 'INDEX'
```

```
and     owner         = '<Name OHI owner account>'
/
```

Use temporary space during installation

The following query can be used to monitor the temp space during installation:

```
select tablespace_name
,      total_blocks
,      used_blocks
,      free_blocks
from   v$sort_segment
/
```

OTHER DATABASE SETTINGS

As large indexes may be created during installation or large join actions may be executed, it is important to permanently (!) set the initialization parameters below (sort area to at least 200MB, hash area to at least 100MB) to adjusted values to speed up processing:

sort_area_size=209715200 (or higher)
hash_area_size=104857600 (or higher)

Formerly also the workarea_size_policy had to be set but this is nowadays done automatically during installations. This means the settings above are only used when workarea_size_policy is set by the installation code to manual (just session based!) during the release installation.

Higher values for the above settings mean that the probability that more can be done in memory increases, so the need for and use of temporary space may be reduced which normally means reduced installation times.

Note: Once the installation is finished, there is no need to return these 2 parameters to their original settings.

Note: This only applies for installations on OHI Back Office environments.

PARALLEL EXECUTION

During installation of the Oracle Health Insurance (patch) releases Parallel Execution is used to reduce the time needed for the installation.

To enable maximum use of this function, a number of settings have to be entered in the database.

These settings are active for OHI Data Marts by default, see:

Oracle Health Insurance - OHI Data Marts Administrator Reference

For OHI Back Office applies that these settings are *mandatory* during the installation of a major release. These settings are optional for the installation of patch sets and interim patches.

Parallel Execution is *not* permitted during OHI Back Office *runtime* except for the work gathering phase of the batches.

GENERAL

To use Parallel Execution during installation of (patch) releases, the following database parameter has to be set:

parallel_max_servers=...

Set this parameter to at the most twice the number of CPUs; see also the section *Different settings* for specific situations.

The following parameters should not be set:

parallel_min_servers
parallel_execution_message_size

The above setting manages of the default parallelization ratio (roughly twice the number of CPUs present) and will apply to most environments.

Note: Once the installation is finished, please make sure to return the parameters concerned to their original settings.

SPECIFIC SITUATIONS AND SETTINGS

parallel_max_servers

The above setting will lead to accelerated installation only when sufficient CPU-, I/O- and memory capacity are available.

When this is *not* the case, *or* when other environments and/or applications are running on the server and you would like to prevent that parallel operations use up your server's capacity, the value may be set to an explicitly lower value than the default value.

This can be done by limiting the value for parameter `parallel_max_servers`. Set this parameter to 2 times the amount of CPUs available for this environment; this will normally ensure an acceptable and affordable load to take place.

When there are for instance 6 CPUs and it is desired to fully load only 2 with parallel jobs, the value has to be set to 4. Generally, these 4 processes will keep two processors quite busy.

It is possible that the characteristics of your own environment are different as this all depends on the mutual relations of the throughput capacity of the I/O system and the CPUs speed.

When it becomes apparent during a (test) installation that the system is primarily waiting for I/O operations, it is possible that too many parallel processes are running. In this case, reduce the maximum number of parallel processes by using a lower value for `parallel_max_servers`.

sort_area_size

It is possible that during an installation several parallel server processes temporarily use 200MB of working memory each with the above setting for `sort_area_size`.

This can be the case for at most half of the number of parallel servers available (determined by `parallel_max_servers`). Therefore sufficient working memory needs to be available in the server. This will normally be available, as the application is not in use, but might be a problem on test servers. For test servers the ***sort_area_size*** can be decreased to for example 100MB although this will lead to warnings during instance parameter checks.

This also applies to the `hash_area_size` parameter.

UNZIPPING A RELEASE

The file containing the Oracle Health Insurance major release is a Unix zip file in the format p<Oracle patch number>_< OHI release>_Generic.zip.

Example

p31052224_1020200_Generic.zip

This file can be unzipped by executing the following command in the Unix directory indicated by environment variable $OZG_PATCH.

```
unzip p<Oracle patch number>_< OHI release>_Generic.zip
```

Example

```
unzip p31052224_1020200_Generic.zip
```

DIRECTORY STRUCTURE

Once the release zip file has been unzipped, a <release> directory is created containing the complete release update.

You can now navigate to the release directory:

```
cd <OHI release>
```

Example

```
cd 10.20.2.0.0
```

The directory structure of this Oracle Health Insurance release directory is:

Directory	Source files present
bin	Screens, menus (Forms) (object) libraries and Pro*C sources
conf	Configuration files
doc	Documentation relating to the release.
help	On-line help information
install	Database installation scripts
java	Java-related files
sh	Unix shell scripts
sql	SQL modules
utils	Utilities
xml	XML-related files

TARGET DIRECTORIES

After successful installation, the sources and executables are copied to their target directories (located in the directory indicated by environment variable $OZG_BASE).

After compilation / generation of sources, the following files are created:

Target Directory	Source file Extension	Executable File Extension	Module type
$OZG_BASE/bin	.fmb	.fmx	Oracle*Forms Form
	.pll	.plx	Oracle*Forms Library
	.mmb	.mmx	Oracle*Forms Menu
	.olb	<no executable>	Oracle*Forms Object Library
	.pc	<no extension>	Oracle Pro*C
$OZG_BASE/conf	<various>	<no executable>	Configuration files
$OZG_BASE/help	<various>	<various>	Online help information
$OZG_BASE/java	.ear, .java, .jar	<no executable>	Java-related files

Target Directory	Source file Extension	Executable File Extension	Module type
$OZG_BASE/sql	.sql	<no executable>	SQL module
$OZG_BASE/sh	.sh, .pl, .ctl	<no executable>	OS shell scripts, Perl scripts, SQL*Loader Controlfiles
$OZG_BASE/utils	<various>	<no executable>	Utilities
$OZG_BASE/xml	.xsd, .xsl, .xslt	<no executable>	XML-related files

N.B. : When a new version of the Oracle*Forms (object) library files OZGLIB.pll (and executable) and/or *.olb and/or reference forms OZGREF*.fmb are installed, *all* Oracle*Forms source files (.fmb) have to be *recompiled* (.fmx) with menu option 800.

RELEASE DOCUMENTATION

The release documentation is located in the release documentation subdirectory 'doc' in HTML format (<*release*> *.*htm*). This documentation contains the description of all changes delivered via the release and a summary of the released sources.

Specific details, like test plans, specific installation issues etc., are recorded in separate documents in this documentation directory.

The documentation also contains an overview of files in the release.

THE README.TXT FILE

The documentation directory contains a `readme.txt` document.
This document contains a short description of all steps, and the order in which they have to be performed, to install the release.

USE OF VARIABLES DURING INSTALLATION

VARIABLES DEFINITION

During installation variables will be used when necessary.

Different types of variables can be used: *generic* variables, which are generic for all environments, *environment-specific* variables, which are specific for a certain environment and *(patch) release-specific* variables, which are specific for a certain release.

(Patch) release-specific variables

When variables are used that are specific to this release (this may be the case in .sql scripts), these variables are entered in the file `readme.<PRODUCT>` (where PRODUCT is OZG for OHI Back Office and OBD for OHI Data Marts).

The file is located in the release directory.

This file will be (automatically) *copied* and then *modified* to a file named **`readme.<PRODUCT>.<$TWO_TASK>`**, eg. `readme.OZG.prd`, so that this file is specific for the release concerned in the specific environment (the Oracle Health Insurance .sql scripts also contain a call to `readme.OZG.$TWO_TASK`, *not* to `readme.OZG`).

In this way it is possible to retrieve with which settings the installation has run *for any release on any environment for any product at any given moment*.

IMPORTANT: The installation menu does copying automatically within the $OZG_PATCH release specific folder; the installation menu will also provide the interface to modify the values for the variables.

RESTARTING

The data of the installed release are stored in the `ALG_RELEASES` table during installation. If this release has been installed before in the current environment, an message will occur; *reinstallation* is an option.

Installation of a release can always be restarted.

When errors have occurred during the first installation, these can be solved after which the installation can be restarted. An example is a *space management error* upon creation of a table (e.g. insufficient space in a tablespace).

SQL SCRIPTS

SQL scripts that adapt data may be part of a release.

SQL scripts should always contain:

- totals of changed and rejected and / or erroneous records;
- details of rejections or errors;
- Start- and end time of the script;
- Facilities for monitoring of the script by the DBA (not applicable for check scripts, as these only contain selections instead of updates/deletes) when the script is expected to run a considerable time on a customer environment.

The following functionality is optional (when judged to be necessary by Oracle):

- the way in which data can be restored (manually);
- details of the changed records (n/a for check scripts).

There are 2 categories of SQL scripts: Check scripts and automatic SQL scripts.

CHECK SCRIPTS

Check scripts are meant to help install a (patch) release without functional installation errors; they check for the presence of inconsistent data; only selections are performed. They may also determine certain actions have to be executed before the actual release can be started.

These scripts are always executed before the actual installation takes place. It is not always known prior to the installation if data inconsistencies exist and if they do exist, how they can be resolved. It is not always possible to detail how to resolve inconsistencies detected by the scripts. Check scripts are never allowed to adjust data.

Check scripts can also be started prior to (days or weeks before) an installation. This is a good habit as it can indicate early time consuming actions that need to be executed before the actual release installation can be started.

AUTOMATIC SQL SCRIPTS

Automatic SQL scripts run automatically as part of the database installation part of the release installation.

These scripts perform changes (insert, updates and/or deletes) on data.

WALLET ENTRY

WALLET ENTRY FOR APPLICATION OWNER ACCOUNT

The database login for the application owner, like the account for the batch scheduler, must be defined in a Wallet.

N.B. : For security reasons, the wallet for the application owner should be a separate wallet. Otherwise, the OS user “batch” will be able to log on to the database as the application owner, which defeats the reasons for having a separate OS user “batch”.

The Wallet entry must be named `<$TWO_TASK>_install`, and must login to the application owner in the corresponding database. For example, when the database is named `prd` and the owner is named `OZG_OWNER`, a Wallet entry and tnsnames entry `prd_install` must be created, such that the following command:

```
sqlplus /@prd_install
```

logs in into the database and the sqlplus command:

```
SQL> show user
```

displays `'USER is "OZG_OWNER"'`.

See the ‘Oracle Health Insurance Back Office Installation, Configuration and DBA Manual’ (docs.oracle.com) for more details on how to create this secure wallet entry.

4. INSTALLATION MENU OHIPATCH

INTRODUCTION

Installation menu `OHIPATCH.pl` is intended for the automated installation of Oracle Health Insurance (patch) releases.

The installation menu shows a menu with activities to be performed based on the release properties file `$OZG_PATCH/<release>/doc/<release>.prp`.
It is determined per product line (one has a license for) which actions have to be performed. It can then be used to perform installation of objects in the database and on the filesystem.

The following sections describe the functionality of the menu and examples indicate what the properties and options are.

PERL MODULES

To run the menu, it is required to install a number of additional Perl modules; see

Oracle Health Installation, Configuration and DBA Manual
Appendix D – Installing required Perl modules

GENERIC MENU FUNCTIONALITY

START

The `OHIPATCH.pl` script can be found in the `$OZG_BASE/utils` directory.

The script is started as operating system user `oracle` with the name of the environment (typically equal to the database name) to be patched as the parameter.

$OZG_BASE/utils/OHIPATCH.pl <name environment/database>

It is not mandatory to be in a certain directory when `OHIPATCH` is used provided you specify the environment specific path where it is located.

Configuration

First time that `OHIPATCH` is used for an Oracle Health Insurance environment, some questions will be asked.

The menu `OHIPATCH` will logon using the wallet entry **`<$TWO_TASK>_install`** as mentioned in the previous Chapter.

List of releases and patches to be installed

Subsequently, you need to indicate which release or interim patch has to be used to start the patching and which release or interim patch is the last to install.

The chosen values are stored in a configuration file and will be the default choices for the next installation.

Whenever a range of releases or patches is selected, the installation menu will *propose* what releases and interim patches in this range need to be installed.
After this proposal, the user will get the opportunity to override the proposed

selection.
The following algorithm is used for the proposed selection:

1. The provided begin and end release ("*boundaries*") will be selected (unless it's an interim patch that does not require installation; see 4 & 5);
2. All *major releases* in the range will be selected;
3. All interim patches on the last major release in the range will be selected;
4. All *interim patches* will be selected when a range of interim patches is selected *for the same major release*;
5. In case of migrating to another major release, do *not select intermediate patches;*

By default, step 120 is postponed and executed only for the last release in the list of major and patch releases to be installed. If the last release in the list is not valid for the current environment, an error message will be given: "ERROR: Last patch in selection of releases must be valid for current environment.".

Preconditions

Subsequently, a number of preconditions (i.e. availability of utilities needed and use of the recommended directory structure) will be checked.

When these checks do not result in any issues, the menu will appear and show the first release or interim patch selected.

When the selected release (interim patch release, a major release is meant for both product lines) does not belong to the correct product line (e.g. OHI Back Office and OHI Data Marts) this will be reported and the release will automatically be skipped.

Starting with release 10.22.7.0.1, the OHI patch program will check if the pre-requisite (patch) release has been installed, before starting the installation of that patch. Absence of the pre-requite (major) release will result in a fatal error. Absence of the pre-requite (interim) release will result in a error in auto mode, and a question in interactive mode.

FILE AUTHORIZATION

The files created by the installation menu are created with access rights (`umask 077`) so that these can only be read under the account used to start the installation menu (normally `oracle`).
Other accounts cannot view these files.

THE MENU

The menu shows the activities for each release. The overview of the activities to be performed per product line can also be read in the `$OZG_PATCH/<release>/doc/readme.txt` file.

Based on the Oracle Health Insurance release 10.20.2.0.0a (fictitious) example of the menu is included:

```
+--------------------------------------------------------------------+
|            Oracle Health Insurance installation menu               |
+--------------------------------------------------------------------+
  Release 10.20.2.0.0, database prd

  (+) A v Preparation
  (+) B v Database installation
  (+) C v Filesystem installation
  (+) D v Completion
  (+) ?   Help

+--------------------------------------------------------------------+
Make your choice [10]:
```

When this menu has been expanded completely, the following is visible:

```
+--------------------------------------------------------------------+
|            Oracle Health Insurance installation menu               |
+--------------------------------------------------------------------+
  Release 10.20.2.0.0, database prd

  (-) A v Preparation
           10 v Fill parameter file
           20 v Process specific installation instructions
           30 v Place installation modules
           90 o Perform object validation
          100 v Run SQL check scripts
  (-) B v Database installation
          110 v Run database installation script
          115 v Validate non-validated constraints
          120 v Process synonyms, grants and schema compilation
  (-) C v Filesystem installation
           C1 v Complete filesystem installation WITHOUT compilation/generation
           C2 v Complete filesystem installation WITH   compilation/generation

          200 v Place OS modules
          205 v Place OHI Data Marts OS modules
          210 v Place SQL modules
          215 v Place OHI Data Marts SQL modules
          220 v Place online help information
          230 v Drop obsolete objects
          240 v Place Java/XML-related files
          250 v Place database DDL scripts
          260 v Place configuration files
          270 v Deploy ear files
          300 v (Compile and) place (v 300CP) Pro*C modules
          500 v Place reference forms
          510 v Place Forms object library
          520 v (Generate and) place (v 520CP) Forms libraries
          530 v (Generate and) place (v 530CP) forms and/or menu's
  (-) D v Completion
```

```
          800 v Compile/Generate objects in $OZG_BASE/bin
          820 o Gather table/index statistics
          830 o Show Business Rules validation status
          840 o Drop obsolete columns
          850 o Partition/compress tables
          855 o Drop old/empty partitions
          860 o Move tables/indexes
          870 o Rebuild unusable indexes
          880 o Enable/disable VPD policies
          890 v Process manual deployment instructions
          900 v Perform object validation

  (-) ?   Help
          _____________________________ Symbols _____________________________
          v Mandatory execution           * Successful execution
          o Optional execution            x Errors after execution

          ____________________________ Menu control____________________________
          - Open entire menu              + Close entire menu
          A Submenu A open/close          C Submenu C open/close
          B Submenu B open/close          D Submenu D open/close

          ____________________________ Menu options ____________________________
          L Show log file for last choice M Automode
          O Skip default choice            R Read readme.txt file
          S Reset log file(s)              T Show logfile timing
          U Update client releasetoken     N Next (patch)release
          Q Quit                           Z Create logfiles zip
          $ OS command line                @ SQL*Plus (ozg_owner)

          _____________________________ Info ______________________________
          Perl version: 5.8.5             RCS Version: 4.348
                                                                        [-]
+--------------------------------------------------------------------------+
Make your choice [10]:
```

COLLAPSING AND EXPANSION OF THE (SUB-)MENU

The entire menu can be expanded by keying in - (minus).
The entire menu can be collapsed by keying in + (plus).

A submenu can also be collapsed or expanded by entering the code in the submenu; `A`, `B`, `C`, `D` or `?`.
The codes entered are *case insensitive*.

CHOOSING OPTIONS

An option can be chosen by entering the option number, e.g. `100`, `410`, `530`, etc.

The generic options can be chosen by entering the indicated code: `L`, `M`, `S`, `Q`, etc.

These codes are also *case insensitive*.

ANSWERING QUESTIONS

Answering a question can only be done with `Y`, `y`, `N`, `n` or <enter>.
For backward (Dutch) compatibility, `J` and `j` are also valid answers (to act as `Y`).
When <enter> is given, the answer as proposed between `[ ]` is chosen as default.
The proposed answer does not have to be copied.

LOG FILES

Naming

When an activity is chosen and executed, a log file is created in the following format `<release>.<activity>.log.<database SID>.`

When an activity is performed again, the existing log file will be renamed; a date/time component will be concatenated at the end of the name.

Example:

10.20.2.0.0.110.log.prod

This becomes:

10.20.2.0.0.110.log.prod.Mon_01-Mar-2014_14:11:35

Showing logfiles

To view an activity's log file, the option `L` has to be chosen.
The log file of the last activated activity will then be shown; in this case the "*context*" of the menu is set to this last activated activity.

When you want to view a log file which is not the last activated one, the *context* should first be set to this activity.
This can be done by choosing the activity, and answering the following question '`Do you want to execute activity … again?`' with `N` (No). It will subsequently be possible to choose option `L` again.

To view *all* of a submenu's activities, again the *context* has to be set to the submenu, by choosing it first (e.g. `A`; it does not matter if the submenu is expanded or collapsed with this choice, the only requirement is that this is the current choice), and then option `L`.
To view all log files for *all* activities for the *complete* menu, the main menu has to be chosen first (with + or -) and subsequently option `L`.

The *active context* can be seen on the *right bottom of the menu*; in the expanded menu example above the context is set to -, as indicated by `[-]` on the last line.

Viewer used

To show the log files the installation menu uses the `more` utility.

ERRORS

During the performance of a menu activity, the following errors may occur (which will be registered in that activity's log file):

1. `ORA-`
   Oracle DBMS errors.
2. `IMP-`
   Errors during execution of the Oracle import utility.

3. `ERROR:`
   Errors during installation of an Oracle Health Insurance (patch) release.
   The installation will continue.
4. `PL/SQL ERROR`
   Errors during compilation of Oracle Health Insurance customer sources.
5. `SP2-`
   SQL*Plus errors during installation of objects in the database.
6. `RTC-` en `OMBXXXXX`
   Oracle Warehouse Builder Control Center Service errors during OWB mapping packages installation in the database with the `OMBPlus` utility.
7. `FATAL:`
   Fatal installation errors during an Oracle Health Insurance (patch) release installation. The installation is aborted.

At the bottom of the screen, at the end of the log file, before
"`Press <Enter>` " (press a key) it will state if any errors have occurred.
Before patching continues, each technical error will have to be resolved.

In some cases, especially when installing a range of patches, one might want to suppress certain 'known errors' which are resolved in subsequent patches. Using this mechanism is only useful when using the automatic installation or silent mode. Details about the usage of the error suppressing mechanism can be found in Appendix C.

Messages with the following prefix are meant as a warning and will for instance occur when a patch is installed for the second time or when invalid objects still exist at the beginning of an installation. An installation is not perceived as erroneous in case of a warning:

- `WARN :`

Other messages with the following prefix are informative:

- `INFO :`

Messages with the following prefix mark the end of an acitivity. A message with this prefix is added to all activities in the menu's A to C and activities 800, 890 and 900 in the D menu. It enables the patch menu to verify if an activity has ended successfully:

- `END :`

If this is not the case a `FATAL :` error will be logged: "[OHILOGSCAN.pl] Activity [activity] did not complete gracefully."

OPTIONS DESCRIPTION

SYMBOL USE

o = Optional execution

When an option is optional (and is not mandatory for this (patch) release) but has not been executed yet, an `o` will appear in front of the option.

v = Mandatory execution

When an option is mandatory but has not yet been executed, a *v* will appear before the option.

* = Successful execution

When an option has been executed successfully, an * will appear in front of the option. This applies to optional and mandatory options.

x = Errors after execution

When execution of an option has caused errors or the activity was aborted, an *x* will appear in front of the option. This applies to optional and mandatory options.

Examples

Example 1 – Option 800 is mandatory

800 v Compile/Generate objects in $OZG_BASE/bin

Example 2 – Option 800 is optional

800 o Compile/Generate objects in $OZG_BASE/bin

Example 3 – Option 800 was executed successfully

800 * Compile/Generate objects in $OZG_BASE/bin

Example 4 – Option 800 was not executed successfully

800 x Compile/Generate objects in $OZG_BASE/bin

Sub-menus

The following rules apply for submenus

1. When 1 or more activities in the submenu are mandatory and have not yet been executed, the submenu will be indicated as mandatory and not yet executed.
2. When 1 or more activities in the submenu have not been executed successfully, the submenu will be indicated as not successful.
3. When there are no mandatory activities in the submenu and only successfully executed activities exist in the submenu, but only (still) optional activities, one or more of which have not yet been executed, then the submenu will be indicated as optional.
4. When only successfully executed activities are present in the submenu, the submenu will be indicated as successfully executed.

A - PREPARATION

The options in this submenu are used to execute preparatory activities, like the preparation of installation modules and utilities, which may be used.

10 Fill parameter file

Filling of the release-specific parameter file, needed for the database installation of the (patch) release.

This action has to be performed manually, which is why confirmation of its execution is requested explicitly.

Editor used

The editor is used to fill in the parameter file the way it is set in the OS shell variable `$EDITOR`.

When it is not set, `vi` is used.

20 Process specific installation instructions

Specific installation instructions are shown; these have to be processed first.

This action has to be performed manually, which is why confirmation of its execution is requested explicitly.

30 Place installation modules

Placing installation modules in `$OZG_BASE/<subdirs>`.

Care is taken during this activity that any higher versions of the objects concerned are not overwritten.

This check is performed because it concerns objects in a generic / common directory, which means that they apply to several releases.

90 Perform object validation

Performing an Object Validation; used to check the presence, versions and status of the Oracle Health Insurance objects.

Choose to check only the database objects [D], only the filesystem objects [F] or both [B].

This option is optional and can be used to check the environment before a release is installed.

This option is *always* visible in the menu.

100 Run SQL check scripts

Running of (patch) release SQL check scripts.

Note: This option can and should be run as soon as the (patch)release becomes available; this way the possible reported issues can be fixed in time, before the actual installation of the complete (patch)release.

B - DATABASE INSTALLATION

The (patch) release database installation activities are executed through options in this submenu.

110 Run database installation script

Running of the database installation script. For major releases and patchset releases this step usually consumes most of the installation time. All database objects are updated and data structure of tables are changed during this step when necessary.

Please see Appendix D for checks that are executed when this step is started.

115 Validate non-validated constraints

Validation of constraints with status `NON-VALIDATED` under the Oracle Health Insurance owner account.

This option does not apply to OHI Data Marts.

A constraint for an existing table is always created `ENABLED NOVALIDATE` for an Oracle Health Insurance database installation; which means that the constraint is enforced to all *new* records entered afterwards constraint enablement.

To validate the constraint for the *existing* records this option is used.

The number of existing records which cannot be validated is indicated per table/constraint where applicable. As this usually means application data is no longer consistent and reliable this endangers the stability of the application and should result in contacting Oracle Health Insurance Support unless it is a known and acceptable issue that will be solved at a later moment. When you can correct the data through the application to make the data comply to the constraint that is of course a preferred way of solving these issues.

This option is *always* mandatory; as all constraints present always have to be validated.

120 Process synonyms, grants and schema compilation

The creation of private and public synonyms for the Oracle Health Insurance objects, the revocation and distribution of grants to the Oracle Health Insurance database roles and schemas as well as compilation of still invalid objects in the Oracle Health Insurance owner schema.

When patching a list of (patch) releases step 120 will be postponed to the last (patch) release in the list to reduce the run time of patching by executing this step only once per range of patches. A message will be logged in the logfile of step 120 for all but the last patch: “INFO : Installation step 120 is postponed to patch <patch>”.

Grants to roles and accounts that are not supported and allowed for security and stability reasons will be revoked automatically. Make sure the consequences are reviewed beforehand.

When Warnings or Errors occurred during this step not all revokes could successfully be executed and you need to take action.

Depending on your custom code and privileges you granted in the past you may have to manually revoke a number of privileges. The commands for these revokes will be listed and you can simply copy/paste the output text to a SQL*Plus session connected as the OHI table owner and ignore the errors caused by the surrounding text.

When a lot of additional messages of succeeded revokes surround these isolated failed revokes you may simply re-execute step 120 to get a list of only the failed revokes.

In rare cases the revoke may take a lot of time due to a deadlock timeout while waiting for a conflicting session. Be sure there is no running OHI process (batch, background, custom code, etc.) to minimize the possibility for a deadlock timeout to occur.

An example of a message that indicates you have to execute a revoke manually (as the OHI table owner):

```
WARN : Privilege EXECUTE on ALG_DPS_PCK could not be revoked from TST_OWNER due
       to dependency being present. Please execute manually:

       revoke EXECUTE on ALG_DPS_PCK from TST_OWNER;
```

C - FILESYSTEM INSTALLATION

The (patch) release filesystem installation activities are executed through options in this submenu.

C1 Complete filesystem installation WITHOUT compilation/generation

When this option is chosen, the below numbered filesystem installation activities will be performed, *without* compilation/generation taking place (see the terms in brackets for the numbered activities; these will *not* be executed here); *this still will have to be done at a later stage!*

C2 Complete filesystem installation WITH compilation/generation

When this option is chosen, the below numbered filesystem installation activities will be performed, *including* compilation/generation taking place (see the terms in brackets for the numbered activities; these *will* be executed here); *this completes the filesystem installation!!*

200 Place OS modules

Placing OS modules in the `$OZG_BASE/sh` directory.

This option is not used to process the OHI Data Marts OS modules (see 205).

205 Place OHI Data Marts OS modules

Placing OHI Data Marts OS modules in the `$OZG_BASE/sh` directory.

210 Place SQL modules

Placing SQL modules in the `$OZG_BASE/sql` directory.

This option is not used to process the OHI Data Marts SQL modules (see 215).

215 Place OHI Data Marts SQL modules

Placing OHI Data Marts SQL modules in the `$OZG_BASE/sql` directory.

220 Place online help information

Placing online help information in the directory `$OZG_BASE/help` directory. After placing the online help files, all `$OZG_BASE/java/HSL*swagger.json` and `$OZG_BASE/xml/HSL*.xsd` files are copied to the online help directory to make sure the latest version of these files are included in the online help.

This option only applies to OHI Back Office.

230 Drop obsolete objects

The deletion of obsolete objects (sources & executables) on the filesystem.

Obsolete objects are objects that no longer apply. These objects have to be deleted from the runtime environment. The object source file and the object executable are deleted from the filesystem.

This only applies for objects not present in the database, but only on the filesystem. Database objects are, when applicable, automatically deleted from the database installation and are not listed out either.

240 Place Java/XML-related files

Placing Java/XML-related files in the `$OZG_BASE/java` and `$OZG_BASE/xml` directories. All HSL* files are also copied to the online help directory to make sure the latest version of these files are included in the online help.

250 Place database DDL scripts

Placing database DDL scripts (for the creation of tables including the constraints, indexes and triggers concerned) in the `$OZG_BASE/install` directory.

The scripts are (for the time being) used when option 850 is chosen to partition table(s).

260 Place configuration files

Placing configuration files in the directory `$OZG_BASE/conf` directory.

270 Deploy ear files

Deploy ear files.

This option only applied formerly to OHI Self Service. It is maintained for future usage.

300 (Compile and) place (300CP) Pro*C modules

The compilation (only in case of C2) and placing of the Pro*C modules in the `$OZG_BASE/bin` directory.

This option only applies to OHI Back Office.

430 (Generate and) place (430CP) OHI Data Marts reports

Compilation (only in case of C2) of the OHI Data Marts reports.

Activity 430CP (is executed automatically as part of activity 430, but can also be (re)started separately) will subsequently place the files in the `$OZG_BASE/report` directory.

500 Place reference forms

Placing of the reference forms in the `$OZG_BASE/bin` directory.

This option only applies to OHI Back Office.

510 Place Forms object library

Placing of the Forms object library in the `$OZG_BASE/bin` directory.

This option only applies to OHI Back Office.

520 (Compile and) place (520CP) Forms libraries

Compilation (only in case of C2) of the Oracle*Forms pl/sql libraries.

This option only applies to OHI Back Office.

Activity 520CP (is executed automatically as part of activity 520, but can also be (re)started separately) will subsequently place the files in the `$OZG_BASE/bin` directory.

530 (Generate and) place (530CP) forms and/or menu's

Compilation (only in case of C2) and placing of the forms and/or menus in the `$OZG_BASE/bin` directory.

Activity 530CP (is executed automatically as part of activity 530, but can also be (re)started separately) will subsequently place the files in the `$OZG_BASE/bin` directory.

This option only applies to OHI Back Office.

D - COMPLETION

The options in this submenu are used to perform activities that can be performed after the installation of 1 or more (patch) releases.

800 Compile/Generate objects in $OZG_BASE/bin

This option only applies to OHI Back Office.

Compilation of sources in the `$OZG_BASE/bin` directory.

It is mandatory to execute this option when option C1 is chosen for the filesystem installation (no compilation/generation) and the options 300 or 520 or 530 are present. Compilation of *all* sources will then have to be chosen.
In other cases, this option is optional.

Even when other generic actions have been performed, e.g. the installation of a Developer Patch Set, it is still mandatory to execute this activity.
This is why this option is *always* visible in the menu.

This option offers the possibility to compile specific sources; the option to not compile all sources will have to be chosen, in that case.
Subsequently, a prefix and an extension can be given. When no value is given, no restriction applies for the determination of the sources to be compiled.

Beware, even when this action is optional it is strongly advised to always execute this action at the end of a series of release installations (after the last release has been installed). Reason is that a patch that does not change a client source may change the 'database interface' by for example recreating a table with a different column order, add a column, add a procedure/function parameter with default, change the order of such parameters. Such changes may not result in a change in a $OZG_BASE/bin source but may influence the way the compiled code needs to access the database. So when you are not sure whether such a change was incorporated in the installed patches it is advised to always execute this action.

Example 1 – give prefix and extension

Prefix `SYS11`, extension `pc` results in compilation of `SYS11*.pc`, causing source `SYS1108S.pc` to be selected.

Example 2 – give prefix

Prefix `OZGA` results in compilation of `OZGA*.*`, causing sources `OZGABOUT.fmb` and `OZGAMENU.mmb` to be selected.

Example 3 – give extension

Extension `pc` results in compilation of `*.pc`, causing sources `SYS1108S.pc` and `SYSS004S.pc` to be selected.

Tip: When the timestamp/signature of a stored object, a screen refers to, is changed, the following error message will occur during use of the screen:

ORA-04062: timestamp/signature of %s has been changed

The following message is also possible, in that case:

ORA-01403: `no data found`

By recompiling the source(s) concerned, these messages are resolved.
To *always* prevent this message, one can choose to always execute activities 800 after installation of a release.

Tip: The compilation / generation of step 800 in itself can be performed with multiple processes.

In $OZG_BASE/conf/ohipatch.conf the parameter ozg_max_comp_processes can be altered to set the maximum number of processes used for step 800.

The parameter defaults to 1. A value equal to the number of available application server CPU cores is recommended.

820 Gather table/index statistics

Collecting table/index statistics under the Oracle Health Insurance owner account with the `$OZG_BASE/sql/OZGISTAS.sql` script.

This option is also *always* visible in the menu; given that this activity will always have to be executed at the end of an installation. It is up to the installer to decide when this activity is to be performed.

830 Show Business Rules validation status

This option only applies to OHI Back Office.

This option produces a report of the status of the validation of static Business Rules under the Oracle Health Insurance owner account that are not implemented by a declarative constraint because of their complexity.

This option is *always* mandatory; as all checks to data present always have to be validated.

This option is also *always* visible in the menu; as this activity can always be executed during an installation.

How to validate Business Rules

The actual validation *itself* is a batch in the OHI Back Office application (“Validate Business Rules”), and should be started using the application menu.

Running this batch, validating all non validated static Business Rules, may take a long time, therefore a parameter may be used to indicate how many hours this step may take at the maximum. If not all Business Rules can be validated in this period, the remainder may be validated at a later stage.

No later than *two weeks after the release goes live*, all Business Rules have to be validated.

A static Business Rule (= a complex constraint on a table) is always `ENABLED` for an Oracle Health Insurance database installation; i.e. the Business Rule applies to all *new* records.

To validate the Business Rule for the *existing* records, the batch is used.

The number of records that cannot be validated is indicated per table/Business Rule where applicable.

To query the status of the static complex Business Rules please use table ALG#BUSINESS_RULES. To view all records in the application that do not comply to a static rule after validation please query ALG#BUSINESS_RULE_VALIDATIONS.

Likewise declarative constraints violations of static complex business rules should be resolved as they endanger the stability and reliability of the application. Please consult the description for step 115 for further information.

840 Drop obsolete columns

Deletion of obsolete columns under the Oracle Health Insurance owner account.

Obsolete columns are deleted *logically* during the database installation to reduce installation time, which means that these columns can no longer be used (but are still invisible present in the tables concerned).

This activity physically deletes the columns concerned and can be time consuming.

This option is also *always* visible in the menu; as this activity can always be executed during or after an installation.

850 Partition/compress tables

The partitioning/departitioning or compression/uncompression of tables under the Oracle Health Insurance table owner account.

The partitioning of tables can only be influenced in OHI Back Office. OHI Data Marts requires and uses partition functionality by default. Compression of tables is offered for both OHI Back Office as well as for OHI Data Marts, but the implementation differs.

A limited number of tables within OHI Back Office is eligible for partitioning, mainly to improve scalability when using parallel processing with increasing numbers of parallel processes. A partitioning key is determined (by OHI) for these tables as well as a division into partitions. The set of tables within OHI Back Office that support partitioning might slowly be increased over the years depending on the necessity.

For OHI Back Office all ‘transactional’ or ‘fact’ tables and indexes - in other words tables not used to register setup data - can be compressed using Advanced Compression, if they have either a Primary Key, or a Unique Key without column(s) that can be NULL. Please note that Advanced Compression is a licensed option of the database.

For OHI Data Marts the compression functionality implementation differs from OHI Back Office and is described in the Administrator Reference for OHI Data Marts.

Once a table is partitioned or compressed, this same menu choice is available for departitioning/decompression of these tables, when applicable.

For more information about partitioning and compression, see the chapter Partitioning & Compression.

This option is *always* visible in the menu.

855 Drop old/empty partitions

This option only applies to OHI Data Marts.

Deletion of old and empty partitions of partitioned tables. These partitions will no longer be used and can be removed.

This option is also *always* visible in the menu.

860 Move tables/indexes

All tables and indexes must be stored in specific tablespaces. Warnings are raised during installation step 110 and during object validation (step 900) when tables and/or indexes are not in the correct tablespace. This option can be used to move tables and/or indexes to the correct tablespace.

This option cannot be used to move tables and/or indexes to an arbitrary tablespace.

This option is also *always* visible in the menu.

870 Rebuild unusable indexes

When a table has been moved, the indexes on the table still refer non-exiting ROWIDs and become UNUSABLE. Some installation tasks may also result in unusable indexes, depending on how for example a conversion has been implemented.

As such it is a good practice to always run step 870 after a set of patches has been installed or after some maintenance work has been executed, to rebuild potential unusable indexes.

This option can be used to rebuild unusable indexes.

This option cannot be used to rebuild usable indexes.

This option is also *always* visible in the menu.

880 Enable/disable VPD policies

Enabling or disabling VPD policies; used to activate (or deactivate) VPD policies in a database environment.

For VPD policies to be effective, setup in the application is required.

Be aware that disabled VPD policies may imply that sensitive data is available for anyone with access to the database.

The available policies to enable or disable will be presented in the menu.

890 Process manual deployment instructions

If applicable, instructions will be shown for manual deployment of:

- OHI Data Marts .rpd file(s), located in the /report subdirectory of the patch release
- OHI Back Office Analytic Cloud .rpd file(s), located in the /report subdirectory of the patch release
- OHI Back Office Analytic Cloud .catalog file(s), located in the /report subdirectory of the patch release.
- OHI Back Office Service Layer .ear file(s), located in the /java subdirectory of the patch release.
- OHI Back Office HTTP Service Layer .war file(s), located in the /java subdirectory of the patch release.

- OHI Connect to Back Office .ear file(s), located in the /java subdirectory of the patch release.

This action has to be performed manually, which is why confirmation of its execution is requested explicitly.

Beware: When you install a range of patches and in one or more of these patches manual instructions are present step 890 will be mandatory for these patches. Even when you confirm the mandatory 890 steps in the respective releases you will get a warning in later releases in that range when you proceed to a next release of leave the menu and have not confirmed the optional step 890 in that release. This mechanism only serves as a 'reminder' (or an 'alarm bell') to pay attention to the instructions as delivered in the relevant releases. You need to interpret these instructions and determine yourself what is relevant for the environment at hand.

Only when you acknowledge the execution of step 890 in the latest release (even if it is optional in that patch release) of a range the waning will not be given when you leave OHIPATCH.

Further information about the deployment steps to be taken is available in the documentation of the respective products.

900 Perform object validation

Performing an Object Validation; used to check the presence, versions and status of the Oracle Health Insurance objects.

Choose to check the database objects, the filesystem objects or both.

This option is always mandatory.

Tables and indexes must be stored in specific tablespaces. When tables and/or indexes are not in the correct tablespace, then during the object validation warnings are reported. Installation step 860 can be used to move tables and/or indexes to their correct tablespaces.

Note that when the collection of statistics and the validation of Business Rules has not (yet) been performed/completed, this will be reported by the Object Validation.

Note: Always *start* and *end* an installation session with an object validation.

L - SHOW LOGFILE FOR LAST CHOICE

Show the log file of the activity, which was chosen last.

M - AUTOMODE

This option makes it possible to install the (patch) releases completely automatic.

The mandatory options in submenus A, B and C are executed automatically for the selected range of (patch) releases.

Automode can be enabled at any time; the installation will run automatically from that point onwards.

Automode will stop processing in case of an error (unless it is a known error that may be skipped, please see the Appendix regarding this topic).

Execute D menu activities in Automode

Automode will automatically skip the mandatory activities from the D submenu for each (patch) release it processes.

It will, however, keep track of what mandatory D menu activities should be executed.

When Automode finishes the automatic installation, it can then either report the mandatory D menu activities that need to be executed, or execute them.

This behavior is managed by the configuration file parameter `ozg_run_dmenu_in_automode`;

When set to `N`, the mandatory D menu activities are reported, not executed (this is the default behavior).

When set to `Y`, the mandatory D menu activities are executed automatically.

Conditions

Conditions for a completely automatic installation are that the following *manual activities* for the selected releases *must have been executed*

10

The release-specific parameter files have to be filled out previously for an automatic installation of database objects (submenu B).

20

The specific installation instructions have to be processed beforehand, because they may contain actions specified for a successful further installation of database- and/or customer objects.

O - SKIP DEFAULT CHOICE

The menu item is automatically preselected for `"Make your choice …"` (the so-called *default* choice); this is the first mandatory activity that has not yet been performed.
When an activity is performed, however, causing errors, *this* will always be shown first, as the errors have to be resolved first. This is why in that case the next activity is *not* automatically preselected.

There is always the option *not* to execute the default choice.

To prevent this from being the default choice (*within the same (patch) release*) *each time* for each following activity, this function can be used to indicate that this activity (within this (patch) release) should not (yet) be performed and should therefore not be selected as the default choice.

Example 1 – 120 postpone

When a range of patches is installed, the option exists to execute activity 120 only for the last patch.
By choosing O, when activity 120 is suggested as the default choice, the activity is skipped and (for this patch) no longer preselected as default.

Example 2 – 110 delivers errors

When activity 110 results in errors during patch installation, this activity will be suggested as the default choice (until errors have been resolved).

When the installation has to be completed, however (e.g. because the error concerns a known installation problem), O may be chosen again, so that the activity will be skipped (for this patch) and will no longer be preselected as default.

R - Read Readme.txt File

Show the `readme.txt` file, the "plan of action for the installer".

S - Reset Log File(s)

`OHIPATCH` determines whether or not an activity has been executed and the result of the possible execution of an activity based on the presence and content of the activity log file.

This option resets current log files for the selected activity (activities), i.e. a backup is made (conform re-execution of the activity, see section Log files).
This achieves that `OHIPATCH` sees the activity as not yet executed and will thus indicate if the activity has to be executed mandatorily or optionally.

By default, this option resets the log file for the activity to which the *context* is set.

T - Show Logfile Timing

Showing the log file in which the installation process' overall timing is registered.

A start and end time can be found for each chosen activity.

This log file can for instance be used to determine the total installation throughput time.

Warnings on mandatory activities which have not been executed for a specific (patch)release (e.g. on exiting the menu or navigating to a next (patch)release), are also recorded in this logfile.

U - Update Client Releasetoken

Save the name of the *highest* release installed, for which (at least) the database installation has been successful, in the `$OZG_BASE/ozg.conf` configuration file.

This information is used by `$OZG_BASE/utils/OHI_CMD.pl` to determine if a certain release has been installed. This data is needed to determine against which Oracle system software certain executables or utilities have to be started.

This activity is *automatically* performed by the installation menu in case of a successful database installation (activity 110) for a release.

The information with regards to the highest installed release is therefore not only available in the *database*, but also on the *filesystem* in the customer software in `$OZG_BASE` belonging to the release.

The activity can be performed manually, whenever desired, e.g. when the token has been changed/deleted as a result of manual management activities; the use of this activity will bring the highest release in the database and on the filesystem *in sync* again.

N - Next (Patch) Release

When a range of (patch) releases is given during the start of OHIPATCH, it can be used to navigate to the next (patch) release.

For that next (patch) release, the check for the pre-requisite patch will fire as an initial step, with its' own log file <release>.init.log.<sid> and optional error file <release>.init.err.<sid>

Q - Quit

Close the menu.

Z - Create Logfiles Zip

Creation of a zip file with all installation log files present at that time, for the installation of the current release on the current database.

The zipfile is created in the $OZG_PATCH/<release> directory in the format <release>.log.<sid>.zip, e.g. 10.14.2.2.0.log.prod.zip.

$ - OS Command Line

Starting an OS shell.

By leaving the shell, with an "`exit`" or "`^D`", you will return to the installation menu.

@ - SQL*Plus (<owner>)

Starting SQL*Plus under the Oracle Health Insurance owner account.

Leaving SQL*Plus, with a "`quit`" or an "`exit`", you will return to the installation menu.

? - Help

This submenu shows online help information regarding use of symbols, menu control, menu options and universal OHIPATCH options.

The version number of the used OHIPATCH version will also be shown.

License Check

An Oracle Health Insurance (patch) release contains all necessary installation objects for the different product lines (core + options).

When a certain installation activity does not apply based on the license present, this activity will not be shown.

When an activity does apply, but not all sources within this activity do, then only the sources for which a license is present will be installed.

First Use on an Environment

The OHIPATCH installation menu has to be used for installations in an OHI Back Office or OHI Data Marts environment.

Initially, the question "`Is this an OHI Data Marts environment?`" has to be answered with `Y` or `N`. When the answer is `N` the question "`Is this an OHI Self Service environment?`"has to be answered with Y or N (this product was formerly supported and the installation menu still needs to be adapted for this).

When the answer on both questions is N, the environment will be considered an OHI Back Office environment.

The menu will automatically ensure that non-relevant menu options are not shown. Also, actions that do not apply (e.g. checks on the running batch scheduler) will not be executed.

Note: During installation, checks are performed to validate the availability of an OHI BI license and the provided answers to the previous questions.

If things do not match, this will be signaled and you will be requested to restart the menu and re-answer the previous questions.

CONFIGURATION FILE

`OHIPATCH` uses the configuration file `$OZG_BASE/conf/ohipatch.conf` in the background. For this file also applies that other accounts (besides the installation operating system account) do not have access rights (see section *File authorization*).

This file is automatically created when starting the installation menu for the first time.

PARAMETERS

The table below indicates which parameters are present in the configuration file and what their content/function is.

NOTE: any defaults mentioned are valid for new environments only. Once the file `$OZG_BASE/conf/ohipatch.conf` has been created for an environment, the default that is valid at that time is written to the file. Update the value in the file to change the default value for existing environments.

Parameter	Description
`alert_cmd`	Optional – specifies filename of executable in $OZG_ADMIN which is called from OHIPATCH on start, finish and failure of Automode installation. This is documented in a later Chapter.
`ozg_begin_ree`	The (patch) release chosen to start the most recent installation session.
`ozg_eind_ree`	The (patch) release chosen to end the most recent installation session.
`ozg_eul`	Indication Oracle Discoverer End User Layer: • **N** Indicates that no Discoverer End User Layer is used; even though it is possibly present in the (patch) release, it will not be installed. • **Y** Indicates that a Discoverer End User Layer is being used; when it is possibly present in the (patch) release, an option is shown which can confirm that it has been installed.

Parameter	Description
ozg_max_comp_processes	The maximum number of processes used during Forms compilation. Defaults to 1.
ozg_obd	Indication OHI Data Marts: • **OBD** Indicates that this ($TWO_TASK) environment is an OHI Data Marts environment. • **N** Indicates that this ($TWO_TASK) environment is an OHI Back Office environment and that no OHI Data Marts license is available; hence OHI Data Marts interface software will not be installed on this OHI Back Office environment. • **Y** Indicates that this ($TWO_TASK) environment is an OHI Back Office environment and that that an OHI Data Marts license is available; hence OHI Data Marts interface software will be installed on this OHI Back Office environment. The values Y/N are automatically derived from the available licenses.
ozg_prd	Name of the product for the environment. Allowed values: • **OSI** was used to indicate the environment was an OHI Self Service environment. • **OBD** indicates the environment is an OHI Data Marts environment. • **OZG** indicates the environment is an OHI Back Office environment.
ozg_suppress_warnings	During step 110 warnings can occur. Default value is **N**, in that case the installation is paused and the user is asked whether to continue or not. If set to **Y**, installation will continue without pausing.
ozg_run_dmenu_in_automode	• **N** Report mandatory D menu activities at the end of Automode installation • **Y** Automatically execute mandatory D menu activities at the end of Automode installation
ozg_force_client_comp_in_automode	• **N** Only perform Forms compilation (activitity 800) when this is a mandatory activity (i.e. when a library was updated) • **Y** Force activity 800 to be performed at the end of Automode installation and ozg_run_dmenu_in_automode=Y
ozg_820_hours_in_automode	Number of hours option 820 is allowed to run if this option is executed in Automode. Default is 0 hours for new installations.

SILENT INSTALLATION

It is possible to run the `OHIPATCH` installation menu in *silent* mode, which is without any manual intervention.

This enables the running of an installation session as called by a script, scheduling an installation using Unix crontab, etc.

To use silent installation, start the menu with the `-silent` tag, e.g.

$OZG_BASE/utils/OHIPATCH.pl –silent prd

When configured correctly, the installation will start immediately and fully automatic.

Automode

Internally, silent installation makes full use of the Automode feature; therefore all conditions that apply to Automode, apply to silent installation (e.g. perform manual activities before starting the automatic installation!).

Silent installation will also, like Automode, stop automatically when it runs into any error.

In silent mode it is advised to set the `ozg_run_dmenu_in_automode` parameter to `Y` to fully automate the installation process.

The difference between silent installation and Automode is that Automode is manually enabled (at any time) when an administrator uses the installation menu in interactive mode, while silent installation is a fully automated process from start to end (which enables automated scheduling & processing of (patch)release installations).

CONDITIONS

Parameters

The `ozg_begin_ree` and `ozg_eind_ree` parameters in the configuration file *have to be specified* for silent mode; the installation menu uses the values of these parameters to determine at which (patch)release to start and end the automatic installation.

5. ERROR HANDLING

GENERAL

SUCCESSFUL INSTALLATION

When the installation log files no longer contain errors, the installation of the (patch) release has been performed successfully.

ACTIONS BASED ON ERRORS

If errors occur during a release installation, a number of steps have to be taken, depending on the location of the error.

Always ensure that a backup can be restored, for the database as well as the file system.

First determine if the error can be resolved by you. (e.g. in case of lack of disk space).

Unless stated otherwise, all (patch) releases can be restarted; once the cause of the error has been resolved, the (patch) release will have to be installed again.

If an error cannot be resolved, first determine if it is a known installation problem. Check the document "OHI BO and DM - known installation issues.docx" and the announcements on Sharepoint.

When it is not a known installation problem please contact Oracle Health Insurance Support. They can determine what the consequences are and if the release installation can be continued. It may be possible to resolve the issue over the phone. If not, an incident may have to be reported which will be dealt with according to its urgency.

When an error involves an `ORA-600` (i.e. `ORA-006XX`) or `ORA-07445` message, this error *almost always* has to be reported with a Service Request for the database product to Oracle Support too.

RESTORE DELIVERIES

When errors that occur during a (patch) release installation have to be resolved by Oracle, this will be dealt with by:

- A fix in an earlier (patch) release.
  When the error can be prevented by delivering a patch on an earlier release this may be used as a solution to make sure the installation of the later release can continue without errors.
- A fix in a later (patch) release.
  This later patch release may not only contain a fix for the error but may also contain other parts of the installation which have not been installed as a result of the errors. If acceptable (judged by Oracle) the error may be ignored until that later fix.

A fix by redelivery of the (patch) release.
When a (patch) release cannot be installed as a result of a blocking error (installation will not continue), the complete patch (release) will be redelivered.
The patch (release) will then be redelivered to MOS Patches & Updates.
Reinstallation only has to be executed by customers who had errors. When no errors occurred during the installation of the original release, the new delivery does not have

to be reinstalled. A explanation will be placed on MOS Patches & Updates for each redelivery, for the (patch) release concerned, stating ‘Known installation problems’. The documents “OHI BO and DM - known installation issues.docx” and the announcements on Sharepoint will be updated.

EXAMPLE

Suppose in the 10.20.2.0.0 release, a SQL script is delivered and the customer encounters performance problems during installation such that the installation cannot be completed within reasonable limits.

Oracle will fix the SQL script; the modified script will be delivered in a subsequent patch. E.g. in 10.20.2.0.1.

Apart from the fix in patch 10.20.2.0.1 the patch/release which contained the original script may also be redelivered at the customer’s requests.

In that case, a new patch/release (10.20.2.0.0) will be made available, stating it is a solution for ‘Known installation problems’. Because such a redeliveries can be confusing, this will only be done in exceptional circumstances.

It is not mandatory for customers to install this redelivered patch. They may also use the subsequent patch (10.20.2.0.1).

Customers experiencing problems are offered a viable way to overcome the problem.

Customers who do not experience problems do not have to re-install an earlier patch/release.

Note: When, after the original 10.20.2.0.0, the 10.20.2.0.1has already been installed and the decision is made to install the adapted 10.20.2.0.0, then all patches delivered after that patch concerned have to be reinstalled, too. So, after the restored 10.20.2.0.0 the 10.20.2.0.1 has to be reinstalled, too.

Preconditions

The above construction is ONLY used for SQL scripts causing insurmountable performance problems. When a script for instance runs longer than 8 hours, without any indications of completion of the script, a request may be made for the redelivery of a (patch) release or a fix in an earlier release that can prevent the script to run that long (whatever is viable). When a script contains an error which is corrected in a later patch (when there is no performance issue) NO redelivery will take place. Only when a situation occurs which cannot be resolved by a delivery in a later patch, will this mechanism be used for errors.

In relation to the times stated in the preconditions, it may differ per script what does and what does not take a long time. The customer has to clearly mention why a patch redelivery request is justified. Based on this, Oracle will decide if the request will be honored.

6. SPECIFIC INSTALLATION INSTRUCTIONS

PREPARATION & COMPLETION

Before a release is installed in an environment, a number of activities can and has to be performed to prepare the environment.

A good preparation can reduce the actual installation time (as it can 'keep running' without interruptions).

Once a release has been installed, a number of things have to be done *before the environment can be released to the end users*.

PREPARATION

Following activities can be performed in preparation of the release installation.

IMPORTANT: It is not required to wait with these activities until the actual installation of the (patch)release; the mentioned activities can all be performed (long) before.

1. Ask Functional Specialists to provide the values for parameters in the `readme.<PRODUCT>` files.

   These values can be entered in activity 10 in the installation menu.

2. Perform the installation instructions.

   This is activity 20 in the installation menu.

3. Run the check scripts twice: 1x well before the release is installed (so that there is ample time to solve any issues and so that this does not affect actual installation time), and 1x right before the release installation to determine if no data have been added which violates the checks.

   This is activity 100 in the installation menu.

   Note that during the database installation (activity 110 in the installation menu) the check scripts are rerun automatically; when these result in errors, the database installation is stopped.

4. Perform **Object Validation** to determine if the environment is initially correct and fix reported problems before installation.

   This is activity 900 in the installation menu.

   Start the **Object Validation** *via the installation menu* to check if all is set up correctly for *installation*.

COMPLETION

After installing (patch)releases, execute following activities:

1. Perform the activities from the installation submenu D (Completion).
2. Switch the database from installation mode to normal mode (e.g. by unsetting `parallel` settings).
3. Start, if applicable, the application to check that the environment can be accessed.

4. For OHI Back Office, check whether .ear files were delivered in one of the installed releases (you can check the list with sources for the installed release(s) or check whether in the folder where each release was unpacked whether the `java` folder contains relevant .ear files, meaning .ear files that were actually already deployed for the upgraded environment before the installation started).

   If .ear files were delivered and you have deployed one or more of these in some application server environment(s) make sure you redeploy them, using the latest version that was delivered. For deployment instructions, please use the documentation regarding installation of the 'web services' and/or the 'service layer' (you should be familiar with this as this has been used during the initial deployment).

   After the new .ear files are deployed, you can validate if the version of the .ear file as shown in weblogic, corresponds with the version that is mentioned in the Release installation instructions.
5. Start the **Object Validation** *via the application* to check if all is set up correctly for *runtime*.
6. Fix any remaining errors; if this is not possible, contact Oracle Health Insurance Helpdesk to decide if the errors are blocking (and the environment can thus not be delivered to the end users), or non-blocking.

INSTALLING AN INITIAL RELEASE

INTRODUCTION

An initial release is used for the creation of a new and empty Oracle Health Insurance environment. A special patch will be delivered on request containing all sources needed for an 'empty' Oracle Health Insurance environment. Empty means that only a series of system tables (e.g. system messages, module definitions, etc.) are populated.

Regular patch releases only contain the changes since a specific previous and are so called 'delta' releases instead of initial releases.

For the installation of an initial release different installation activities apply than for the installation of a regular release.

ACTIVITIES

Install the database and application server software, create a database and create the correct OHI directory structure on the file system. See the following document for more information:

Oracle Health Installation, Configuration and DBA Manual

Execute the steps below to create an empty Oracle Health Insurance environment (an 'empty' OHI database containing the complete data structure with only the initial seed data for starting the application):

1. Unzip the <release>.zip file in the usual way in `$OZG_PATCH`.
2. Change directory to the <release> directory.

3. Run the following scripts with SQL*Plus under the SYS account (make sure you connect to SYS providing the SYS password to connect to the pluggable database) to create generic objects, accounts, roles, directories, privileges, etc. First, set the Unix environment to the correct OHI database with

   ```
   oraset <env>
   ```

   Example:

   ```
   oraset PRD
   ```

   ☞ This assumes you have configured oraset and added a line in /etc/oraset for your environment.

   Next logon to SYS within the target pluggable database and run the appropriate scripts.

   ```
   @sql/OZGI001S.sql (for OHI Back Office)

   @sql/OBDI001S.sql (for OHI Data Marts)
   ```

4. Run the following script (in the $OZG_PATCH/<release> directory) to create the objects needed for the installation. Run this script with SQL*Plus under the Oracle Health Insurance owner account (e.g. OZG_OWNER for OHI Back Office or OBD_OWN for OHI Data Marts):

   ```
   @sql/OZGI004S.sql (for OHI Back Office)

   @sql/OBDI004S.sql (for OHI Data Marts)
   ```

   First, set the Unix environment to the correct Oracle Health Insurance database with

   ```
   oraset <env>
   ```

   Example:

   ```
   oraset PRD
   ```

   After this a list of plsql objects will remain invalid, you can ignore this.

   During the further installation additional missing objects will be created so these objects can become valid.

5. In case of an OHI Data Marts installation, create a private database link named SRC_OPENZORG to a valid OHI Back Office schema. Check the OHI Data Marts Administrator Reference for more information.
6. Copy from $OZG_PATCH/<release>/utils to $OZG_BASE/utils.
   - OHIPATCH.pl
   - OHIPLIB.pm
   - OHI_CMD.pl
   - OHILOGSCAN.pl
7. Copy from $OZG_PATCH/<release>/sql/ to $OZG_BASE/sql.
   - OZGC003S.sql
   - OZGC007S.sql

- OZGCOMPILE.ins

Depending on the environment copy the files:

- OHI Back Office
  - OZGSYNONYMS.ins
  - OZGGRANTS.ins
- OHI Data Marts
  - OZGC008S.sql
  - OBDSYNONYMS.ins
  - OBDGRANTS.ins
  - OBDCOMPILE.ins

8. In case of an OHI Back Office installation copy from `$OZG_PATCH/<release>/sh/` to `$OZG_BASE/sh`.
   - OZG_PROC.sh
9. In case of an OHI Back Office installation copy from `$OZG_PATCH/<release>/utils` to `$OZG_BASE/utils` all files which have a name that starts with the prefix below (so use d12c2* to identify them all).
   - d12c2
10. Create an empty file `ozg.conf` in folder `$OZG_BASE`.
11. Start `OHIPATCH.pl` and perform the (patch) release installation.

    You will be asked what kind of environment you are installing. A default username and password will be suggested. Be aware that the suggested defaults are in uppercase. These will not be accepted when you use case sensitive usernames and passwords in the database and the owner account is in small case.

    Make sure you did install the application server software and did run at least the configuration wizard to create the `$ORACLE_INSTANCE` environment. This folder structure is needed to make the check on the Developer software version succeed.

    Run option U so the menu places relevant information in file $OZG_BASE/ozg.conf.

    During installation of the database objects the message will appear that the previous release has not been installed. This message can be ignored. For other messages, the usual procedure applies.

EARLIER INSTALLATION OF PATCH RELEASES

In specific cases, Oracle may indicate that it is necessary to install a patch release before other, previously released, patch releases. In this case, the following rules apply:

1. Early installation of a patch is only permitted when written permission has been given by Oracle. This permission is sent via e-mail and is registered by Oracle.
2. The permission only applies to the patch indicated and not to any subsequent patches.
3. The re-creation of a patch at the customer's request (customer-specific delivery of one or more incidents) is not supported.

7. Performing Installation of Multiple (Patch) Releases

Using the installation menu `OHIPATCH`, it is possible to install a range of (patch) releases sequentially.

Methods

When installing a range of (patch) releases, 2 methods can be used:

1 - Automode

When running in Automode, all options will be performed automatically.
Automode will stop when an error occurs.

The main advantage of this mode is that no manual actions are necessary (only in case of errors or manual activities, see Automode), so this is the easiest way of installing (patch) releases.

2 - Manual Installation

The main advantage of this option is that during a multiple (patch)release installation a few steps can be skipped for all but one of the releases.
This means manual installation for a large number of large (patch) releases can save installation time.

The installation procedure in case of manual installation should be:

1. For every (patch) release in the range:
2. Perform all installation activities in submenu's A and B.
3. For submenu C, *only* use option C1
   This will prevent intermediate compilation/generation of executables.
4. At the end of the complete (range) installation, perform all activities in the D submenu.
   In submenu D, activities 800 and 810 will then take care of the compilation/generation of all executables.

Errors During Installation

When installing a *range* of (patch) releases, it is important to understand all errors (that may occur *during* the installation process) should be gone *at the end* of the complete (range) installation.

For example, when installing patches 1 to 10, an error may occur in patch 3. However, if this error is fixed in patch 8, the error will *automatically be fixed during the installation of the range* of (patch) releases.

It is therefore *not necessary and not required* to contact the Oracle Health Insurance Helpdesk for these *intermediate* errors; only errors that (still) exist at the end of the complete (range) installation must be reported (if they cannot be solved by the DBA; see Error handling).

Please see the Appendix ‘ Suppressing known installation errors’ for a method to suppress these *intermediate* known errors such that automatic installation can install the complete range without being cancelled for each (known) error.

8. CONFIGURING AN ALERT FOR INSTALLATION ERRORS

Installing (a range of) patches can be time consuming. Using automatic installation (automode) alleviates the work for the DBA as he/she does not have to manually step through each activity of every patch. However when (unexpected) errors occur during installation, the DBA might want to know this as soon as possible to take appropriate action. OHIPATCH supports an external alert command which is called at specific situations. This alert command can be used to notify stakeholders about the progress of an ongoing release installation.
This chapter describes the supported method to implement this.

EXTERNAL ALERT COMMAND

In the OHIPATCH configuration file ohipatch.conf., the parameter `alert_cmd` can be set to the file name of an executable program. This command needs to be present in the `$OZG_ADMIN` directory and is invoked from OHIPATCH when automatic (automode) installation is used.

The command needs to be executable by the same operating system user as is used for OHIPATCH, usually this is the ‘oracle’ user.

It should handle 4 command line arguments, in the following order:

1. the **type** of the alert {‘I’,’E’} - for **I**nformational or **E**rror messages
2. the **environment** being patched, e.g. 'ACC01'
3. the **release** being installed, e.g. '10.20.2.0.23'
4. the **message**, e.g. 'Error in activity 110', or 'Automatic installation started'

Example of a call made from OHIPATCH for `alert_cmd=alert.pl`:

```
$OZG_ADMIN/alert.pl 'E' 'ACC01' '10.20.2.0.23' 'Error in activity 110'
```

The alert is activated when the following conditions are met:

- ozg.conf.<sid> parameter `alert_cmd` is set
  AND
- the specified command is present in `$OZG_ADMIN` and is executable (this is checked at startup of OHIPATCH)
  AND
- automode (including silent) installation is activated when running OHIPATCH

ALERT TYPES

The supplied alert type is intended for use inside the custom alert program to determine whether to actually trigger a notification or not.
For example, only send an SMS to the DBA when the type is ‘E’.

The following table shows the type of alerts and the conditions under which these alerts are sent.

Type	Condition	Message
I	Automode installation is started.	'Automatic installation started'
I	Manual input required.	'Manual input needed to process activity *X*'
I	Automode installation finished.	'Installation finished'
E	One or more errors occurred in an activity, automatic installation is cancelled.	'Errors in activity *X*'

EXAMPLE ALERT PROGRAM

In `$OZG_BASE/conf/configuration-templates/generic` an example script is available: `alert_example.pl`.

In this script an annotated example (in Perl) is given. It sends an email containing information about the given arguments using the Linux 'sendmail' program.

9. WEBSERVICES AND THE PATCH PROCESS

Online patching for webservices is possible depending on the contents (sources) in an interimpatch.

ONLINE PATCHING

Online patching is implemented by putting service calls in wait for the duration of the critical part of the database installation that is performed in step 110 of the installation menu.

To achieve this the configuration options of the WebLogic Data Source Connection Pool are used (see next paragraph). After successfully finishing the relevant database installation part the webservice calls will be resumed.

Timeout settings configured in the web service calling environment should be greater than the time needed to perform this part of the installation to prevent errors. This also applies to any customer specific service calling indirectly one of the OHI Back Office webservices.

PATCH CLASSIFICATION

Installation step 110 will report if online installation is considered an option for that specific interim patch. This is determined based on the contents of the patch release. There are three distinct possibilities, one of which will be reported in the logfile:

1. Online patching is not possible because the contents delivered are critical to the functioning of the OHI Back Office webservices. The following informative message will be shown in the logfile:
   ```
   INFO : This patch requires OHI Back Office webservices to be
   OFFLINE.
   ```
2. Online patching is possible, but the time needed to install the patch must be evaluated on a customer specific representative environment. Patching may take too long and result in timeout errors being returned to the webservice client(s). Message shown in the logfile:
   ```
   INFO : The amount of time needed to install this patch should
   be evaluated.
          If configured, webservices will be halted until patching
   has finished.
          If installing this patch takes too long, timeouts may
   occur for webservice
          calls issued during installation of this patch.
   ```

   In this situation, the logfile will also report the timespan during which webservices will become unresponsive:
   ```
   INFO : Patching database objects took 00:00:25
   ```

   By running the patch release on an acceptance environment this information can be used to decide if an interim patch can be deployed online or should be deployed offline.
3. Online patching is possible because the contents are not in any way related to webservices:
   ```
   INFO : This patch does NOT require OHI Back Office webservices
   to be offline.
   ```

Installing a range of interim patches of the second category is not fundamentally different from installing a single one. Webservice calls will just experience multiple potential waiting periods instead of just one.

Patch releases from both the second and third category can be part of a range of interim patches to be deployed in automode and online.

RESTRICTED SESSION PRIVILEGE

During the patch process restricted mode is activated for the database. This means that current and new database sessions will be restricted. For the majority of users this makes sense as they may or will likely experience system errors during patching.

If patching is performed online, the database user(s) defined to execute webservice calls should receive the RESTRICTED SESSION privilege from the DBA to prevent these errors by issuing: `grant restricted session to <user>;`

WEBLOGIC DATASOURCE CONFIGURATION

To enable using a 'wait' in OHI Back Office webservice calls a configuration change must be made to the datasource(s) of OHI Back Office webservices defined in the Weblogic application server.

Use the WebLogic Remote Console. See the *Oracle Health Insurance*

Back Office Installation, Configuration and DBA Manual on oradocs.com for instructions on how to install the WebLogic Remote Console.

- Start the WebLogic Remote Console
- Connect to the Admin Server of the web services Domain.
- Select the "Edit Tree".
- Select "Services"
- Select "Data Sources"
- Select the data source you want to modify
- Open the "Connection Pool" tab and then the sub-tab "General"
- Set "Initial Capacity" to 0.
- Click on "Save"

This way, the check_patch routine is not executed until a web service call is actually executed, and no shared lock is acquired until then.

- Go to the sub-tab "Advanced".
- You need to specify several properties:
    - Test Connections On Reserve: `On`
    - Test Table Name:
      `SQL begin ALG_PUBLIC_PCK.check_patch(); end;`
    - Test Frequency: `0 seconds`
    - Seconds to Trust an Idle Pool Connection: `0 seconds`

This will result in a (small) overhead because for each call a shared lock will be requested on a specific (unique) lock handle.

10. COMMAND LINE MODE

In addition to the interactive mode to install OHI patches that is decribed in previous chapters, OHIPATCH can also be used in Command Line mode.

Command line mode is intended for use in automated deployment and patching scripts.

In Command Line mode, exactly one step of one patch can be executed without user interaction, by supplying command line parameters. The output that appears on the screen in interactive mode is also written to a log file in command line mode, and the result of the step can be tested by the caller after the command completes.

Command Line mode differs in several ways from the silent and automatic options of the interactive mode described above:

- Command Line mode supports more patch steps without user interaction, e.g. step 10 and step 20.
- Command Line mode executes only the specified step for the specified patch, so there is no need to define the range of patches in the ohipatch.conf file, and there is no Automode cancellation when an error occurs.

Command Line mode writes identical log files as the interactive mode, and interactive mode will show the result of steps executed in Command Line mode.

Command Line mode has the following limitations:

- No support for ranges of patches
- No support for automated restart of OHIPATCH if a patch places new installation software.
- No support for installing out-of-order patches.
- No automode. Each step has to be called explicitly.
- No check if all required steps are executed, or if they are executed in the right order.
- No support for wildcards in steps, e.g., for table names.
- No support for recompiling Forms and Libraries without stopping the batch scheduler.
- For new OHI environments, interactive mode needs to be run first, to answer questions about the environment. Answers will be stored in $OZG_BASE/conf/ohipatch.conf.
- Some steps are only applicable for certain releases/patches or products. In interactive mode the non-applicable steps are not shown in the menu. In Command line mode there is no support for testing if a step is applicable. Trying to execute a non-applicable step will result in a FATAL message.

  HINT: the steps that are applicable for a patch and product are listed in the file $OZG_PATCH/<patch>/doc/<patch>.prp. See the lines starting with “ACT#”.

Command Line mode is executed with the following command:

```
./OHIPATCH.pl -patch <patch number> -step <step> [-steppar1 <value1>] \
  [-steppar2 <value2>] [-steppar3 <value3>] \
  [-steppar4 <value4>] [-steppar5 <value5>] <environment>
```

Where

Parameter	Description
-patch <patch number>	number of the release or interim patch of the step to execute
-step <step number>	patch step to execute in command line mode
-steppar1 <string>	optional: first parameter for the step to execute
-steppar2 <string>	optional: second parameter for the step to execute
-steppar3 <string>	optional: third parameter for the step to execute
-steppar4 <string>	optional: fourth parameter for the step to execute
-steppar5 <string>	optional: fifth parameter for the step to execute
<environment>	mandatory: name of the OHI environment to patch

Note that the environment is the only positional parameter, to ensure compatibility with the interactive mode.

PARAMETERS

The table below lists the steps that are supported in Command line mode, and the parameters the steps require. If a parameter is defined, a value must be specified.

Step	Description	Parameters	Parameter Description
10	Fill parameter file	-steppar1	File action • P = Place This will create readme.<product>.<environment> based on the template file readme.<product> • R = Register This will register the step as completed
20	Process specific installation instructions	--	
30	Place installation modules	--	
90	Perform object validation	-steppar1	Location to check (D = Database, F = Filesystem, B = Both)
		BO only: -steppar2	Mode (R = Runtime or I = Installation)
100	Run SQL check scripts	--	
110	Run database installation script	-steppar1	Stop at errorlevel (W = Warning, E = Error, F = Fatal)
115	Run database installation script	--	
120	Process synonyms, grants and schema compilation	-steppar1	Display all grants issued (Y/N)?

C1	Complete filesystem installation WITHOUT compilation/generation	--	
C2	Complete filesystem installation WITH compilation/generation	--	
800	Compile/Generate objects in $OZG_BASE/bin	-steppar1	Prefix to generate specific objects in $OZG_BASE/bin ("*" (with the double quotes) for All)
		-steppar2	Extension to generate specific objects in $OZG_BASE/bin (Use "*" (with the double quotes) for All)
820	Gather table/index statistics	-steppar1	Maximum #hours to Gather table/index statistics?
830	Show Business Rules validation status	--	
840	Drop obsolete columns	--	
850	Partition/compress tables	BO only: -steppar1	F = Finalize or R = Rollback table redefinition
		DM only: -steppar1	Compression Mode A = Compress Advanced Q = Compress for Query (Exadata only) U = Uncompress (revert compression)
		-steppar2	Table name
855	Drop old/empty partitions	--	
860	Move tables/indexes	-steppar1	Table Name. Wildcards are not supported. Escape the $-sign with a backslash. Use "*" (with the double quotes) for All
		-steppar2	Object Type (T = Tables, I = Indexes, U = Unusable Indexes, B = Tables and Indexes)
		-steppar3	Max hours (integer). 0 means no restriction. Use 0 for Unusable Indexes
		-steppar4	Max number of parallel processes (integer). Maximum = 10. 1 means no parallel processing.
		-steppar5	Source Tablespace. Wildcards are not supported. ("*" (with the double quotes) for All)
870	Rebuild unusable indexes	--	
890	Process manual deployment instructions	--	

900	Perform object validation	-steppar1	Location to check (D = Database, F = Filesystem, B = Both)
		BO only: -steppar2	Mode (R = Runtime or I = Installation)
S	Reset log file(s)	--	
Z	Create logfiles zip	--	

Remarks:

- In step 10, action P = Place will create an initial version of the parameter file for the specific environment, based on the template file for the specific patch, if the file is not present yet. You can then fill in the appropriate parameter values. Action R = Register will then register that you completed step 10. Alternatively, you can (programmatically) place a file with the correct parameter values first, and only execute Action R = Register.
- Steps 200 – 530 cannot be executed individually. These steps are executed as part of steps C1 and C2.
- Step 880 is not supported.

Some examples of valid commands:

```
./OHIPATCH.pl -patch 10.21.4.0.0 -step  10 -steppar1 R OHIDEV01

./OHIPATCH.pl -patch 10.21.5.0.0 -step  30 OHIDEV03

./OHIPATCH.pl -patch 10.21.4.0.3 -step  90 -steppar1 B DMTEST4

./OHIPATCH.pl -patch 10.21.4.0.0 -step 850 -steppar1 F -steppar2 \
        FSA_IAF_CATEGORIEEN OHITST01
./OHIPATCH.pl -patch 10.21.4.0.0 -step 860 -steppar1 FSA_IAF_CATEGORIEEN \
   -steppar2 T -steppar3 0 -steppar4 4 -steppar5 OZG_DIM_SYS_TAB OHITST22
```

Technical information:

- Each call will return a result of 0 (success) or 1 (failure)
- Each call that passes an initial validation of the parameters is logged in file $OZG_PATCH/<patch>/<patch>.steps.log.<environment>. The output that is sent to the screen only in interactive mode is added to that file. The logfile for the specific step itself is still leading. For example:

```
INFO : ================================================================================================
INFO : Mon_03-May-2021_14:55:53 STEP 10 PATCH 10.21.5.0.0
INFO : Preparing installation on BATEA16 ...
FATAL: Required parameter steppar1 for step 10 is missing. Description : File action (P =
Place, R = Register)
INFO : ================================================================================================
INFO : Mon_03-May-2021_14:56:20 STEP 10 PATCH 10.21..0.0
INFO :  -steppar1 R
INFO : Preparing installation on BATEA16 ...
INFO : Check database connection using Wallet entry /@BATEA16_install ...
INFO : Connected to BATEA16 as OZG_OWNER
INFO : Determine last installed (patch)release ...
INFO : The last installed release on this environment is    : 10.21.5.0.0
INFO : Installation of release 10.21.5.0.0
INFO : Determine installation options using present licenses ...
```

```
INFO : Check version of OHIPATCH.pl ...
INFO : Current   version of OHIPATCH.pl = 4.138
INFO : Delivered version of OHIPATCH.pl = 4.138
INFO : $OZG_BASE/utils/OHIPATCH.pl is up-to-date and therefore not updated
INFO : Check the Developer version + utilities ...
END  : Activity "Fill the parameter file" is executed
INFO : End Oracle Health Insurance installation menu

INFO : ================================================================================
INFO : Mon_03-May-2021_14:58:44 STEP 20 PATCH 10.21.5.0.0
INFO :  -steppar1 R
INFO : Preparing installation on BATEA16 ...
FATAL: Parameter steppar1 for step 20 is not supported.
```

- The output that is sent to the screen in interactive mode can also be redirected by the calling program.

APPENDIX A PARTITIONING & COMPRESSION

INTRODUCTION

CONDITIONS

If Partitioning is used in an Oracle database, two conditions must be met:

1. A license for the *Oracle Partitioning Option* is required.
2. The Oracle Partitioning Option has to be installed.

When the Partitioning Option is installed but *is not used*, it can be **deinstalled** using the *Oracle Universal Installer*. When the Partitioning Option is not installed, only the **departitioning** functionality of this menu choice is available.
As long as the Partitioning Option is active but not used, a warning will appear on installation of Oracle Health Insurance (patch) releases.

If Compression is used for OHI Back Office, the license for Oracle Advanced Compression is required. Advanced Compression is the only type of compression supported for OHI Back Office. Tables and indexes can be compressed.

For OHI Data Marts, indexes will NOT be compressed. Hybrid Columnar Compression (HCC) Query Low is used for table compression on Engineered Systems. The license for this compression type comes with the platform.

For other platforms (not Engineered Systems), Advanced Compression is used for the OHI Data Marts tables. This requires a separate license.

OHI DATA MARTS

Partitioning is required for OHI Data Marts and cannot be deactivated. Compression is optional and can be activated per table on demand. The main goal is reducing storage requirements, but the benefit may also be reduced I/O at the cost of a slight increase in CPU usage may be observed. Only the partitioned tables (a subset of the Fact tables) can be compressed.

OHI BACK OFFICE

For OHI Back Office the use of Partitioning is supported. Partitioning is *optional*; the customer has the choice if partitioning is to be used or not.

Partitioning is intended to increase ‘Scalability’. OHI Back Office uses hash partitioning to reduce concurrency when multiple processes need to access the same set of data. Without hash partitioning, the processes involved in parallel execution in many OHI Back Office batches would try to access and lock the same data blocks (e.g. the latest set of claim lines). Hash partitioning stores this data in multiple partitions so the processes will access different data blocks.

For OHI Back Office the use of compression is also optional. Compression can be activated per table on demand. When activated, the indexes on the table will also be compressed. The main goal of using these forms of compression is to reduce the storage footprint of the OHI Back Office database at the cost of a slight increase in CPU usage. Given the regular overall load of the OHI Back Office application the experience is that the additional CPU load is hardly noticeable, compared to the normal CPU load.

FUNCTIONALITY

OHI DATA MARTS

For OHI Data Marts the tables are created as partitioned tables in the designated table spaces. The tables can be compressed interactively on demand. Typically, the OHI Data Marts database is not available for use when compression or decompression activities are running.

See for more information:

Oracle Health Insurance - OHI Data Marts Administrator Reference

OHI BACK OFFICE

For OHI Back Office the tables are *created* as non-partitioned tables, but they can be *partitioned* at any time afterwards. The same holds for compression.

To reduce downtime for the OHI Back Office application a two-phase approach is offered for changing the partition and/or compression characteristic of an OHI Back Office table and its indexes:

1) Preparation phase – can be done 'online'
   During this phase a copy is created of the table and its indexes in the required structure using the 'online redefinition' functionality as provided by the database. This phase is done using batch process 'Modify table structure' (SYS1155S) to prepare the table through an implementation of the online redefinition functionality. Meanwhile the application can be used and changes to the table data remain possible without any problem. Changes to the data in the original table are applied to the copy table by a background job.
2) Finalizing phase – requires 'downtime'
   Using menu step 850 a table that has successfully finished its preparation phase can be converted to finalize the process. The contents of the copied table will be synchronized one last time and the copied table will become the new definitive table while the previous table structure will be dropped. As the bulk of the work is done during the preparation phase this finalize phase normally will take only minutes, strongly reducing the downtime required to implement partition/compression for a table.

Package ALG_STORAGE_PCK can be used to mark a table for (de)compression and/or (de)partitioning. Batch SYS1155S will detect the mark and create a copy of the table with the new structure using redefinition. As mentioned above, this copy action can be performed while the environment is online.

For this goal four procedures are provided by the package to specify your action:

```
procedure enable_compression( pi_naam in alg_tabellen.naam%type );

procedure disable_compression( pi_naam in alg_tabellen.naam%type );

procedure enable_partitioning( pi_naam in alg_tabellen.naam%type );

procedure disable_partitioning( pi_naam in alg_tabellen.naam%type );
```

These procedures only accept actual table names, that is: not synonyms that point to tables with a different name.

Additional requirements:

- The table must belong to FACT tablespace (as specified in ALG_TABELLEN.TABLESPACE; if the actual table is not in the specified FACT tablespace, it will be moved to that specified FACT tablespace during the redefinition.)
- The table must have either a Primary or a Unique key without nullable columns.
- The table must not be an Index Organized table (IOT).

Only a limited set of tables supports table and/or index hash partitioning. Whether specifying the partitioning flag is useful for a table can be determined by querying some OHI Back Office configuration tables. The following should apply when you want to enable the partitioning functionality for a table:

- The record in table ALG_TABELLEN should have a not null value for column PARTITIONING_KOLOM;

If you specify the partitioning flag for a table that cannot be partitioned, you will receive an error message.

Beware: if you do not specify a table name as a parameter for SYS1155S, the batch run will process all tables which are marked to receive a different storage structure. SYS1155S will create a copy for all these tables and thus require additional storage space until the redefinition is finalized or rolled back. So, although it might be tempting to activate compression for a large set of tables at once, this might result in the need for a large amount of temporary additional storage, which may be hard to regain after the copies have been dropped during the finalization/rollback phase.

Using the installation menu activity 850, the new table structure can replace the current/previous one. Because the new structure is already present and has all the data, the downtime needed for this step is relatively minimal and depends on the volume of changes that have occurred since the SYS1155S run and subsequent scheduled jobs to update the data in the new table structure.

When step 850 is executed, the database must be in ‘OHI installation mode’ (i.e., restricted mode, no end users, parallel servers enabled, specific parameter settings, etc.) to optimally use resources during the downtime, thus limiting the time needed for these operations.

Of course, you should always be prepared - as for regular installation activities - to revert to a valid database state in case something fails irrecoverably during these types of maintenance operations.

Note that the objects, once partitioned and/or compressed, can also be un*partitioned* and/or de*compressed* using the same process using the batch SYS1155S and option 850.

APPENDIX B INSTALLATION CHECKLIST

Following is a *quick reference checklist* for installing (a range of) OHI (patch)releases.

Most steps are hyperlinks to the relevant paragraphs in this document.

1 - GET THE RELEASE(S)

Decide which (patch)release(s) to install.

Download the releases.

Unzip the files in $OZG_PATCH directory on the application server.

2 - PREPARE FOR INSTALLATION

Make sure non-validated Business Rules are validated.

Make sure statistics have been gathered.

Make a backup.

Read *Known Bugs* list(s) for the release(s) on My Oracle Support.

Get functional input to fill the parameter file for the release(s).

Read *Specific installation instructions* for the release(s).

Configure the database & application server for installation.

Fill in requirements for major release installation, including Parallel Execution.

Start the installation menu.

Perform *Object Validation* in installation mode.

3 - PERFORM THE INSTALLATION(S)

Perform installation of (all) the release(s).

Use Automode or Silent Mode if required.

Fix errors & rerun installation(s) if necessary.

4 - COMPLETE THE INSTALLATION(S):

Reset the database & application server to runtime mode.

Gather statistics.

Make sure non-validated Business Rules are validated.

Perform *Object Validation* in runtime mode.

Address *all* remaining errors.

Check whether .ear files were delivered which are in use for your environment. If so, redeploy them.

APPENDIX C SUPPRESSING KNOWN INSTALLATION ERRORS

During installation, errors can occur. Normally an error must be resolved before continuing to the next installation activity. In some cases, however, especially when installing a range of patches, it might be useful to suppress 'known errors' such that automatic installation can continue installing subsequent patches.

By 'known errors' we mean errors occurred earlier on other pre-production environments which are either reported as 'Known installation issue' on My Oracle Support, or so-called *intermediate* errors which are solved eventually by installing subsequent patches.

Please note: this functionality should be used with great care. The errors should be thoroughly reviewed before adding them to the error suppression file. The functionality offers nice flexibility but brings great responsibility in using it.

This appendix describes how specific errors can be suppressed.

1. CONCEPTUAL OVERVIEW

In Figure 1 the process flow of the error handling after processing an installation activity is depicted.

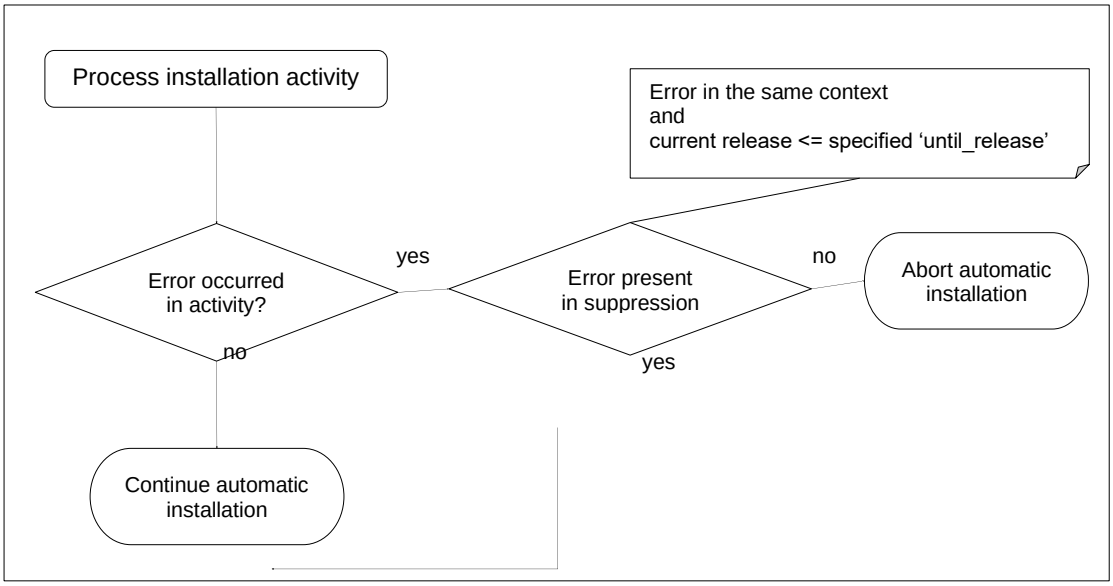

Figure 1: process flow of error handling

When errors are found in the activity's log file, normally an error is raised in OHIPATCH and automatic installation will be cancelled.
When an error suppression file is present, OHIPATCH will try to match the errors in the log file with the errors in de suppression file. Not only the error line must match, also the error context and the release until which the error should be suppressed are taken into account.

When all errors found in the activity's log file are matched in the suppression file the release installation will continue as if no errors occurred.

Starting with release 10.24.1.0.0, the suppression of an error will be logged in the <release>.>.<activity>.err.<sid> file, by adding a line:

```
INFO : The error above is suppressed in
<$OZG_BASE>/suppressed_installation_errors.txt
```

If you copy this line into `suppressed_installation_errors.txt` (e.g., by using `cat`, see below) it will be ignored when comparing with the actual error text in the next run.

2. GATHER ERRORS TO SUPPRESS

In order to suppress errors, they must first be encountered to retrieve the literal text and the context in which they occur.

During release installation all errors found in the log file of an activity are extracted and written to a corresponding file named <release>.<activity>.err.<sid> (e.g. '10.14.2.1.0.110.err.ohiacc1'). The error lines are grouped by their context. This context, marked with 'START:', can be an installed object (i.e. a specific table, module, script) or a more general activity (e.g. compile all invalid objects).
The context is needed to prevent errors from being incorrectly suppressed when they occur in a different context (e.g. a compilation error in Form X should be treated as a different error when the textually identical compilation error occurs in Form Y).

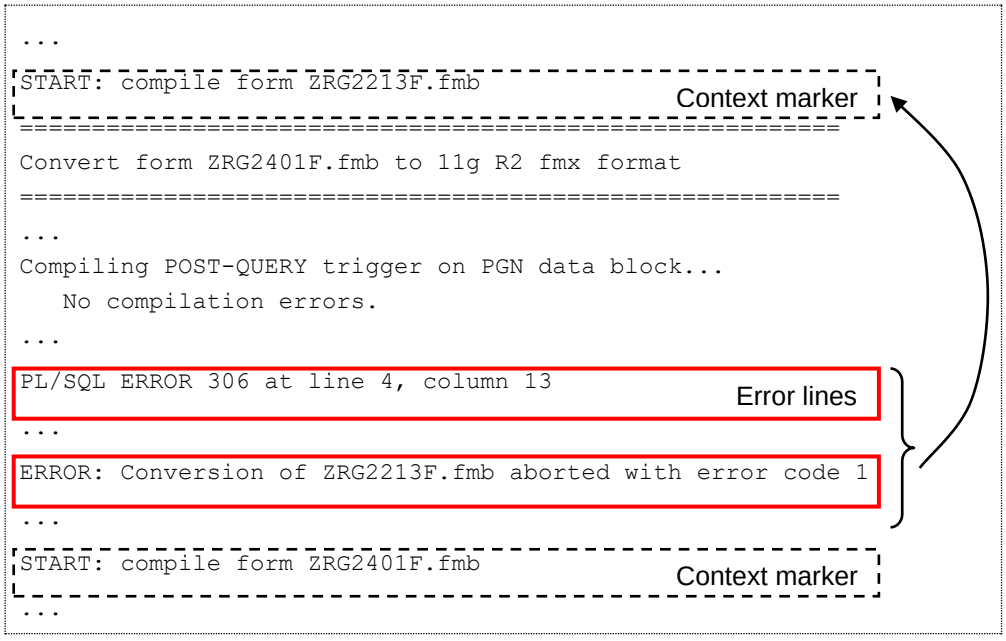

Figure 2: error lines in an activity's log file are captured by context

The annotated fragments of a log file (for example 10.20.2.0.1.800.log.ohiacc1) in Figure 2 result in the following output in the 10.20.2.0.1.800.*err*.ohiacc1 error file:

```
START: compile form ZRG2213F.fmb
@until_release=10.20.2.0.1
PL/SQL ERROR 306 at line 4, column 13
ERROR: Conversion of ZRG2213F.fmb aborted with error code 1
```

The next section describes how this error can be suppressed.

3. PLACE ERROR SUPPRESSION FILE IN $OZG_BASE

In $OZG_BASE a file can be created named ‘suppressed_installation_errors.txt’. When this file is present, all errors encountered during release installation are checked for a match in this file. The errors can be copied from the corresponding .err files in the patch directories.

The structure of the file is as follows:

```
START: {Context of the error, e.g. '*** BEGIN COMPILE ***'}
@until_release={release until the errors should be suppressed}
{list of error messages within the context}
```

These error messages are literally equal to how they appear in the log file. Example:

```
ERROR: In database (instance ID 1) non-RESTRICTED sessions are possible at the
       moment.
ERROR: Orace Health Insurance user MANAGER is still active (instance ID 1).
```

Example:

```
START: compile form ZRG2213F.fmb
@until_release=10.20.2.0.1
PL/SQL ERROR 306 at line 4, column 13
ERROR: Conversion of ZRG2213F.fmb aborted with error code 1
```

The file can contain multiple blocks consisting out of ‘START: ..’, ‘@until_release=..’ and one or more error lines. These should be copied from the .err. file(s) created during earlier runs.

When the exact error matches and the current release being installed is lower than or equal to the specified “@until_release”, the error is suppressed. This means the error will not result in aborting the automode for automatic release installation and the activity appears in the menu to have been installed without any error.

Suppressing an error which repeatedly occurs in a range

When a certain error occurs during multiple releases, the error does not have to be copied multiple times for each release to the error suppressing file.
Placing the error including context of the highest release is sufficient.

Fictitious example:

```
START: Activity 900
@until_release=10.13.3.4.1
ERROR: TABLE - Unexpected existence
       ALG#INSTALL_OPTIONS

START: Activity 900
@until_release=10.13.3.4.2
ERROR: TABLE - Unexpected existence
       ALG#INSTALL_OPTIONS
ERROR: TABLE - Unexpected existence
       ALG_DUMMY_TABLE

START: Activity 900
@until_release=10.13.3.4.3
ERROR: TABLE - Unexpected existence
       ALG#INSTALL_OPTIONS
```

Is functionally equivalent to:

```
START: Activity 900
@until_release=10.13.3.4.2
ERROR: TABLE - Unexpected existence
       ALG_DUMMY_TABLE

START: Activity 900
@until_release=10.13.3.4.3
ERROR: TABLE - Unexpected existence
       ALG#INSTALL_OPTIONS
```

Because the error about the ALG#INSTALL_OPTIONS is the same for all releases until '10.13.3.4.3'

Creating a single suppression file for all errors in $OZG_PATCH

To quickly create the `suppressed_installation_errors.txt` file containing all errors that occurred during the patch installations present in the $OZG_PATCH directory, the following Linux command could be helpful:

```
$ cd $OZG_PATCH
$ cat */*.err.ohiacc1 > suppressed_installation_errors.txt
```

Here 'ohiacc1' is an example SID of the environment.
Then this file should be inspected, optionally edited and placed into $OZG_BASE of the target environment.

Important: keep the line endings in UNIX-style, be careful when using Windows text editors. The errors otherwise are not matched correctly.

Suppressing multiple errors in the same activity

If you want to suppress multiple errors in the same activity, you can just copy them from an error file (e.g. with `cat` as described above), but then they are all under the same heading. The effect is that the whole set of errors needs to match the set of

errors that occur next time. If a new error occurs in addition to the previous errors, one error does not occur, or even the order changes, there will no longer be a match and none of the errors will be suppressed. To prevent this, you need to "split" the set of errors and give each error its' own "heading".

An example:

If you include this in 'suppressed_installation_errors.txt':

```
START: Activity 900
@until_release=10.13.3.4.2
ERROR: TABLE - Unexpected existence
       ALG#INSTALL_OPTIONS
ERROR: TABLE - Unexpected existence
       ALG_DUMMY_TABLE
```

No errors will be suppressed if only the following error occurs:

```
START: Activity 900
@until_release=10.13.3.4.2
ERROR: TABLE - Unexpected existence
       ALG#INSTALL_OPTIONS
```

It will be suppressed if you add this in 'suppressed_installation_errors.txt' instead:

```
START: Activity 900
@until_release=10.13.3.4.2
ERROR: TABLE - Unexpected existence
       ALG#INSTALL_OPTIONS

START: Activity 900
@until_release=10.13.3.4.2
ERROR: TABLE - Unexpected existence
       ALG_DUMMY_TABLE
```

Suppressing errors about out of order installation

- NOTE: you are not allowed to install interim patches "out of order" without requesting a patch advice and receiving written permission from OHI Support.

It is possible to suppress errors related to out of order installation of interim patches – after you have received permission – so you can install a range of patches in "silent" mode. This works in the same way as described above, by adding the errors in `suppressed_installation_errors.txt`, but you need to be aware that the check on out of order installation is executed twice:

- As part of the initial checks when starting OHIPATCH. The error is logged in <release>.>.init.err.<sid>
- As part of the initial checks of step 110. The error is logged in <release>.>.110.err.<sid>

Therefore, you need to add the errors from both files in `suppressed_installation_errors.txt`.

Example:

```
START: Activity init
@until_release=10.23.8.0.5
ERROR: Previous release (10.23.8.0.4) has NOT been installed. You need
       PERMISSION to continue installation.

START: Perform several checks
@until_release=10.23.8.0.5
ERROR: Previous release (10.23.8.0.4) has NOT been installed. You need
       PERMISSION to continue installation.
```

APPENDIX D DATABASE CHECKS

Before the actual database installation in step 110 of the patch menu is performed, the database installation is checked on many recommended and mandatory parameter settings. During Automode installation of more than one release these checks are only performed during the execution of step 110 for the first release.

This appendix gives an overview of all checks performed, one generic checkup and one specific checkup per product (OHI Back Office or OHI Data Marts). Combine both parts per product to have the complete checkup.

Checks are performed with output written to console and log file. Each check reports its severity either as FATAL, ERROR, WARNING or INFO.

GENERIC OHI CHECKS

FATAL	Installed release must be higher than current present release
FATAL	Installation cannot be performed as user SYS or SYSTEM
FATAL	Oracle RDBMS Enterprise Edition 19.0.0.0.0 is required
FATAL	Oracle Component XDB version 19.0.0.0.0 must be installed and VALID
FATAL	During non-Major installations in ARCHIVELOG mode FORCE_LOGGING=NO is not allowed
FATAL	The Default Tablespace of the schema owner must exist and be available
FATAL	The Temporary Tablespace of the schema owner must be equal to the DEFAULT_TEMP_TABLESPACE of the database
FATAL	Database system statistics must have been collected
FATAL	All tablespaces must be locally managed
FATAL	The SYSTEM tablespace must use extent allocation type SYSTEM or USER
FATAL	In RAC; Partitioning option must be consistent across all instances
ERROR	No PLAN_TABLE table may exist for the schema owner
ERROR	All data dictionary statistics must have been created / gathered
ERROR	All fixed objects statistics must have been created / gathered
ERROR	All mandatory Object Privilege must be present; see “Mandatory Privilege” overview
ERROR	During installation, non-RESTRICTED sessions are not allowed
ERROR	End-user sessions are not allowed
ERROR	Revoke privilege <PRIVILEGE> on <OBJECT> from <USER> has failed because of non-compliant dependencies
WARNING	Reported when installed release is already present
WARNING	Reported when the required previous release has not been installed before
WARNING	Reported when objects are still invalid after (re)compilation
WARNING	Reported during Major installations in ARCHIVELOG mode; FORCE_LOGGING=NO is not recommended
WARNING	Reported when database character set is not set to the supported AL32UTF8 character set
INFO	Reported if ARCHIVELOG mode is used
INFO	Reported when FORCE LOGGING is set to YES
INFO	Reported all compressed objects, and recommendation to re-apply compression
INFO	Reported the number of SQL Profiles found / registered

INFO	Reported (first 20 by name) objects with missing dictionary statistics
	Database instance parameter checks:
FATAL	Any deprecated parameter must be defaulted (not set) on all instances
FATAL	Parameter _COMPLEX_VIEW_MERGING must be defaulted (not set) on all instances
FATAL	Parameter event may not be set on any instance
FATAL	Parameter JOB_QUEUE_PROCESSES must be >= 10 on all instances
FATAL	When ASMM is not used; parameter LARGE_POOL_SIZE must be >= 16MB on all instances
FATAL	Parameter NLS_SORT must be set to BINARY on all instances
FATAL	Parameter OPEN_CURSORS must be >= 500 on all instances
FATAL	Parameter PROCESSES must be >= 200 on all instances
FATAL	Parameter SESSION_CACHED_CURSORS must be >= 500 on all instances
FATAL	Parameter STATISTICS_LEVEL must be set to TYPICAL on all instances
FATAL	Parameter UNDO_MANAGEMENT must be set to AUTO on all instances
ERROR	Parameter NLS_LENGTH_SEMANTICS needs to be set to BYTE
ERROR	Parameter PGA_AGGREGATE_TARGET must be > 0 on all instances
ERROR	Parameter WORKAREA_SIZE_POLICY must be set to AUTO on all instances
ERROR	Parameter OPTIMIZER_FEATURES_ENABLE must be '19.1' on all instances
WARNING	Reported when database parameter DB_CACHE_SIZE is not set to the recommended multiple of 16Mb on all instances
WARNING	When ASMM is not used; reported when database parameter SHARED_POOL_SIZE is not set to the recommended multiple of 16Mb on all instances
WARNING	Any other not prescribed optimizer parameter must not be set on all instances
WARNING	Parameter AWR_PDB_AUTOFLUSH_ENABLED must be set to TRUE
WARNING	Parameter OPTIMIZER_ADAPTIVE_PLANS must be set to TRUE.
WARNING	Parameter OPTIMIZER_ADAPTIVE_STATISTICS must be set to FALSE.
WARRNING	Parameter _OPTIMIZER_VECTOR_TRANSFORMATION must be set to FALSE on all instances

--Info: Table last updated OHI release 10.24.1.0.0

SPECIFIC OHI BACK OFFICE CHECKS

FATAL	Queue SYS_BATCH_QUEUE must exist and belong to the OZG schema owner
FATAL	All user tablespaces must use ASSM, 8K block size and SYSTEM extent allocation type
ERROR	All mandatory Privileges for OZG schema owner must be present; see “Mandatory Privilege” overview
ERROR	All mandatory Privileges for BATCH account must be present; see “Mandatory Privilege” overview
ERROR	All mandatory Privileges for OBD_SELECT_USER account must be present; see “Mandatory Privilege” overview
ERROR	Check if a database user that owns customizations on OHI BO (<USER>) owns one or more database objects with the same name as an object owned by the OZG schema owner
ERROR	Check if the required “private grants” on SYS and XDB objects have been issued to OHI BO owner, other OHI BO accounts and OHI_ROLE_ALL.
ERROR	Check if OHI BO roles have been granted illegally to other OHI roles or OHI BO accounts.
ERROR	Check if obsolete BATCH user and roles have been removed.
WARNING	Reported the number of not validated Business Rules
WARNING	Reported when Partitioning Option is installed, but not or partially used; recommended to execute step 850
WARNING	Reported when the tablespace for the partitioned table is not correct
WARNING	Recommendation to set the retention time for queue SYS_BATCH_QUEUE to two weeks
WARNING	Reported on any table or index with incorrect tablespace
WARNING	Recommendation to set INI_TRANS to value >=4 for all normal data tables (e.g. table names without '$' on 4th position)
WARNING	Recommendation to set PCT_FREE to OHI advised value (default 15) for all normal data tables (e.g. table names without '$' on 4th position)
WARNING	Recommendation to set PCT_FREE to OHI advised value (default 5) for mutation log tables (e.g. table names with '$' on 4th position)
WARNING	Recommendation to revoke all granted but unnecessary privileges for OZG schema owner
WARNING	Recommendation to revoke all granted but unnecessary privileges for BATCH account
WARNING	Recommendation to revoke all granted but unnecessary privileges for OBD_SELECT_USER account
ERROR	Check that SYS does not have a role privilege for any OHI role
ERROR	Check that OHI BO owner has ADMIN option for all (!) present OHI roles
ERROR	Check that OHI BO owner has all OHI roles
ERROR	When DV is enabled, the role DV_SECANALYST must be present for the role OHI_ROLE_ALL

INFO	When DV is enabled, the type of DV accounts is reported: Message: “Database Vault is managed with common accounts (C##*) for all containers.” or “Database Vault is managed with local accounts.”
ERROR	When DV is enabled, the role DV_PATCH_ADMIN should not be granted.
WARNING	When DV is enabled, privileges for Data Pump should not be granted permanently (using DBMS_MACADM.AUTHORIZE_DATAPUMP_USER)
	Database instance parameter checks:
FATAL	Parameter AQ_TM_PROCESSES must be 1 on all instances
FATAL	Parameter DB_BLOCK_SIZE must be 8k
FATAL	Parameter DML_LOCKS must be >= 500 on all instances
FATAL	Parameter OPTIMIZER_MODE must be set to ALL_ROWS on all instances
FATAL	During installation, parameter PARALLEL_MAX_SERVERS must be set > 1 on all instances
INFO	Parameter OPTIMIZER_USE_SQL_PLAN_BASELINES is set to TRUE (checked for all instances)
ERROR	Parameter HASH_AREA_SIZE must be set to a specified value at least during installations, on all instances
ERROR	Parameter PGA_AGGREGATE_LIMIT must be >= 12GB on all instances
ERROR	Parameter REMOTE_DEPENDENCIES_MODE must be set to SIGNATURE at runtime
ERROR	Parameter SORT_AREA_SIZE must be set to a specified value at least during installations, on all instances
ERROR	Parameter “_OPTIMIZER_USE_STATS_ON_CONVENTIONAL_DML” must be set to FALSE on Engineered Systems (ExaData, ExaCS, etc.)
WARNING	Parameter HASH_AREA_SIZE must have a specified value of at least 100Mb on all instances
WARNING	Parameter PARALLEL_ADAPTIVE_MULTI_USER must be set to TRUE when PARALLEL_MAX_SERVERS is set on a value larger than zero.
WARNING	Parameter PARALLEL_DEGREE_POLICY must be MANUAL or AUTO when PARALLEL_MAX_SERVERS is set on a value larger than zero.
WARNING	Parameter PARALLEL_FORCE_LOCAL must be set to TRUE in a clustered database when PARALLEL_MAX_SERVERS is set on a value larger than zero.
WARNING	Parameter SORT_AREA_SIZE must have a specified value of at least 200Mb on all instances
WARNING	Parameter _COMMON_DATA_VIEW_ENABLED needs to be set at TRUE (either explicit or by not setting it using the default)
ERROR	Check that OHI BO owner has ADMIN option on all OHI roles
ERROR	Check that SYS does not have ADMIN option on OHI roles
ERROR	Check that OHI BO owner has all OHI roles
ERROR	Check that OHI Application users do not have system privileges or OHI Database roles

INFO	List all database accounts and database roles that have been granted an OHI database role.

--Info: Table last updated OHI release 10.26.1.0.0

SPECIFIC OHI DATA MARTS CHECKS

FATAL	All user tablespaces must use ASSM, either 8K or 16k block size and must use either SYSTEM or UNIFORM extent allocation type
ERROR	Only one database link to OZG may exist
ERROR	OHI Back Office release must correspond to the OHI Data Marts release
ERROR	All mandatory Privileges for OBD schema owner must be present; see “Mandatory Privilege” overview
WARNING	Reported when the most recent OHI BI data load process did not complete (successfully)
WARNING	Recommendation to set PCT_INCREASE to value 0 for all normal data tables (e.g. table names without ‘$’ on 4th position)
WARNING	Recommendation to revoke all granted but unnecessary privileges for OBD schema owner
INFO	Reported database link SRC_OPENZORG to which host
INFO	Reported the OHI BackOffice release found
INFO	Reported the status of the OHI BackOffice data load process
ERROR	When DV is enabled, the role DV_SECANALYST must be present for the role OBD_ROL_ADMIN
	Database instance parameter checks:
FATAL	Parameter DB_BLOCK_SIZE must be either 8k or 16k
FATAL	Parameter OPTIMIZER_MODE must be set to ALL_ROWS on all instances
FATAL	Parameter PARALLEL_MAX_SERVERS must be set > 1 on all instances
FATAL	Parameter STAR_TRANSFORMATION_ENABLED must be set to TRUE on all instances
WARNING	With the use of parameter PARALLEL_DEGREE_POLICY an IOResource Plan is recommended.
WARNING	Parameter PARALLEL_FORCE_LOCAL must be set to TRUE in a clustered database when PARALLEL_MAX_SERVERS is set on a value larger than zero.
WARNING	Parameter PARALLEL_THREADS_PER_CPU is only recommended on Exadata and should be 1 on Exadata.
ERROR	Parameter SYSDATE_AT_DBTIMEZONE must be set to TRUE; this requirement is only for an Autonomous Data Warehouse instance

--Info: Table last updated OHI release 10.26.1.0.0

APPENDIX E DATABASE MANDATORY USER PRIVILEGES

During the Database Parameters check, described in Appendix D, mandatory privileges for system/owner accounts are checked. Missing mandatory privileges are reported with an ERROR, unnecessary privileges are reported as WARNING.

This appendix gives an overview of all mandatory user privileges required for OHI to function correctly.

The BATCH account must be present in both OHI BO and OHI DM and the OBD_SELECT_USER account must exist within OHI BO for extracting data.

When a privilege value is a role and the role is optional - meaning it depends on the way OHI is used - the role is placed between parenthesis.

PrivType	Privilege	BackOffice Owner	Back Office View Owner	Back Office Dyn.PL/SQL user	Data Marts	BATCH account	OBD_SELECT_USER
DB Priv	ADMINISTER DATABASE TRIGGER	X			X		
DB Priv	ADMINISTER SQL MANAGEMENT OBJECT	X					
DB Priv	ADVISOR	X			X		
DB Priv	ALTER SESSION	X	X	X	X	X	
DB Priv	ANALYZE ANY				X		
DB Priv	CREATE JOB	X			X		
DB Priv	CREATE PROCEDURE	X	X	X	X		
DB Priv	CREATE PUBLIC SYNONYM	X			X		
DB Priv	CREATE SEQUENCE	X			X		
DB Priv	CREATE SESSION	X	X	X	X	X	X
DB Priv	CREATE SYNONYM	X	X		X		
DB Priv	CREATE TABLE	X			X		
DB Priv	CREATE TRIGGER	X	X		X		
DB Priv	CREATE VIEW	X	X		X		X
DB Priv	DROP PUBLIC SYNONYM	X			X		
DB Priv	MANAGE SCHEDULER	X			X		
DB Priv	RESTRICTED SESSION	X	X	X	X		
DB Priv	SELECT ANY DICTIONARY	X	X	X	X	X	
DB Role	SELECT_CATALOG_ROLE	X			X	X	
DB Priv	CREATE DATABASE LINK				X		
DB Priv	CREATE DIMENSION				X		
DB Priv	CREATE EXTERNAL JOB				X		
DB Priv	ALTER SYSTEM	X					
DB Role	AQ_ADMINISTRATOR_ROLE	X					
DB Priv	CREATE ANY CONTEXT	X					

PrivType	Privilege	BackOffice Owner	Back Office View Owner	Back Office Dyn.PL/SQL user	Data Marts	BATCH account	OBD_SELECT_USER
DB Priv	CREATE TYPE	X			X		
DB Priv	CREATE MATERIALIZED VIEW	X					
DB Priv	DROP ANY CONTEXT	X					
DB Priv	EXEMPT ACCESS POLICY				X		
DB Priv	RESOURCE						
DB Priv	INHERIT PRIVILEGES ON USER SYS	X					
OHI Role	OZG_ROL	X					
OHI Role	OHI_ROLE_ALL	X				X	
OHI Role	OZG_ROL_SELECT	X					
OHI Role	OHI_ROLE_EXTRACT	X					
OHI Role	OZG_ROL_DIRECT	X					
OHI Role	OHI_ROLE_BATCH	X					
SysObjPriv	EXECUTE ON SYS.DBMSOUTPUT_LINESARRAY	X					
SysObjPriv	EXECUTE ON SYS.DBMS_ALERT	X			X		
SysObjPriv	EXECUTE ON SYS.DBMS_APPLICATION_INFO	X					
SysObjPriv	EXECUTE ON SYS.DBMS_AQ	X			X		
SysObjPriv	EXECUTE ON SYS.DBMS_AQADM	X			X		
SysObjPriv	EXECUTE ON SYS.DBMS_ASSERT	X					
SysObjPriv	EXECUTE ON SYS.DBMS_CRYPTO	X			X		
SysObjPriv	EXECUTE ON SYS.DBMS_FLASHBACK	X					
SysObjPriv	EXECUTE ON SYS.DBMS_HPROF	X					
SysObjPriv	EXECUTE ON SYS.DBMS_JOB	X	X				
SysObjPriv	EXECUTE ON SYS.DBMS_LOB	X					
SysObjPriv	EXECUTE ON SYS.DBMS_LOCK	X					
SysObjPriv	EXECUTE ON SYS.DBMS_METADATA	X					
SysObjPriv	EXECUTE ON SYS.DBMS_OUTPUT	X					
SysObjPriv	EXECUTE ON SYS.DBMS_PROFILER	X					
SysObjPriv	EXECUTE ON SYS.DBMS_RANDOM	X	X				
SysObjPriv	EXECUTE ON SYS.DBMS_REDEFINITION	X					
SysObjPriv	EXECUTE ON DBMS_RLS	X			X		
SysObjPriv	EXECUTE ON SYS.DBMS_ROWID	X					
SysObjPriv	EXECUTE ON SYS.DBMS_SCHEDULER	X					
SysObjPriv	EXECUTE ON SYS.DBMS_SESSION	X					
SysObjPriv	EXECUTE ON SYS.DBMS_SHARED_POOL	X					
SysObjPriv	EXECUTE ON SYS.DBMS_SPM	X					
SysObjPriv	EXECUTE ON SYS.DBMS_SQL	X		X			

PrivType	Privilege	BackOffice Owner	Back Office View Owner	Back Office Dyn.PL/SQL user	Data Marts	BATCH account	OBD_SELECT_USER
SysObjPriv	EXECUTE ON SYS.DBMS_STANDARD	X	X	X			
SysObjPriv	EXECUTE ON SYS.DBMS_STATS	X					
SysObjPriv	EXECUTE ON SYS.DBMS_SYSTEM	X			X		
SysObjPriv	EXECUTE ON SYS.DBMS_TRANSACTION	X					
SysObjPriv	EXECUTE ON SYS.DBMS_UTILITY	X					
SysObjPriv	EXECUTE ON SYS.DBMS_WORKLOAD_REPOSITORY	X			X		
SysObjPriv	EXECUTE ON SYS.DBMS_XMLGEN	X					
SysObjPriv	EXECUTE ON SYS.DBMS_XPLAN	X					
SysObjPriv	EXECUTE ON SYS.UTL_CALL_STACK	X					
SysObjPriv	EXECUTE ON SYS.UTL_COMPRESS	X					
SysObjPriv	EXECUTE ON SYS.UTL_ENCODE	X					
SysObjPriv	EXECUTE ON SYS.UTL_FILE	X					
SysObjPriv	EXECUTE ON SYS.UTL_HTTP	X					
SysObjPriv	EXECUTE ON SYS.UTL_I18N	X					
SysObjPriv	EXECUTE ON SYS.UTL_RAW	X					
SysObjPriv	EXECUTE ON SYS.UTL_RECOMP	X					
SysObjPriv	EXECUTE ON SYS.UTL_URL	X					
SysObjPriv	SELECT ON SYS.V_$DATABASE	X					
SysObjPriv	SELECT ON DBA_TABLES	X					X
SysObjPriv	SELECT ON DBA_TABLE_STATISTICS	X					X
SysObjPriv	EXECUTE ON XDB.DBMS_XMLDOM	X					
SysObjPriv	EXECUTE ON XDB.DBMS_XMLPARSER	X					
SysObjPriv	EXECUTE ON XDB.DBMS_XMLSCHEMA	X					
SysObjPriv	EXECUTE ON XDB.DBMS_XSLPROCESSOR	X					

--Info: Table last updated OHI release 10.20.5.0.0

APPENDIX F DATABASE VAULT – RELEVANT CHECKS

OHI BACK OFFICE

When Database Vault is enabled and the OHI realm for OHI Back Office has been created, a number of additional checks are executed.

- If the OHI realm is present, it should be enabled.
- If the OHI realm is present, a message is given whether Operations Control is enabled.
- The OHI table owner must have Database Vault proxy authorizations for the target users OHI_VIEW_OWNER and OHI_DPS_USER.
- All users with name OHI_REALM_AUTH% that are currently logged on are listed. This is because they all might have authorized the potential current OHIPATCH action by logging in (if the check is performed during an OHIPATCH action and not as part of a batch action to run the checks).
- Check the users that have OHI realm authorization:
    - There should be no owners other than the three kown owners.
    - Participants registered by the customer should have at least one direct grant on OHI BO objects.
- All unknown users are listed that have OHI realm authorization. These should be participants registered by the customer, e.g .for custom development. This should be a very limited list that should be reviewed. This direct realm authorization is retrieved from view DBA_DV_REALM_AUTH.
- If the OHI Realm is absent, report on users having role OHI_REALM_ACCESS. When the realm is present report on users missing the role.
- Check users with a name like OHI_REALM_AUTH%.
    - They should not be registered as a Back Office functional user (a “functionaris”).
    - They should have no database roles, no privileges on OHI BO objects and only CREATE SESSION privilege and RESTRICTED SESSION privilege.
- Check that database users with role OHI_REALM_ACCESS are not a Back Office functional user (a "functionaris").

OHI DATA MARTS

Please see the checks for OHI Back Office and ignore the Back Office specific checks, like those for the additional owners present in OHI Back Office only.