

Oracle Health Insurance Back Office

Agent Installation & Configuration Manual

Version 1.2

Part number: G49637-01

October 7, 2026

Copyright © 2026, Oracle and/or its affiliates. All rights reserved.

Public

Purpose statement

This document provides an overview of the architecture used for the Oracle Health Insurance Back Office AI Agents and installation and configuration instructions to integrate the AI Agents in your Oracle Health Insurance Back Office environments.

Disclaimer

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this software or related documentation is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS

Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are “commercial computer software” or “commercial technical data” pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation shall be subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License (December 2007). Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

This software is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications which may create a risk of personal injury. If you use this software in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure the safe use of this software. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software in dangerous applications.

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

This software and documentation may provide access to or information on content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

Where an Oracle offering includes third party content or software, we may be required to include related notices. For information on third party notices and the software and related documentation in connection with which they need to be included, please contact the attorney from the Development and Strategic Initiatives Legal Group that supports the development team for the Oracle offering. Contact information can be found on the Attorney Contact Chart.

The information contained in this document is for informational sharing purposes only and should be considered in your capacity as a customer advisory board member or pursuant to your beta trial agreement only. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described in this document remains at the sole discretion of Oracle.

This document in any form, software or printed matter, contains proprietary information that is the exclusive property of Oracle. Your access to and use of this confidential material is subject to the terms and conditions of your Oracle Software License and Service Agreement, which has been executed and with which you agree to comply. This document and information contained herein may not be disclosed, copied, reproduced, or distributed to anyone outside Oracle without prior written consent of Oracle. This document is not part of your license agreement nor can it be incorporated into any contractual agreement with Oracle or its subsidiaries or affiliates.

Change History

Release	Version	Changes
10.26.5.0.0	1.0	Creation
10.26.6.0.0	1.1	Documented the use of an Autonomous Transaction Processing (ATP) database instead of a regular 26ai database for the RAG component. Added instructions for OCI privileges that are needed by the OCI principal account used for accessing an OCI Generative AI endpoint.
10.26.7.0.0	1.2	• Use of non-oci credential and profiles added for a non-OCI provided LLM. CPU sizing advice adjusted. • Use of Autonomous Transaction Processing (ATP) database is now mandatory and the only supported option • Dynamic database schemas

Table of contents

Purpose	5
Installing the OHI Online Help AI Assistant objects in an Oracle database 26ai PDB	5
Overview	5
Scope and assumptions	6
Release scope	7
DBA actions before running the installer	7
DBMS_CLOUD and Oracle AI packages	8
Vector memory	8
Database accounts and grants	8
Tablespaces	8
Database accounts	8
Quota	9
System privileges for OHI_AGENT_OWNER	9
Runtime and model-owner privileges	9
Network access to non-OCI endpoints	9
OCI credential and AI profile	10
Non-OCI credential and AI profile	12
Input files	13
Wallet aliases	15
Preparing the SQL scripts	16
Running the installation	16
Installer activities	17
Post-installation validation	17
Trace retention	18
OHI Back Office integration	19
Installing a higher major release of the Online Help data	19
Cloning an OHI Back Office environment	20
Patching an OHI Back Office environment	20

Purpose

This document provides an overview of the architecture used for the Oracle Health Insurance Back Office AI Agents and it contains installation and configuration instructions to integrate the AI Agents in your Oracle Health Insurance Back Office (OHI BO) environments.

Currently, Oracle Health Insurance Back Office has only one AI Agent: the OHI Online Help AI Assistant.

When new AI agents become available, instructions for those agents will be added in this document.

Installing the OHI Online Help AI Assistant objects in an Oracle database 26ai PDB

This chapter describes the database preparation and installation procedure for the OHI Online Help AI Assistant database objects in a dedicated Oracle Database 26ai Pluggable Database (PDB). This must be an Autonomous Transaction Processing Database (ATP).

Given the very frequent functional enhancements for an autonomous database, AI agents for OHI BO are only supported on an autonomous database, so OHI AI agents can use agentic functionality early. Agentic functionality is often released later for regular databases.

At this moment the Standard Edition of the 26ai database is sufficient for running the agentic functionality.

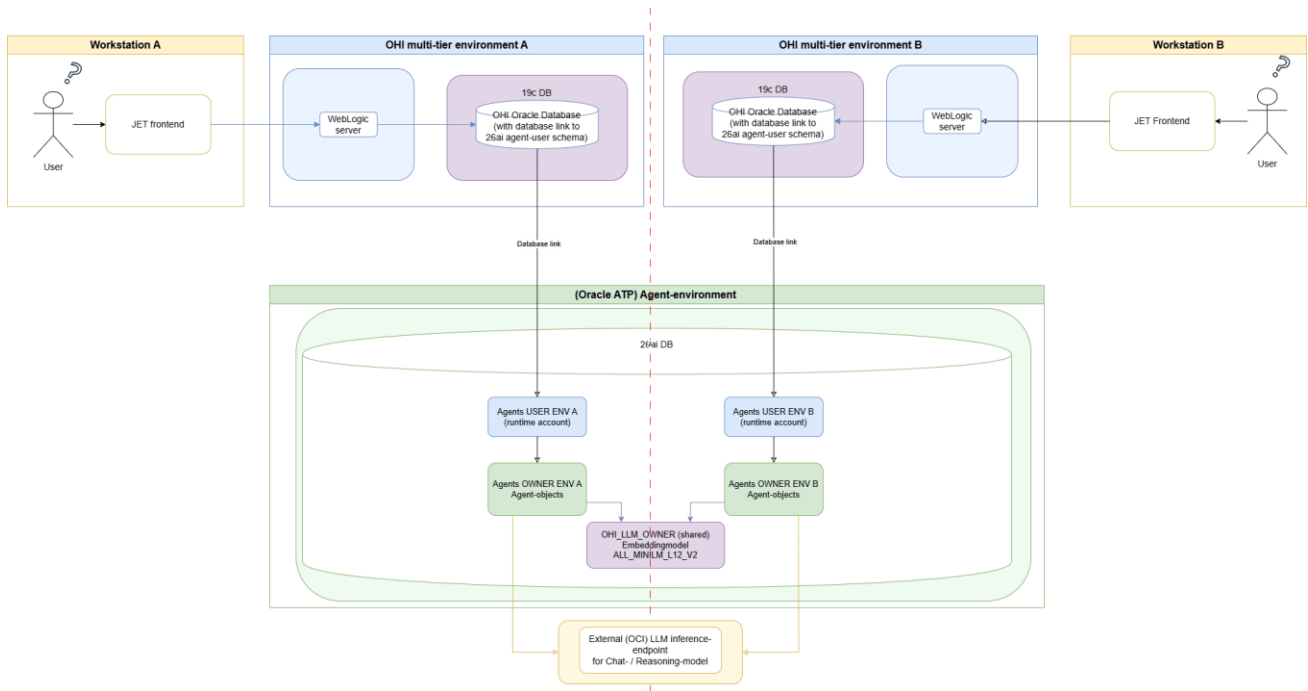
Overview

The database stores and indexes OHI Back Office Online Help content using Oracle Text and Oracle Vector Search. This enables a Retrieval-Augmented Generation (RAG) solution, where a Large Language Model (LLM) combines its reasoning capabilities with information retrieved from the OHI 'RAG database'. The reasoning model determines a search phrase, based on the question asked and the history in the chat conversation. That search phrase is passed to the search function in the RAG database. The search phrase is decomposed into a tokenized sentence for Oracle Text to search the Online Help information. An LLM embedding model is used to generate a vector representation that is used to search in the vector representation of the Online Help information. The top N best scoring results are returned to the reasoning LLM, enabling accurate and context-aware responses.

The agentic functionality itself is implemented through [Select AI Agent](#) functionality as this is offered by the Oracle 26ai database.

A database link between the OHI BO database and the 26ai database is used to call the 26ai agentic functionality, where the agent itself uses the described search functionality as a 'tool' to extend its general knowledge with OHI BO specific Online Help information. As such the OHI Online Help agent/assistant has no access to the OHI BO database and cannot answer any questions related to the data in the OHI BO environment.

The following figure depicts at a high level how the different components interact.



Multiple OHI Back Office environments may share the same PDB, provided that all environments use their own agent owner and agent user schemas. The LLM owner schema may be shared across different agent owner schemas as it only holds the embedded LLM model, but one LLM owner per agent schema is also possible.

The agent owner schema contains the agent objects. The agent user schema is the account used to submit user questions to the agent. The LLM owner schema contains the embedded LLM model used to generate the vector representations for the Online Help agent's RAG implementation.

For each OHI Back Office environment, choose unique account names for the agent owner and agent user schemas. Choose a unique name for the LLM owner schema as well, unless the LLM owner account is shared. These account names are requested during installation. If the LLM owner account is to be shared, specify the existing LLM owner account to be used.

For readability, this document uses OHI_AGENT_OWNER, OHI_AGENT_USER and OHI_LLM_OWNER to refer to the respective accounts. These names are illustrative; the account names used during installation may be selected according to local naming conventions.

The installation of the required schema objects in the 26ai database is performed using SQL*Plus or SQLcl and a set of SQL scripts delivered in an OHI release. The scripts are located in the "\$OZG_BASE/conf/ohi-agents" directory of the OHI Back Office environment.

As described above, the OHI Online Help AI Assistant uses the Oracle Database 26ai "Select AI Agent" and Vector Search capabilities.

The procedures in this chapter are intended for database administrators responsible for:

- Preparing the target 26ai PDB.
- Creating and configuring the required database schemas.
- Configuring access to Oracle Cloud Infrastructure (OCI) Generative AI services.
- Executing the database installation scripts and validating the installation.

Scope and assumptions

This installation procedure assumes a 26ai Autonomous Transaction Processing PDB (running as a service in Oracle Cloud Infrastructure and managed by OCI).

For the OHI Online Help AI Assistant, the target PDB should have a minimum SGA size of 5 GB available, including at least 1 GB allocated to the shared pool and 1 GB allocated to the database buffer cache.

As it is hard to predict the actual workload generated by the OHI Online Help AI assistant, we suggest you use autoscaling for the Autonomous Transaction Processing database and start with a peak ECPU configuration of 4. If you observe the peak load is reached very often and the handling of the database load is delayed, you may increase the base ECPU setting and increase the peak ECPU and set it to 8 or 12.

The installation creates and configures three database accounts, installs the OHI Agent(s) database objects, configures access to Oracle AI functionality, loads the documents and terms as needed for the (RAG) agent solution, and builds the hybrid vector index.

- OHI_AGENT_OWNER owns the tables, packages, views, Oracle Text configuration, scheduler job, and hybrid vector index for the RAG and other agents.
- OHI_AGENT_USER is the runtime user that calls the agent packages.
- OHI_LLM_OWNER owns or exposes the embedding model that is used to store vector representation in the vector index for the Online Help textual information.

The installation is executed with Secure External Password Store (SEPS) wallet aliases for the three accounts.

During the installation three files need to be loaded.

One file contains a so-called embedding model in ONNX format, which is used to determine the vector values representing the meaning of the text that is embedded. It is a relatively small LLM model that will be loaded into the database.

The Online Help information itself also needs to be loaded and is stored in two *.ndjson files.

Release scope

The OHI (RAG) agent installation is tied to a specific OHI release level. The installer validates the release level of the existing installation and prevents incompatible deployments.

DBA actions before running the installer

Carry out the following actions in the target database, typically using ADMIN in an autonomous database, unless explicitly stated otherwise.

Determine the passwords, default tablespace, quotas, and the input-file location appropriate for the target database type before starting the installation.

- Account names for the agent owner, agent user and LLM owner.
- Password for OHI_AGENT_OWNER.
- Password for OHI_AGENT_USER.
- Password for OHI_LLM_OWNER.
- Default tablespace, typically DATA. Make sure this can grow with 15Gb.
- Tablespace quota for OHI_AGENT_OWNER, 10G will be sufficient.
- Tablespace quota for OHI_LLM_OWNER, 5G will be sufficient.
- OCI Object Storage bucket URI prefix for the ONNX model and NDJSON files that need to be loaded.

NB. The (schema) names listed here are included for reference. You are free to define your own schema names. A reference to the corresponding OHI Back Office environment can, for example, be added to these names.

DBMS_CLOUD and Oracle AI packages

The OHI Online Help AI Assistant relies on Oracle Database components that provide access to Oracle Cloud Infrastructure (OCI) services and AI functionality:

- DBMS_CLOUD provides database access to OCI resources and credentials management.
- DBMS_CLOUD_AI and DBMS_CLOUD_AI_AGENT provide integration with Oracle AI services, including the Select AI Agent functionality used by the OHI Online Help AI Assistant solution.

Vector memory

Oracle Vector Search uses a Hierarchical Navigable Small World (HNSW) index to efficiently perform approximate nearest-neighbour searches on vector embeddings. Creating and maintaining HNSW indexes requires vector memory and must be at least 1 GB. In an autonomous database this is already arranged.

Check the current setting:

```
SHOW PARAMETER vector_memory_size;
```

The ATP service must provide sufficient vector capacity to build the HNSW index. If index creation fails because of capacity, ask the Autonomous Database administrator to assess the service configuration.

Database accounts and grants

The following actions must be performed in the target pluggable database (PDB).

Tablespaces

The OHI agent accounts use the database default tablespace in the autonomous database.

The current default tablespaces can be verified as follows:

```
select property_name
,      property_value as default_tablespace
from   database_properties
where  property_name in ('DEFAULT_TEMP_TABLESPACE', 'DEFAULT_PERMANENT_TABLESPACE');
```

In the commands below, replace <DATA_TABLESPACE> with the tablespace name appropriate for the target environment. This usually is the DATA tablespace.

Database accounts

Create the three accounts in the target PDB. Replace the password placeholders and tablespace name with values appropriate for the local environment.

```
create user OHI_AGENT_OWNER identified by "<OHI_AGENT_OWNER_PASSWORD>"
  default tablespace <DATA_TABLESPACE>
  account unlock;
create user OHI_AGENT_USER identified by "<OHI_AGENT_USER_PASSWORD>"
  default tablespace <DATA_TABLESPACE>
  account unlock;
create user OHI_LLM_OWNER identified by "<OHI_LLM_OWNER_PASSWORD>"
  default tablespace <DATA_TABLESPACE>
  account unlock;
```

Make sure the passwords do not expire or add an action to your calendar to change them before the expiration date. Do not forget to update the wallet entries too when the passwords are changed (see below).

Quota

Assign quota to the owner schemas. The values shown below are the recommended minimum quotas for the current implementation and may be increased if additional content is expected in the future.

```
alter user OHI_AGENT_OWNER quota 10G on <DATA_TABLESPACE>;
alter user OHI_LLM_OWNER quota 5G on <DATA_TABLESPACE>;
```

System privileges for OHI_AGENT_OWNER

Grant the privileges required to create the agent schema objects, text index, scheduler job, public synonym, and Oracle AI integration objects.

```
grant create session to OHI_AGENT_OWNER;
grant alter session to OHI_AGENT_OWNER;
grant create table to OHI_AGENT_OWNER;
grant create view to OHI_AGENT_OWNER;
grant create procedure to OHI_AGENT_OWNER;
grant create public synonym to OHI_AGENT_OWNER;
grant create job to OHI_AGENT_OWNER;
grant select any dictionary to OHI_AGENT_OWNER;
grant ctxapp to OHI_AGENT_OWNER;
grant execute on ctxsys.ctx_ddl to OHI_AGENT_OWNER;
grant execute on ctxsys.dbms_vector_chain to OHI_AGENT_OWNER;
grant execute on dbms_cloud to OHI_AGENT_OWNER;
grant execute on dbms_cloud_ai to OHI_AGENT_OWNER;
grant execute on dbms_cloud_ai_agent to OHI_AGENT_OWNER;
grant create credential to OHI_AGENT_OWNER;
grant create any context to OHI_AGENT_OWNER;
```

Runtime and model-owner privileges

Grant the required system privileges to OHI_AGENT_USER.

```
grant create session to OHI_AGENT_USER;
grant alter session to OHI_AGENT_USER;
grant create synonym to OHI_AGENT_USER;
```

Grant the required system privileges to OHI_LLM_OWNER.

```
grant create session to OHI_LLM_OWNER;
grant alter session to OHI_LLM_OWNER;
grant create mining model to OHI_LLM_OWNER;
grant execute on dbms_cloud to OHI_LLM_OWNER;
grant create credential to OHI_LLM_OWNER;
```

Network access to non-OCI endpoints

The instructions below are not required for OCI Generative AI endpoints when using an Autonomous Database. Network access to OCI endpoints is already managed by Oracle.

If the AI profile uses an endpoint outside OCI—for example, an OpenAI endpoint or another third-party AI provider—network access must be explicitly granted through a network ACL.

The value supplied for the host parameter must match the hostname of the external AI endpoint configured in the AI profile. For security reasons, use the most specific hostname possible rather than a broad wildcard pattern.

The following example uses OpenAI as a hostname pattern. Replace this value with the hostname of the external endpoint that needs to be accessed.

Execute the following as a DBA or APEX administrator in the OHI BO PDB:

```
begin
  dbms_network_acl_admin.append_host_ace
  ( host => 'api.openai.com'
  , ace => xs$ace_type
    ( privilege_list => xs$name_list('http')
    , principal_name => 'OHI_AGENT_OWNER'
    , principal_type => xs_acl.p_type_db
    )
  );
end;
/
```

OCI credential and AI profile

You need to use the `dbms_cloud` package to create an OCI credential in the database as well as a Generative AI profile that uses this credential. Create the OCI credential for the Generative AI profile while connected as `OHI_AGENT_OWNER`. Replace the anonymized example values with the values for the target tenancy, user/principal, key, and fingerprint.

For instructions on obtaining the required OCI values (user OCID, tenancy OCID, API key fingerprint, and private key), refer to the [Oracle Cloud Infrastructure API signing key documentation](#).

Information on locating tenancy and user OCIDs is available in the [OCI documentation](#).

Make sure you use a user principal that has received access to the Generative AI services. To provide this access you can create a group within your (default) domain under the User Management tab and grant this specific group access. Below an example group is shown. Assign the relevant users/principals to the group.

The screenshot shows the Oracle Identity & Security console. The left sidebar has a menu with 'Identity & Security' at the top, followed by 'Identity' (with a sub-menu: Overview, Domains, Network Sources, Policies), 'Comppartments', and 'My profile'. The main content area is titled 'User management' and includes a search bar, 'Applied filters', and a table of groups. The table has columns for 'Name' and 'Description'. One group is listed: 'GenAIEndpointAccess' with the description 'Group to test minimal required privs for GenAI endpoint access'.

An example to provide the group access to the generative AI functionality is shown below by creating a policy with a statement stating the access in a generic way for the tenancy:

The screenshot shows the Oracle Identity & Security console. The left sidebar is the same as the previous screenshot. The main content area is titled 'Policies' and shows a policy named 'GenAITestAccessPolicy' with an 'Active' status. Below the policy name, there are tabs for 'Policy Information', 'Statements', and 'Tags'. The 'Policy Statements' tab is selected, showing a search bar and an 'Edit Policy Statements' button. Below this, a single statement is listed: 'Allow group GenAIEndpointAccess to use generative-ai-family in compartment id ocid1.tenancy.oc1.aaaaaaaapfrconjyw5yvwsw5...'. The statement text is partially obscured by a black redaction bar.

Using the OCI user principal selected above for access to OCI Generative AI, create a credential in the OHI_AGENT_OWNER schema of the agent environment being installed. Connect as OHI_AGENT_OWNER and run the following command, adapting the values to your OCI configuration:

begin

```
dbms_cloud.create_credential
( credential_name => 'OHI_OCI_CRED'
, user_ocid       => 'ocid1.user.oc1..example_anonymized_user_ocid'
, tenancy_ocid    => 'ocid1.tenancy.oc1..example_anonymized_tenancy_ocid'
, private_key     => 'EXAMPLE_ANONYMIZED_PRIVATE_KEY_LINE_1
                     EXAMPLE_ANONYMIZED_PRIVATE_KEY_LINE_2
                     EXAMPLE_ANONYMIZED_PRIVATE_KEY_LINE_3'
, fingerprint     => '00:00:00:00:00:00:00:00:00:00:00:00:00:00:00'
, comments        => 'OHI OCI Cloud credentials'
);
```

end;

/

Create the AI profile in the same OHI_AGENT_OWNER schema after the credential is available. Configure the compartment OCID, region and large language model (LLM) for that agent environment.

If multiple OHI Back Office environments are installed, create a separate credential and AI profile in each environment's OHI_AGENT_OWNER schema. The same OCI principal and key may be used to create the credential separately in each schema, provided that the principal has the required access.

begin

```
dbms_cloud_ai.create_profile
( profile_name => 'OHI_OCI_PROFILE'
, attributes   => JSON_OBJECT
  ( 'provider'      value 'oci'
  , 'credential_name' value 'OHI_OCI_CRED'
  , 'region'        value 'eu-frankfurt-1'
  , 'oci_compartment_id' value 'ocid1.compartment.oc1..example_anonymized_compartment_ocid'
  , 'model'         value 'google.gemini-2.5-flash'
  , 'max_tokens'    value 65536
  , 'object_list'   value json_array( json_object('owner' value 'OHI_AGENT_OWNER')
                                     ) format json
  )
);
```

end;

/

The names OHI_OCI_CRED and OHI_OCI_PROFILE are used as examples in this document. You may use these names or choose names that follow your local naming conventions. Ensure that the credential_name specified in the AI profile matches the name of the credential created in OHI_AGENT_OWNER.

Test the profile before running the RAG installation while connected as OHI_AGENT_OWNER. If you chose a different profile name, use that name in the test query.

```

select dbms_cloud_ai.generate
  ( prompt      => 'tell me something interesting about the city of paris?'
    , profile_name => 'OHI_OCI_PROFILE'
    , action      => 'chat'
    ) as response
from dual;

```

To import files into Autonomous AI Database, create Object Storage credentials in the owning schemas. You may reuse a credential created for access to OCI Generative AI services if its OCI principal also has access to the bucket.

This document refers to the credential in OHI_AGENT_OWNER as OHI_RAG_OCI_CRED and the credential in OHI_LLM_OWNER as OHI_LLM_OCI_CRED. Use the same DBMS_CLOUD.CREATE_CREDENTIAL pattern and grant the underlying OCI principal read access only to the RAG bucket or prefix.

Before running the installation, enter the actual AI profile name, Object Storage credential names and bucket values in the corresponding parameters in OHI_AGENT_DB_INSTALL_PARAMETERS.sql.

Non-OCI credential and AI profile

Alternatively, use a non-OCI endpoint with a supported third-party AI provider. The example below configures direct access to OpenAI. The database sends requests to api.openai.com; it does not use an OCI Generative AI endpoint or OCI API-signing credentials.

Create an OpenAI API key in the OpenAI platform and make sure the database user OHI_AGENT_OWNER has network access to api.openai.com See Network access to non-OCI endpoints.

Create the credential while connected as OHI_AGENT_OWNER. Replace the anonymized API key with the OpenAI API key for the target environment:

```

begin
  dbms_cloud.create_credential
  ( credential_name => 'OHI_OPENAI_CRED'
    , username      => 'OPENAI'
    , password      => 'EXAMPLE_ANONYMIZED_OPENAI_API_KEY'
    , comments      => 'OHI OpenAI API credentials'
    );
end;
/

```

Create the AI profile after the credential is available. The model name is environment specific and must be available for the OpenAI API key used by the target environment.

```

begin
  dbms_cloud_ai.create_profile
  ( profile_name => 'OHI_OPENAI_PROFILE'
    , attributes  => json_object
      ( 'provider'      value 'openai'
        , 'credential_name' value 'OHI_OPENAI_CRED'
        , 'model'       value 'gpt-5.4-mini'
        , 'max_tokens'  value 65536
        , 'object_list' value json_array( json_object
                                          ( 'owner' value 'OHI_AGENT_OWNER' )
                                          ) format json
      )
    );
end;
/

```

Test the profile before running the RAG installation:

```
select dbms_cloud_ai.generate
  ( prompt      => 'tell me something interesting about the city of paris?'
    , profile_name => 'OHI_OPENAI_PROFILE'
    , action      => 'chat'
  ) as response
from dual;
```

For more information about creating and using AI profiles and credentials, see paragraph “About Select AI” in the “Oracle Cloud Using Oracle Autonomous AI Database Serverless” documentation.

Input files

On Autonomous AI Database, the installer retrieves the input files directly from OCI Object Storage, not from a customer-created database directory.

Input files

- documents.ndjson
- terms.ndjson
- all_MiniLM_L12_v2.onnx

Copy documents.ndjson and terms.ndjson from \$OZG_BASE/help/nl_nl/Output/OHIHelp/RAG. Download all_MiniLM_L12_v2.onnx from the link below:

https://adwc4pm.objectstorage.us-ashburn-1.oci.customer-oci.com/p/TtH6hL2y25EypZ0-rrczRZ1aXp7v1ONbRBfCiT-BDBN8WLKQ3lgyW6RxCfIFLdA6/n/adwc4pm/b/OML-ai-models/o/all_MiniLM_L12_v2_augmented.zip

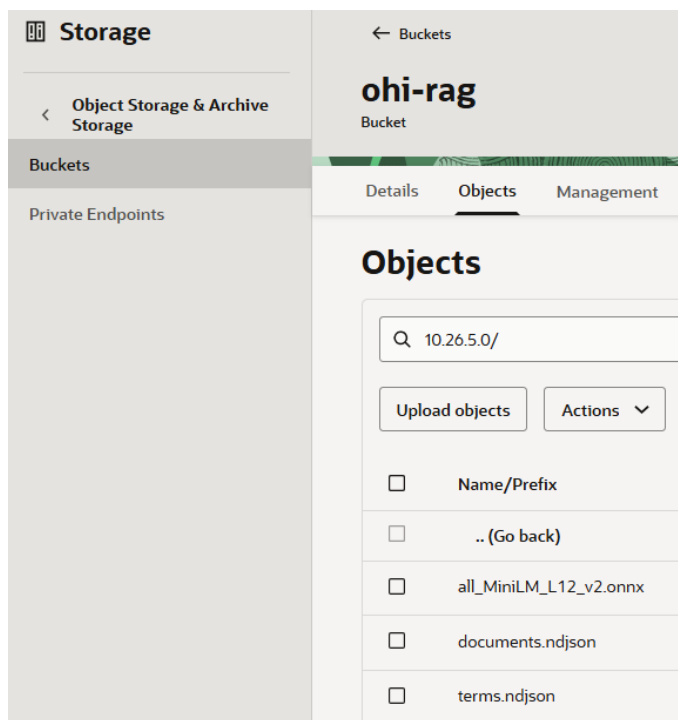
and unzip the downloaded file.

Bucket

Create a bucket and a release-specific prefix, for example ohi-rag/10.26.5.0/.

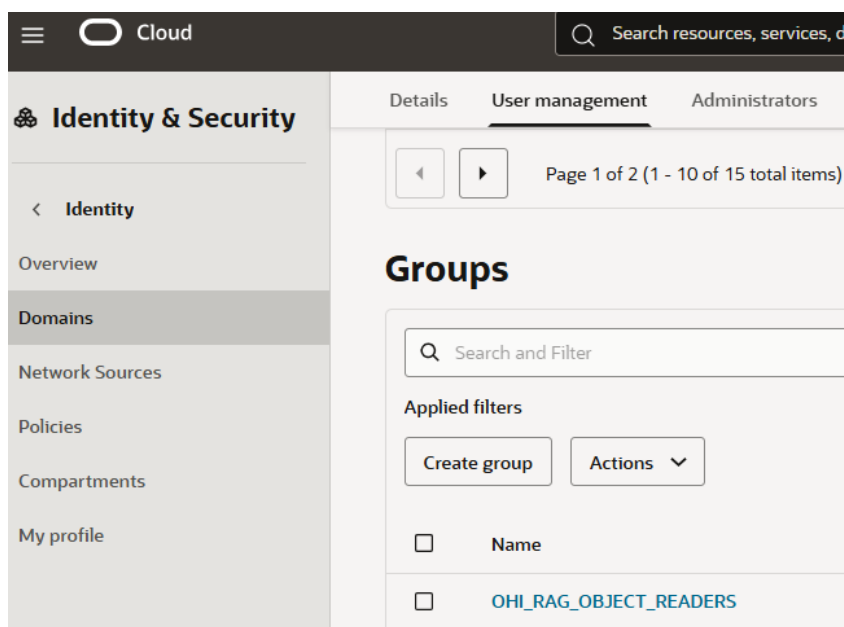
Upload `documents.ndjson` and `terms.ndjson` under the release-specific Object Storage prefix. Upload all_MiniLM_L12_v2.onnx under the same Object Storage prefix if the OHI_LLM_OWNER account has not yet been created or if the embedding model has not yet been loaded into that account. The Object Storage URI prefix supplied to the installer must end in "/". Do not use a pre-authenticated request (PAR) URL or embed credentials in the URI.

See the picture below for an example of such a bucket with relevant files:

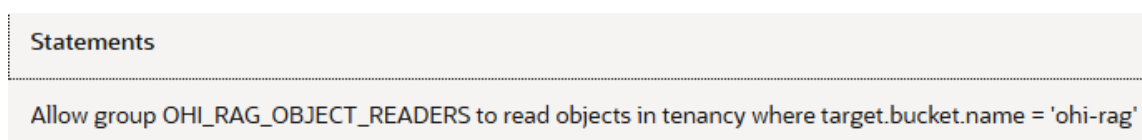


Grant, in the OCI console, the OCI user principal(s) as used by OHI_RAG_OCI_CRED and OHI_LLM_OCI_CRED (the OCI user principal you used when you created these credential entries in the database) read access only to this bucket or prefix. The credentials are database schema-specific (but might of course reference the same OCI user principal): the RAG data loaders use the OHI_AGENT_OWNER credential (default name OHI_RAG_OCI_CRED) and the model loader uses the OHI_LLM_OWNER (default name OHI_LLM_OCI_CRED) credential.

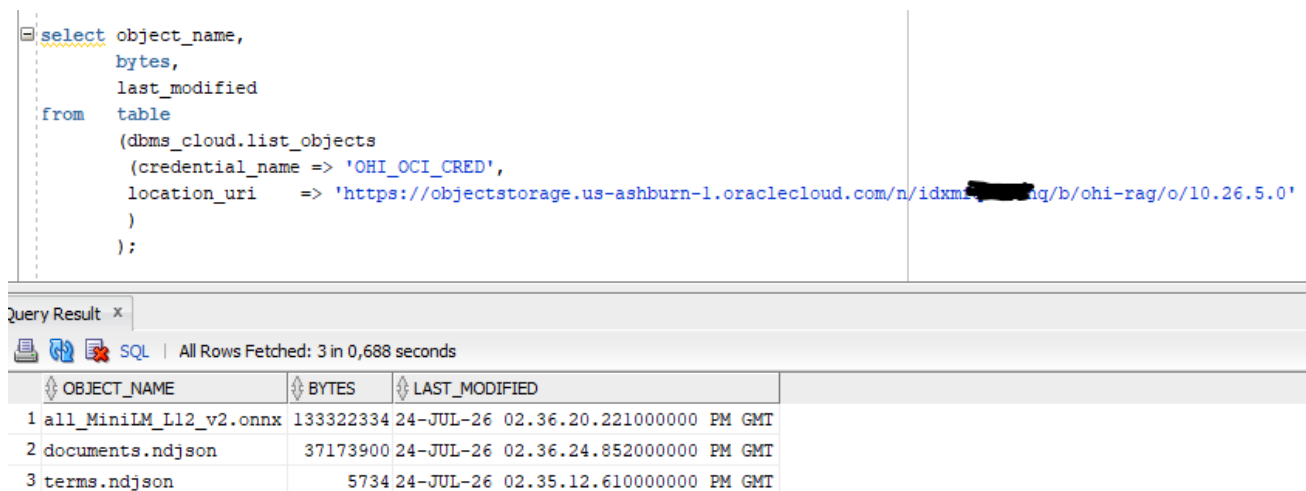
To provide the access you can create a group within your (default) domain under the User Management tab and grant this group access. Below, an example group is shown. Assign the relevant users/principals to the group.



After that, define a policy and add a statement that provides access to the bucket. An example statement is shown below, referencing the bucket name that was shown in the earlier screen print:



To test whether the bucket can be read you can use a similar SQL statement as below, using the OHI_AGENT_OWNER or OHI_LLM_OWNER account and the credential name you used (query it through select credential_name, username, comments, enabled from user_credentials):



```

select object_name,
       bytes,
       last_modified
from table
(dbms_cloud.list_objects
 (credential_name => 'OHI_OCI_CRED',
  location_uri    => 'https://objectstorage.us-ashburn-1.oraclecloud.com/n/idxm1-1q/b/ohi-rag/o/10.26.5.0'
 )
);

```

Query Result x

SQL | All Rows Fetched: 3 in 0,688 seconds

	OBJECT_NAME	BYTES	LAST_MODIFIED
1	all_MinilM_L12_v2.onnx	133322334	24-JUL-26 02.36.20.221000000 PM GMT
2	documents.ndjson	37173900	24-JUL-26 02.36.24.852000000 PM GMT
3	terms.ndjson	5734	24-JUL-26 02.35.12.610000000 PM GMT

Wallet aliases

Create Oracle wallet aliases and accompanying tnsnames aliases for the three OHI agent database accounts, OHI_AGENT_OWNER, OHI_AGENT_USER, OHI_LLM_OWNER.

The installer prompts for these aliases and validates that each alias connects to the expected database account.

The following naming convention is recommended:

```

/@@<environment>_ohi_agent_owner
/@@<environment>_ohi_agent_user
/@@<environment>_ohi_llm_owner

```

For example: /@@prod01_ohi_agent_owner

Including the environment name in the alias helps distinguish credentials across environments and reduces the risk of accidentally connecting to the wrong database.

For information on creating wallet aliases, refer to section 6.1.3.2 Wallet for OS Account BATCH of the "Oracle Health Insurance Back Office Installation, Configuration and DBA Manual". The same procedure can be used for these three accounts.

Make sure you configure your SQLcl or sqlplus environment so the (mTLS) Cloud Wallet can be used for the connection to the autonomous database.

For instructions please see:

<https://docs.oracle.com/en/cloud/paas/autonomous-database/serverless/adbsb/connect-tools.html>

When you create/add your TNS aliases in your tnsnames.ora you can choose whether to use a low, medium, high, tp or turgent service name. For the OHI_AGENT_OWNER and OHI_LLM_OWNER connections you may choose a medium or high service, depending on how important it is to speed up the OHI RAG object installation step of the RAG database. It can take up to about two hours on slower environments to load and process the LLM ONNX file and the NDJSON files. Of course, the speed also depends on the configuration of your ATP database (number of ECPU's, etc.).

For the OHI_AGENT_USER connection it is advised to use a tp or even turgent service connection.

Preparing the SQL scripts

Copy the content, and keep the directory structure, from the directory \$OZG_BASE/conf/ohi-agents to a location where SQL*Plus or SQLcl can read the scripts and can use the created wallet entries through the created aliases. Start the full installation from that directory so that relative @@ includes work correctly.

Running the installation

Before running the installer, enter the values for the target environment in OHI_AGENT_DB_INSTALL_PARAMETERS.sql. This file is required for every installation and is read automatically by OHI_AGENT_DB_INSTALL.sql.

If you install multiple environments, update OHI_AGENT_DB_INSTALL_PARAMETERS.sql with the correct values for each target environment before running OHI_AGENT_DB_INSTALL.sql. To keep the parameter sets separate, you can place a copy of the installation files in a separate directory for each target environment and maintain the parameter file in that directory.

The parameter file contains values for:

- Schema names for OHI_AGENT_OWNER, OHI_AGENT_USER and OHI_LLM_OWNER.
- Wallet aliases for these three users, each including "/@".
- The DBMS_CLOUD_AI profile name.
- The ONNX model file name, if the model must be loaded.
- The Object Storage URI prefix, ending in "/", and the credential names for OHI_AGENT_OWNER and OHI_LLM_OWNER.

Values indicated with <<...>> in the parameter file are placeholders and must be replaced with values appropriate for your installation. This notation is intended to make scripted deployment easier.

To skip loading the ONNX model, replace its placeholder with one space between the quotation marks:

```
define ohi_llm_onnx_file_name = " "
```

From the installation directory, connect as the ATP ADMIN user with SQL*Plus 19c or SQLcl. For example, with SQL*Plus:

```
sqlplus -L /@<ATP_ADMIN_WALLET_ALIAS>
SQL> @OHI_AGENT_DB_INSTALL.sql
```

When no ONNX file name is specified, the installer verifies and aligns access to the existing model without loading it again. The document loader replaces the existing document dataset, and the installer rebuilds the hybrid vector index.

To verify whether the embedding model is already installed, execute the following query using an account with DBA privileges:

```
select m.owner
,      m.model_name
from   dba_mining_models m
where  m.model_name = 'ALL_MINILM_L12_V2';
```

If the query returns a row for owner OHI_LLM_OWNER and model ALL_MINILM_L12_V2, the embedding model is already available and the ONNX file name can be omitted. This is typically the case when updating the NDJSON

files for a newer OHI release with newer Online Help information or when another agent schema set is already installed.

If no rows are returned, specify the ONNX file name so that the installer can load the model from the configured Object Storage prefix.

The picture below shows a snippet of the output.

```

Detecting input mode

1 row selected.

OCI Object Storage URI prefix (ending in /): https://objectstorage.us-ashburn-1.oraclecloud.com/n/10.26.5.0/o/ohi-rag/o/10.26.5.0/
Object Storage credential for OHI_RAG_OWNER [OHI_RAG_OCI_CRED]: OHI_OCI_CRED
Object Storage credential for OHI_LLM_OWNER [OHI_LLM_OCI_CRED]: OHI_OCI_CRED
Input mode: OBJECT_STORE

```

Installer activities

OHI_AGENT_DB_INSTALL.sql performs the following steps:

1. Validate the wallet aliases for OHI_AGENT_OWNER, OHI_AGENT_USER, and OHI_LLM_OWNER.
2. Install tables, Oracle Text settings, views, and packages in OHI_AGENT_OWNER.
3. Create or update the scheduler job OHI_RAG_TRACE_PURGE_JOB.
4. Compile the Select AI Agent packages from separate *.pks and *.pkb scripts.
5. Register the Select AI Agent definition through OHI_ONLINE_HELP_AGENT.sql.
6. Optionally load the ONNX model in OHI_LLM_OWNER, depending on the optional file name.
7. Align model access from OHI_LLM_OWNER to OHI_AGENT_OWNER.
8. Grant privileges to OHI_AGENT_USER
9. Load documents.ndjson through install/OHI_RAG_LOAD_DOCUMENTS_FROM_NDJSON.sql.
10. Load terms.ndjson through install/OHI_RAG_LOAD_TERMS_FROM_NDJSON.sql.
11. Build the HNSW index.

This last step can take quite some time, depending on the database environment you use. Some runs took around 2 hours to complete.

Post-installation validation

Before executing the validation queries below, review the installation spool file generated by the installer and verify that:

- No "ORA-", "SP2-", "PLS-" or "ERROR" messages are present.
- All expected verification messages reporting successful execution are present.
- The spool file confirms successful completion of:
 - Input-file retrieval and RAG data loading.
 - Embedding model access verification or embedding model loading.
 - Hybrid vector index creation.
 - Agent deployment.

Proceed with the validation steps only after the installation has completed successfully and the spool file confirms that all installation phases have been completed.

Connect as OHI_AGENT_OWNER before executing the validation queries. The validation is performed using the same schema that is used by the OHI RAG application and therefore verifies the installed functionality from an application perspective.

1. Verify that the embedding model can generate vectors and returns the expected vector dimensions:

```
select vector_dimension_count
      ( vector_embedding(OHI_LLM_OWNER.ALL_MINILM_L12_V2 USING 'smoke test' AS data)
      )      as embedding_dimensions
from dual;
```

The query should return a single row with a positive dimension count. For the 'ALL_MINILM_L12_V2' model, the expected value is typically "384".

2. Verify that the hybrid vector index can be used to retrieve documents from the OHI knowledge base:

```
select count(*) as result_count
from json_table
      ( dbms_hybrid_vector.search
      ( json
      ( '{ "hybrid_index_name": "OHI_DOC_HV_IND01"
        , "vector":
          { "search_text": "Virtual Private Database"
          , "search_mode": "DOCUMENT"
          , "result_max": 5
          }
        , "return":
          { "topN": 5
          , "values": ["rowid", "score"]
          }
        }'
      )
      )
      , '$[*]'
      columns (score number path '$.score')
      );
```

The query should return a value greater than zero, indicating that the vector index and indexed documentation content are available for retrieval.

3. Verify that the OCI credential, AI profile, network ACL configuration, and OCI Generative AI connectivity are functioning correctly:

```
select dbms_cloud_ai.generate
      ( prompt      => 'Reply with the word OK only.'
      , profile_name => 'OHI_OCI_PROFILE'
      , action       => 'chat'
      ) as response
FROM dual;
```

The query should return a response containing the word: "OK".

Trace retention

Tracing for the OHI Online Help AI assistant can be configured through the OHI Back Office parameter "Tracelevel" in the Agent parameter group. Valid values are NONE, ERROR, and INFO. When set to NONE, no records are written to table OHI_RAG_TRACE in the 26ai PDB.

The installer creates or updates the scheduler job OHI_RAG_TRACE_PURGE_JOB in the 26ai PDB. The job runs daily at 02:00 and purges records from OHI_RAG_TRACE for which TRACE_MOMENT is more than 30 days old.

OHI Back Office integration

After the OHI agent schemas have been created and the installation scripts have completed successfully, a database link must be created from the OHI Back Office schema to the chosen name for the OHI_AGENT_USER account in the Oracle Database 26ai ATP PDB hosting the OHI Online Help AI Assistant.

The OHI Back Office application uses this database link to access the functionality which is exposed through the OHI_AGENT_USER schema. Create the database link in the OHI Back Office schema using the naming conventions and connectivity settings applicable to the target environment.

Because the 26ai database is an ATP PDB, you may want to specify the cloud wallet and connection details in your create database link command:

```
create database link <26aiPDBname>_ohi_agent_user
  connect to ohi_agent_user identified by "<pwd>"
  using
'(description=
  (retry_count ... <copy connection details from your tp or tपुरgent definition
  as stored in the tnsnames.ora in the Cloud Wallet zip> ...
  )';
```

After the database link has been created, configure the OHI Back Office parameter Agent Database Link in the Agent parameter group with the name of the database link.

This parameter is used by the OHI Online Help AI Assistant integration to locate and access the remote OHI_AGENT_USER schema.

After creating or modifying the database link and configuring the Agent Database Link parameter, execute the following command while connected as the OHI Back Office schema owner, or through a DBA account with sufficient privileges:

```
BEGIN
  sys_install_agent_pck.run;
END;
/
```

This step completes the integration between OHI Back Office and the OHI Online Help AI Assistant.

To test the OHI Online Help AI Assistant, start the OHIJET application and use the AI icon in the upper right corner of the browser window:



Ask what the agent can do for you and verify that you receive a relevant response.

The icon can only be used if you are logged in in the OHI Back Office OHIJET application with an account that has received access to the AI Chat Client module (module code SYS1160J).

Installing a higher major release of the Online Help data

After an initial installation of the OHI Online Help AI Assistant, the Online Help data needs to be refreshed for each major release of OHI Back Office.

OHI Back Office does not need to be placed in or kept in restricted mode during this process. Until the Online Help AI Assistant for the new release has been installed, ALG_AGENT_CALLOUT_PCK may remain invalid and the assistant may be unavailable.

This is done by executing the following steps:

- Copy the new versions of the files `documents.ndjson` and `terms.ndjson` from the directory `"$OZG_BASE/help/nl_nl/Output/OHIHelp/RAG"` to your bucket in OCI.
- Prepare the SQL scripts as described above.
- Run the installation as described above.
- Perform the post-installation validation as described above.

Cloning an OHI Back Office environment

After cloning an OHI Back Office environment, review the database link used by the OHI Online Help AI Assistant. Only one OHI Back Office environment may connect to a given `OHI_AGENT_USER` account.

In the cloned OHI Back Office database, identify the database link and check its owner, remote username and target connect string in `DBA_DB_LINKS`.

If the cloned environment requires the assistant:

- Provision a separate `OHI_AGENT_USER` account for the cloned environment, using the installation instructions above.
- Check that the target Online Help installation matches the major release of the cloned OHI Back Office environment. Select a different target PDB if necessary.
- Drop and recreate the database link using the new `OHI_AGENT_USER` account and its credentials, and the connect string for the selected target PDB.
- Check the link again in `DBA_DB_LINKS` to confirm the remote username and target connect string. Test the connection through the link and perform the assistant validation described above.

If the cloned environment must not have access to the assistant, drop the database link and verify that it is no longer present.

Complete this review before using the assistant in the cloned environment. Otherwise, it may connect to the AI Assistant environment used by the source environment.

Patching an OHI Back Office environment

If you patch an OHI Back Office environment to a higher major release, you should determine:

- if you need to update the Online Help data in the linked database 26ai ATP PDB schema (see above), or
- if you need to point the database link to another (unused) specific `OHI_AGENT_USER` schema with Online Help data of the same OHI major release, or
- if you want to disable the OHI Online Help AI Assistant by removing the database link.

It is recommended to install or upgrade the agent software for the target major release before patching OHI Back Office. To reduce downtime for the assistant, consider installing the corresponding Online Help data in a new `OHI_AGENT_OWNER` schema in advance. At the end of the regular patch activities, switch the database link to the new `OHI_AGENT_USER`.

Copyright © 2026, Oracle and/or its affiliates. This document is provided for information purposes only, and the contents hereof are subject to change without notice. This document is not warranted to be error-free, nor subject to any other warranties or conditions, whether expressed orally or implied in law, including implied warranties and conditions of merchantability or fitness for a particular purpose. We specifically disclaim any liability with respect to this document, and no contractual obligations are formed either directly or indirectly by this document. This document may not be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without our prior written permission.

Oracle, Java, MySQL, and NetSuite are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Some regulatory certifications or registrations to products or services referenced on this website are held by Cerner Corporation. Cerner Corporation is a wholly-owned subsidiary of Oracle. Cerner Corporation is an ONC-certified health IT developer and a registered medical device manufacturer in the United States and other jurisdictions worldwide.

This document may include some forward-looking content for illustrative purposes only. Some products and features discussed are indicative of the products and features of a prospective future launch in the United States only or elsewhere. Not all products and features discussed are currently offered for sale in the United States or elsewhere. Products and features of the actual offering may differ from those discussed in this document and may vary from country to country. Any timelines contained in this document are indicative only. Timelines and product features may depend on regulatory approvals or certification for individual products or features in the applicable country or region.