Migration Portal

Release 4.4.0

EnterpriseDB Corporation

Mar 06, 2023

CONTENTS

1 Release notes **3**
- 1.1 Version 4.0 . . . 3
- 1.2 Version 4.1 . . . 4
- 1.3 Version 4.2 . . . 5
- 1.4 Version 4.3 . . . 6
- 1.5 Version 4.4 . . . 7
  - 1.5.1 End of support notice . . . 8

2 Supported platforms **9**
- 2.1 Supported browsers . . . 9

3 Using Migration Portal **11**
- 3.1 Overview of the Migration Portal home page . . . 12
- 3.2 Overview of the Migration Portal Projects page . . . 13
- 3.3 Migration Portal quick start . . . 14

4 Migrating a database **17**
- 4.1 Performing a schema extraction . . . 17
  - 4.1.1 Extracting schemas using the EDB DDL Extractor . . . 17
  - 4.1.2 Schemas and objects support . . . 19
- 4.2 Performing a schema assessment . . . 24
  - 4.2.1 Perform a schema assessment . . . 24
  - 4.2.2 Generate an assessment report . . . 27
  - 4.2.3 Evaluate an assessment report . . . 28
- 4.3 Performing a schema migration . . . 29
  - 4.3.1 Offline migration . . . 29
  - 4.3.2 Online migration . . . 31
- 4.4 Performing a data migration . . . 34
  - 4.4.1 Advanced data migration . . . 34

5 Known issues, limitations, and notes **35**
- 5.1 Migration Portal . . . 35
  - 5.1.1 Wrapped objects . . . 35
  - 5.1.2 File encoding . . . 35
  - 5.1.3 ALTER statements . . . 35
  - 5.1.4 Users, roles, profiles, and grants . . . 36
  - 5.1.5 Grants . . . 36
  - 5.1.6 Privileges . . . 36
  - 5.1.7 Passwords . . . 36
  - 5.1.8 Profiles . . . 36

5.1.9 Using Oracle default case 37
5.2 EDB DDL Extractor 40
5.3 Oracle Data Pump utilities 40

Migration Portal is a web-based service for migrating Oracle database schemas to the EDB Postgres Advanced Server platform. The Migration Portal assesses and analyzes Oracle database schemas and converts:

- Tables
- Sequences
- Constraints
- Indexes (Except LOB indexes and indexes on materialized views)
- Synonyms
- Views
- Materialized views
- DB links
- Types and type body
- Triggers
- Functions
- Procedures
- Packages
- Users
- Roles
- Profiles
- Role and object grants

The Migration Portal produces data definition language (DDL) statements that are compatible with EDB Postgres Advanced Server.

The user-friendly portal interface simplifies the assessment and migration process. Log in to the portal and start the migration process .

EnterpriseDB has helped companies migrate their existing database systems to Postgres for years. For more information, see the

EnterpriseDB website .

CHAPTER

ONE

RELEASE NOTES

The Migration Portal documentation describes the latest version of Migration Portal 4.0.

Version	Release date
Version 4.4	14 Feb 2023
Version 4.3	08 Dec 2022
Version 4.2	04 Oct 2022
Version 4.1	04 Aug 2022
Version 4.0	22 Apr 2022

1.1 Version 4.0

New features, enhancements, bug fixes, and other changes in Migration Portal 4.0 include the following:

Type	Description
Enhancement	The Migration Portal now supports the use of the EDB DDL Extractor again. Issues related to the DDL Extractor are resolved and the temporary restriction introduced on the use of DDL Extractor file format has been removed. The Migration Portal continues to support the use of the Oracle Data Pump utilities following the instructions in the documentation; however, the use of the EDB DDL Extractor is recommended.
Enhancement	The schema assessment algorithms are optimized to address a few performance related issues and to reduce overall schema assessment time.
Enhancement	Added a repair handler to remove CACHE/NOCACHE keyword from TABLE definition as EDB Postgres Advanced Server does not support it.
Security fix	The jackson-databind and h2 libraries are upgraded to the latest versions to address security issues (CVE-2020-36518, CVE-2022-23221, CVE-2021-42392 and CVE-2021-23463).
Bug fix	Fixed the grammar rule to resolve a parsing error that skipped execution of a repair handler to remove EDITIONABLE keywords.
Bug fix	Fixed the repair handler to perform proper conversion of Oracle PIPELINED and PIPE ROW function calls.
Bug fix	Fixed an issue where the case sensitivity was lost when an object name was based on Unicode characters.
Bug fix	Fixed the issue that displays the Getting Started wizard when Migration Portal landing page is opened for the first time for a new user.
Bug fix	Fixed the issue where a VIEW definition was truncated during transformation in the target EPAS DDL script.

continues on next page

Table 2 – continued from previous page

Type	Description
Bug fix	Previously, uploaded DDL files having characters that were not compatible with UTF-8 were failing at the start of the Assessment phase. Migration Portal now reads the SQL files as UTF-8. In case a non UTF-8 file is uploaded, all the characters that are not compatible with UTF-8 will be converted to the replacement character and assessment will continue without failing.
Bug fix	Fixed constraint name assignment for the unnamed constraint in the source Oracle DDL.
Bug fix	Updated the repair handler to remove the OIDINDEX storage clause for Oracle Data Pump generated source DDL files.
Bug fix	Allowed the COLLATE and COMPUTATION keywords to be used as identifiers.
Bug fix	Fixed the handling of multi-line comments that appear as part of single line comments.
Bug fix	Fixed the grammar to properly identify and skip wrapped objects.
Bug fix	Fixed ConcurrentModificationException occurred while running multiple projects simultaneously.
Bug fix	Fixed unique object names handling across multiple schemas. The object names are not required to be unique across different schemas.
Bug fix	Made some fixes in the create table grammar to ignore the order of some clauses as no specific order is required.

1.2 Version 4.1

New features, enhancements, bug fixes, and other changes in Migration Portal 4.1 include the following:

Type	Description
Enhancement	Migration Portal Schema Assessment Reports now include a new Feature Analysis section that provides a migration complexity score based on the effort type and complexity level relevant to the Oracle constructs (in other words, features) found in the uploaded DDL. This new section also provides a summary of the incompatible features found in the DDL. Note that the Feature Analysis section is currently only generated for new migration projects and not for existing projects that had been created prior to this release.
Enhancement	The user can now apply filtering to the available schemas and underlying objects in a project to generate target database DDL for a subset of selected object types. Additionally, instead of a single DDL SQL file being created for download, a zip archive is created, which contains separate files for each selected object type, as well as consolidated scripts for the project and the selected schemas.
Enhancement	Added a new repair handler, ERH-3005, to transform Q quoted strings to either dollar quoted or E quoted strings.
Enhancement	Enhanced the ERH-2078 repair handler to apply pipelined specific transformations at the package header level.
Enhancement	Updated the ERH-2034 repair handler so that it removes NULL from the TYPE definition provided inside a PACKAGE.
Security fix	Upgraded the async, eventsource, h2, jsdom, and moment libraries to the latest versions to address security issues (CVE-2021-43138, CVE-2022-1650, CVE-2022-23221, CVE-2021-42392, CVE-2021-20066, and CVE-2022-31129).
Bug fix	Fixed a corner case that resulted in the skipping of some indexes when processing source Oracle DDL files generated using the Oracle Data Pump utilities.

continues on next page

Table 3 – continued from previous page

Type	Description
Bug fix	Fixed the behavior where only the ENABLE keyword was being removed from the ENABLE NOVALIDATE clause in ALTER TABLE statements so that the complete clause is now removed. Also, updated the relevant repair handlers ERH-2003 and ERH-2005.
Bug fix	Fixed a synchronization thread issue that was impacting the performance of other assessments running in parallel with a long-running schema assessment.
Bug fix	Fixed an issue where the DEDUPLICATE LOB clause in CREATE TABLE statements that are generated by the Oracle Data Pump utilities resulted in DDL parsing related errors.

1.3 Version 4.2

New features, enhancements, bug fixes, and other changes in Migration Portal 4.2 include the following:

Type	Description
Enhancement	Added support for calculating a schema migration complexity score and generating a feature analysis for existing projects that were created prior to the 4.1.0 release.
Enhancement	Enhanced the schema migration complexity scoring capability to accommodate updating the score after manually repairing objects.
Enhancement	Added a new "Unresolved Objects" table in the Feature Analysis section of the schema assessment report that lists objects with unresolved errors and the list of incompatible Oracle features used in those objects.
Enhancement	Improved the performance of the schema assessment report generation by up to 3x for projects that contain a large number of Oracle features.
Bug Fix	Fixed the Migration Portal to capture and more accurately report the use of incompatible built-in package and SYS schema TYPE objects.
Bug Fix	Fixed the Migration Portal to more accurately capture the presence of the following Oracle features and assess their impact on schema migration complexity: USER DEFINED EXCEPTION RISING, USER DEFINED EXCEPTION HANDLING, XMLAGG, XMLQUERY, XMLTABLE, XMLPI, XMLPARSE, and XMLROOT.
Bug Fix	Created a new version of the DDL Extractor to correct an issue that results in duplicate entries of Oracle tables defined with the CACHE keyword being emitted to the DDL file. The Migration Portal has been updated to provide links to the new version of the DDL Extractor. The Migration Portal is still able to process and assess DDL generated by the older version of the DDL Extractor.
Bug Fix	Fixed the issue where the PRAGMA EXCEPTION_INIT declaration is incorrectly reported as an incompatible feature in the Schema Assessment Report "Feature Analysis" section.
Bug Fix	Fixed the issue where LONG RAW is incorrectly reported as an incompatible feature under the "Feature Analysis" section.
Bug Fix	Fixed the issue where a user-defined exception declaration is incorrectly reported as an incompatible feature in the Schema Assessment Report "Feature Analysis" section.
Bug Fix	Fixed the issue where the BULK COLLECT INTO clause is incorrectly reported as an incompatible feature in the Schema Assessment Report "Feature Analysis" section.
Bug Fix	Fixed the issue where NULLIF function is incorrectly reported as an incompatible feature in the Schema Assessment Report "Feature Analysis" section.

continues on next page

Table 4 – continued from previous page

Type	Description
Bug Fix	Updated the workaround status of the REGEXP_LIKE clause from "None / Needs Investigation" to "Exists" in the Schema Assessment Report "Feature Analysis" section. A repair handler is currently available and automatically applied for the REGEXP_LIKE clause.
Bug Fix	Updated the list of Oracle predefined PL/SQL exceptions that are supported and not supported by EDB Postgres Advanced Server.
Bug Fix	Fixed the issue where the user selection state on the target DDL object selection page is lost if the user opts for a subset of objects (schemas, tables, views etc.) and moves across wizard pages.

1.4 Version 4.3

New features, enhancements, bug fixes, and other changes in Migration Portal 4.3 include the following:

Type	Description
Enhancement	The Migration Portal has been enhanced to support the assessment and migration of users, roles, profiles and grants from the source Oracle database.
Enhancement	The Migration Portal has been updated to provide links to the new version of the EDB DDL Extractor that provides support for extracting users, roles, profiles and grants. The Migration Portal is still able to process and assess DDL generated by the older version of the EDB DDL Extractor; however, the newly supported user related information cannot be extracted by those older versions.
Enhancement	The Migration Portal schema migration complexity scoring and feature analysis capability has been enhanced to provide a score and feature analysis that is based on the selected target EDB Postgres Advanced Server version and not only on version 14 as was the case in the prior releases.
Enhancement	The Migration Portal user interface for migrating schemas to a target database has been simplified to enable offline and online migration modes with a reduced number of steps and redundant options.
Enhancement	The Migration Portal migration complexity and feature analysis capability has been enhanced to capture and report the presence of Oracle features that had previously not been being captured, including the following features: *CONNECT BY CONDITION*, *CONNECT BY NOCYCLE*, and Built-in/SYS owned Function *GROUP_ID*.
Security fix	The jackson-databind library used by the Migration Portal has been updated to address the security vulnerabilities identified in CVE-2022-42003 and CVE-2022-42004.
Bug fix	Fixed the handling of *MAXVALUE* for sequences to avoid warnings and also to resolve the issue with the maximum value allowed for *MAXVALUE* supported in the target EDB Postgres Advanced Server database.
Bug fix	Fixed an issue in the handling of *ALTER TABLE MODIFY* statements related to virtual table columns that are found in the extracted DDL generated using the Oracle Data Pump utility.
Bug fix	Fixed an issue where the *PIPELINED* clause feature is not reported when the clause is used within a package function definition.
Bug fix	Fixed an issue related to catalog views not being detected and reported in compatible features in certain cases.

1.5 Version 4.4

New features, enhancements, bug fixes, and other changes in Migration Portal 4.4 include the following:

Type	Description
Enhancement	The Migration Portal now supports EDB Postgres Advanced Server version 15, the latest major version of EDB Postgres Advanced Server.
Enhancement	The Migration Portal has been enhanced to provide the option to preserve the Oracle default case for object names in a migration project. In Postgres, the default case for object names is lowercase. In Oracle, it is uppercase. In most cases, the Postgres default case should be used when migrating a database to Postgres; however, this new option is being provided to better support those applications that specifically use quoted uppercase identifiers in their database queries. Note that while the new capability enables all objects to be created with quoted uppercase names, the initial implementation does not update object name references specified in procedural code.
Enhancement	The Migration Portal migration complexity and feature analysis capability has been enhanced to capture and report the presence of certain Oracle features that had previously not been captured, including the *OUTER JOIN WITH BETWEEN* feature, which corresponds to the use of a *BETWEEN* clause as a condition of an outer join in a query.
Enhancement	The Migration Portal has been enhanced with the new ERH-3008 repair handler which removes *CONSTRAINT* clause(s) from *CREATE VIEW* statements because this feature is not supported in EDB Postgres Advanced Server. As part of this enhancement, the schema assessment report Feature Analysis section related capabilities have been updated to identify and report *VIEW COLUMN CONSTRAINT* occurrences with a categorization of 'Features with existing available workaround'.
Enhancement	The ERH-2057 repair handler has been enhanced to remove the precision part from *LONG* data type.
Bug fix	The EDB DDL Extractor has been updated to extract *ROLE* to *ROLE* grants that were not previously being fetched.
Bug fix	Fixed an issue where the DDL for a *VIEW* was being truncated after the *DATE* keyword, when used to specify a date literal.
Bug fix	Fixed an issue that resulted in the removal of the *DEFAULT* clause on table columns that are specified with *EXTENDED* as the column name.
Bug fix	Fixed an issue where an incorrect object count was being shown in the assessment report, after a user manually repairs an object.
Bug fix	Fixed an issue with the ERH-2059 repair handler where the schema name was incorrectly being removed from row type objects specified with a schema name prefix after converting from the Oracle *PIPE ROW* syntax to the Postgres *RETURN NEXT* syntax.
Other	Added a Migration Portal Knowledge Base entry with a proposed workaround for migrating *TO_TIMESTAMP_TZ()* function calls with a single argument, which are not supported in EDB Postgres Advanced Server.
Other	Support for EDB Postgres Advanced Server version 10 has been removed. Users can no longer select EDB Postgres Advanced Server 10 as a target database for new migration projects. Users are also no longer able to upload new schemas and edit and reassess objects in existing EDB Postgres Advanced Server 10 projects.

1.5.1 End of support notice

Since the end of standard support for EDB Postgres Advanced Server (EPAS) version 11 is scheduled for November 20, 2023, more recent versions of EPAS that will be supported beyond 2023 should be selected for new Migration Portal projects. Support for EPAS 11 is planned to begin being removed from the Migration Portal on May 20, 2023 to help ensure that no new EPAS 11 projects will be created after this date.

To maximize the length of available support and to take advantage of new features, particularly those that provide additional compatibility with Oracle, EDB recommends selecting the latest EPAS version as the target of a Migration Portal project. See the EDB Platform Compatibility page for more information on the support periods for each EDB Postgres Advanced Server major version.

CHAPTER

TWO

SUPPORTED PLATFORMS

The Migration Portal supports assessment and migration from Oracle 11g, 12c, 18c, and 19c to EDB Postgres Advanced Server 11, 12, 13, or 14. Migration Portal is supported on the following browsers.

2.1 Supported browsers

For the best user experience, we recommend using the Google Chrome browser. Migration Portal is also supported on the following browsers:

Browser	Supported version
Apple Safari on Macintosh OS	14 and above
Google Chrome	68 and above
Microsoft Edge	42 and above
Mozilla Firefox	60 and above

CHAPTER

THREE

USING MIGRATION PORTAL

Migration Portal allows you to easily migrate your database schema from Oracle to EDB Postgres Advanced Server. You can upload schemas for assessment and get immediate feedback and suggestions. The portal allows you to download assessed DDLs for all objects and create your EDB Postgres database on premises or in the cloud.

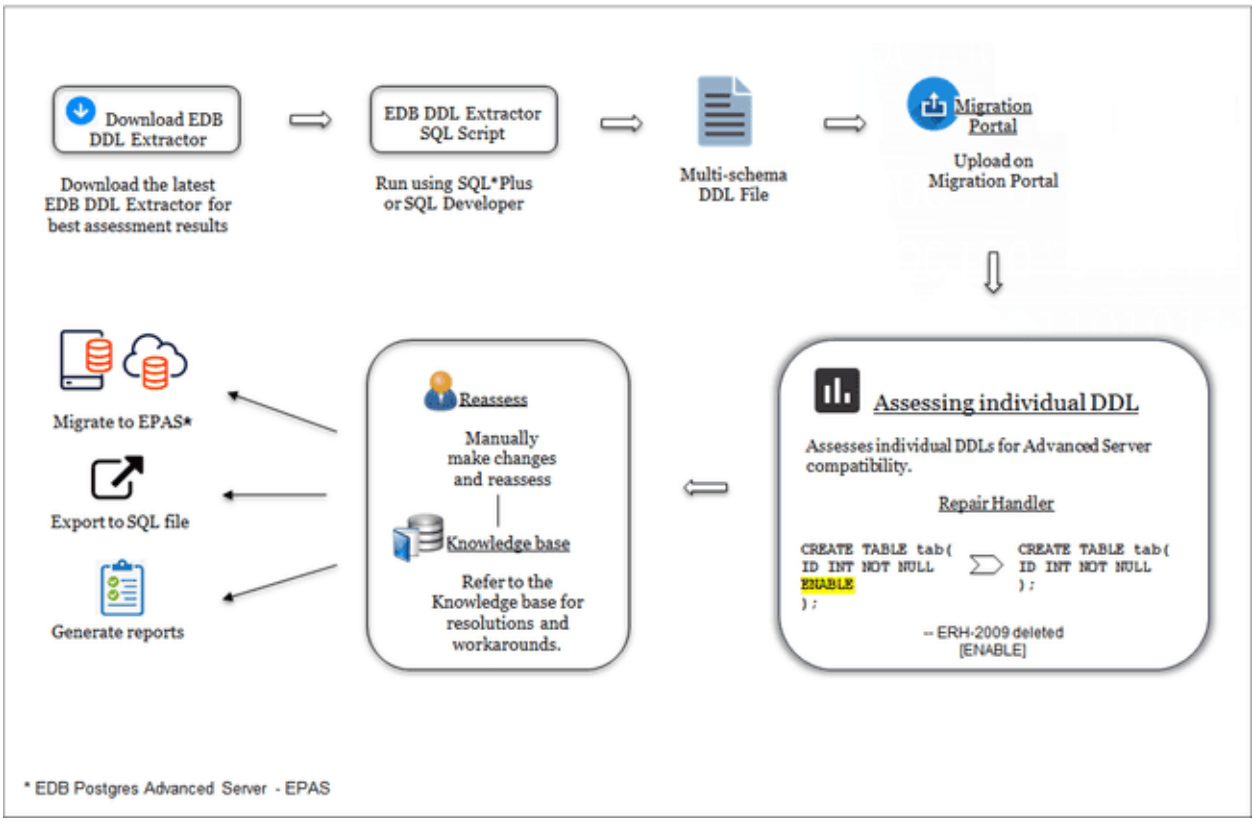


Fig. 1: The assessment and migration process

Note: Migration Portal currently doesn't have team collaboration features. We recommend collaborative meetings, screen sharing, and using the downloadable assessment report to share results.

To access the Migration Portal:

1. Open a browser and navigate to https://www.enterprisedb.com/.
2. On the EnterpriseDB home page, select **PostgreSQLSoftware>Migration** .

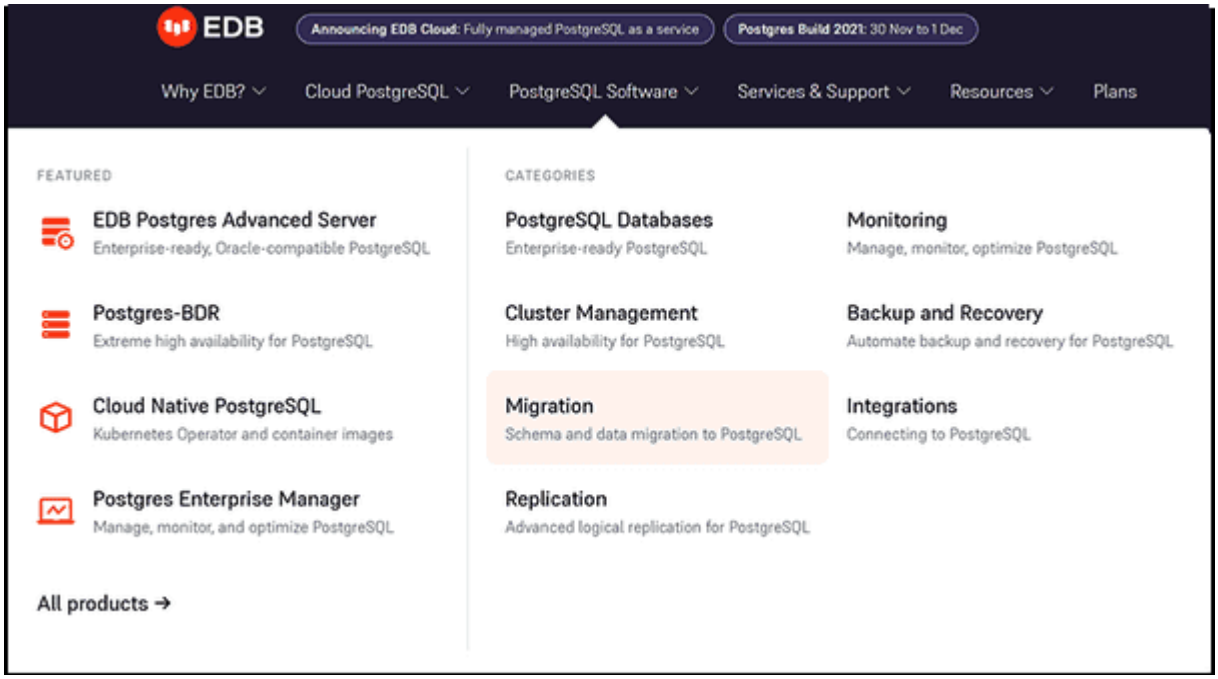


Fig. 2: Migration Portal on the EnterpriseDB home page

3. On the Migration Portal page, select **Findoutmore>OpenMigrationPortal** .
4. Log in using your credentials. See *Migration Portal quick start* for an overview of using Migration Portal to migrate Oracle schemas.

3.1 Overview of the Migration Portal home page

The Migration Portal home page provides quick access to migration tools.

The Migration Portal home page allows access to the following Migration Portal features:

- **Projects** : The Projects panel displays a list of assessed projects.
- **Createproject** : Select **+New** to create a project.
- **Searchprojects** : Use the **Search** box to search for projects.
- **Overview** : The Overview panel provides details about the selected project and displays the compatibility percentage after schema assessment.
- **Migrateto** : Select **Migrateto** either to download an EDB Postgres Advanced Server compatible `.sql` file or migrate a schema to an EDB Postgres Advanced Server on premises or in the cloud. For more information, see *Performing a schema migration* .
- **Report** : Select **Report** to view and download the schema assessment report.
- **Delete** : Select **Delete** to delete a selected project.
- **Warningsign** : A warning message appears if a project or a schema is less than 70% compatible or any DDL doesn't succeed after multiple attempts.
- **UploadDDLfile** : Select **Uploadnewschema** to upload a new DDL file.
- **Schemas** : The Schemas panel displays the assessment result from an uploaded DDL file.

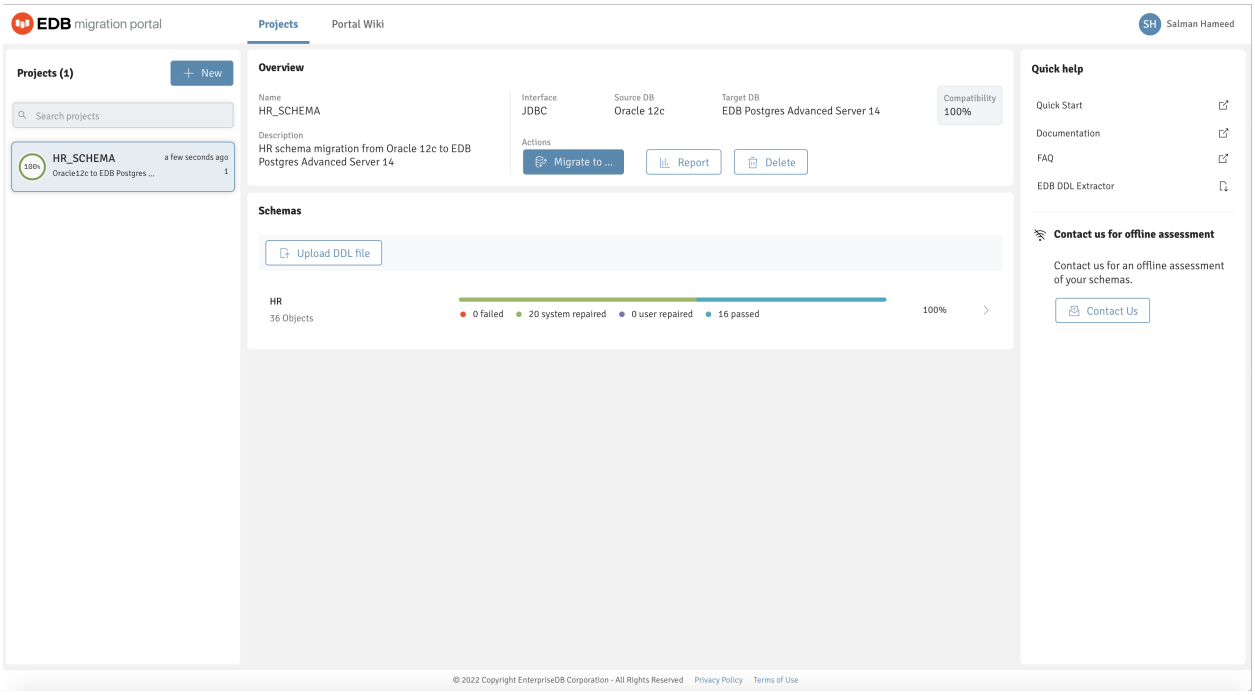


Fig. 3: The Migration Portal home page.

- **Quickhelp** : The Quick help panel contains quick links to a few required resources.
- **PortalWiki** : Select **PortalWiki** to access links to product information and more help guides.

3.2 Overview of the Migration Portal Projects page

The Migration Portal Projects page provides detailed information about your migration.

Use the following resources to gather information about your migration projects:

- **Compatible** : The **Compatible** gauge displays the color on the basis of the compatibility percentage of the assessed schema.
- **SchemaCount** : Displays the number of schemas in a project.
- **Migrateto** : Select **Migrateto** to migrate the schema to an EDB Postgres Advanced Server on premises or in the cloud.
- **Searchobjects** : Use the **Search** box to search for objects.
- **Filters** : From the left panel of the Projects page, you can filter the system repaired and manual repaired objects. You can select one or more filter combinations to refine the information.
- **Objects** : Displays the objects for the selected schemas.
- **CommonFailures** : Displays the reason for the failed objects for the selected schemas.

Note: You can download a *.csv* file for the common failures for the project.

- **Schema** : Displays the result of the assessment.

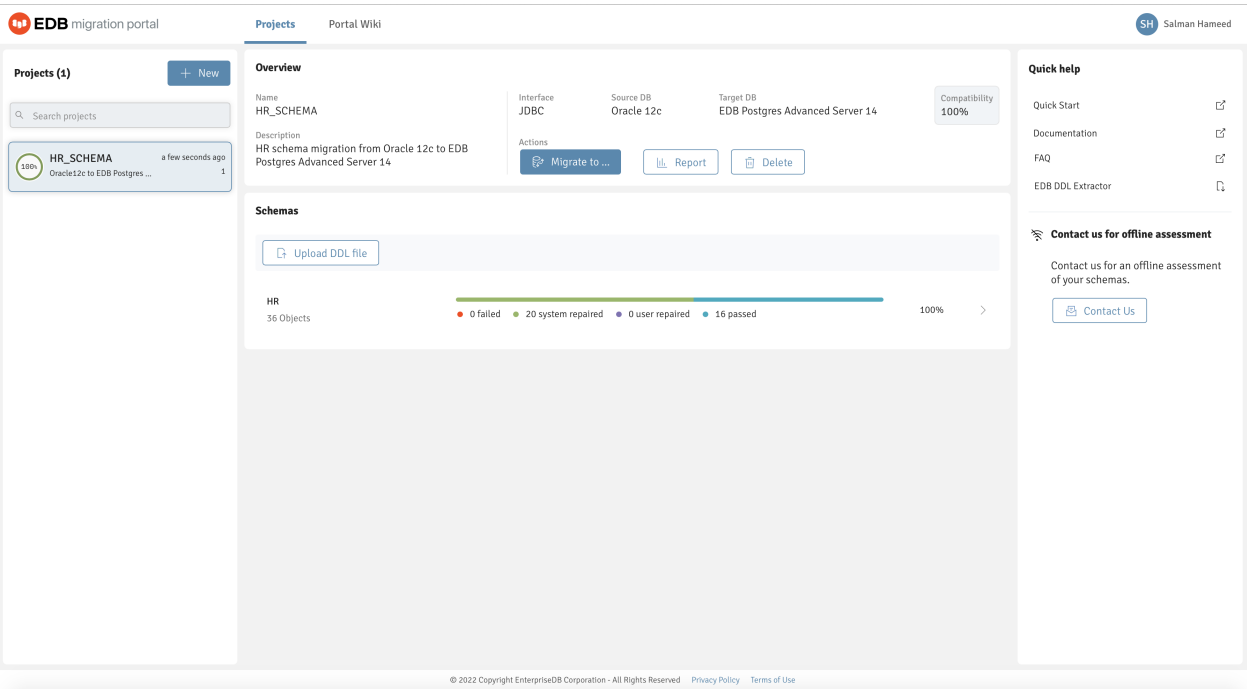


Fig. 4: The Migration Portal Projects Page overview.

- **Tooltip** : Hover over a result set to display a tooltip with the number of passed, failed, and repaired objects.
- **Quickhelp** : The Quick help panel displays links to Knowledge Base articles and repair handler documentation.
- **Search** : Use the **Search** box to search the Knowledge Base entries or repair handler documentation for specific information.

3.3 Migration Portal quick start

To migrate Oracle schemas using Migration Portal:

1. Generate a `.sql` file with metadata information of your schemas. For more information, see *Performing a schema extraction* .
2. Log in to *Using Migration Portal* with your EDB credentials.
3. Create a new project.
4. Upload the `.sql` file with your schema metadata.

Note: Uploading large DDL files might freeze or crash the browser (depending on the browser and the system resources in use). In this scenario, try running Migration Portal in a different browser or extracting fewer schemas and uploading that file into Migration Portal.

1. Select **Create&assess** . Migration Portal reviews every construct, executes repair actions to improve compatibility with EDB Postgres Advanced Server, and flags any remaining errors that require manual intervention.
2. Verify the DDL objects, for example, TABLES that don't show a 100% success ratio.
3. Select the objects that aren't compatible with EDB Postgres Advanced Server.

4. Refer to the Knowledge Base for checking the possible workarounds for the objects that aren't compatible in EDB Postgres Advanced Server.
5. Make the required changes and select **Reassess** to rerun the analysis.
6. After resolving errors in your schemas, you can download the modified EDB compatible DDL file and use any client application such as pgAdmin or PSQL client and migrate the schema to EDB Postgres Advanced Server. You can also migrate to EDB Postgres Advanced Server on the cloud. For more information, see *Performing a schema migration* .

CHAPTER

FOUR

MIGRATING A DATABASE

To migrate a database, you must complete the following steps:

1. Perform a *Performing a schema extraction* .
2. Perform a *Performing a schema assessment* .
3. Perform a *Performing a schema migration* .
4. Perform a *Performing a data migration* .

4.1 Performing a schema extraction

You can perform a schema extraction using either of the following methods. EDB recommends using the EDB DDL extractor to extract your schemas.

- *Extracting schemas using the EDB DDL Extractor* (recommended method)
- *Extracting schemas using Oracle Data Pump utilities*

For more information, see *Known issues, limitations, and notes* .

4.1.1 Extracting schemas using the EDB DDL Extractor

Download the latest EDB DDL Extractor script from the Migration Portal Projects page.

Note: Migration Portal might become unresponsive for very large SQL files, depending on your system and browser resource availability. To resolve this, try extracting fewer schemas at a time to reduce the SQL file size.

Prerequisites

You can run the EDB DDL Extractor script in SQL Developer or SQL*Plus. It uses Oracle's `DBMS_METADATA` built-in package to extract DDLs for different objects under schemas (specified while running the script). The EDB DDL Extractor creates the DDL file uploaded to the portal and analyzed for EDB Postgres Advanced Server compatibility.

Note: You must have *CONNECT* and *SELECT_CATALOG_ROLE* roles and *CREATE TABLE* privilege.

For SQL*Plus

1. Connect to SQL*Plus and run this command:

```
@edb_ddl_extractor.sql
```

2. When prompted, if the Oracle user that is running the script has the required privileges, press **Enter** to continue running the script.
3. When prompted, provide the schema name. When extracting multiple schemas, use a comma (`','`) as a delimiter.

Note: If you want to extract all the user schemas from the current database, don't mention any schema names while extracting. However, we recommend mentioning the schema names that you want to extract.

4. When prompted, enter the path or directory for the extractor to store the extracted DDL. If you don't specify a path, the extracted DDL is output to the directory where you ran the SQL*Plus command.
5. When prompted, enter `yes` or `no` depending on whether you want to extract dependent objects from other schemas.

For example:

1. Run the extractor script:

```
@edb_ddl_extractor.sql
```

2. Press **Enter**
3. Specify three schemas by separating them with commas:

```
Enter a comma separated list of schemas to be extracted (Default all schemas): HR, SCOTT,
↪ FINANCE
```

4. Enter a path for the output file:

On Linux, you might use:

```
Location for output file (Default current location) : /home/oracle/extracted_ddls/
```

Similarly on Windows:

```
Location for output file (Default current location) : c:\Users\Example\Desktop\
```

5. Enter `yes` to extract dependent objects in other schemas:

```
WARNING:

Given schema(s) list may contain objects which are dependent on objects
from other schema(s), not mentioned in the list.` `Assessment may fail
for such objects. It is suggested to extract all dependent objects
together.
Extract dependent object from other schemas?(yes/no) (Default no / Ignored for all␣
↪schemas option): yes
```

For SQL Developer

After loading the `edb_ddl_extractor.sql` script into SQL Developer and connecting to the source Oracle database, run the script. As the script executes, respond to the prompts.

1. If the user for the database connection has the required privileges as listed in the **ScriptOutput** tab, select **Yes** to continue running the script.
2. Enter a comma-separated list of schemas and select **OK** .
3. Enter the path for the output file and select **OK** . The default is the location of the DDL Extractor script.
4. Enter `yes` or `no` to extract dependent objects and select **OK** . The default is to not extract dependent objects.

Output of the DDL Extractor run appears in the Script Output tab. The name of the output file appears after the `Extraction Completed` message in the script output.

Note: You can also enter single schema name in both SQL*Plus and SQL Developer.

The script then iterates through the object types in the source database. Once the task is completed, the `.SQL` output is stored at the location you entered (e.g., `c:\Users\Example\Desktop\`).

See *File encoding* for information about the file encoding format expected by Migration Portal.

4.1.2 Schemas and objects support

The lists and tables that follow show supported and unsupported schemas and objects in Migration Portal.

Unsupported schemas

Exclude these Oracle systems schemas while generating the SQL dump file.

1	2	3	4	5
ANONYMOUS	APEX_PUBLIC_USER	APEX_030200	APEX_040000	APEX_040200
APPQOSSYS	AUDSYS	CTXSYS	DMSYS	DBSNMP
DBSFWUSER	DEMO	DIP	DMSYS	DVF
DVSYS	EXFSYS	FLOWS_FILES	FLOWS_020100	FRANCK
GGSYS	GSMADMIN_INTERNAL	GSMCATUSER	GSMROOTUSER	GSMUSER
LBACSYS	MDDATA	MDSYS	MGMT_VIEW	OJVMSYS
OLAPSYS	ORDPLUGINS	ORDSYS	ORDDATA	OUTLN
ORACLE_OCM	OWBSYS	OWBYSS_AUDIT	PDBADMIN	RMAN
REMOTE_SCHEDULER_AGENT	SI_INFORMTN_SCHEMA	SPATIAL_CSW_ADMIN_USR	SPATIAL_WFS_ADMIN_USR	SQLTXADMIN
SQLTXPLAIN	SYS$UMF	SYS	SYSBACKUP	SYSDG
SYSKM	SYSRAC	SYSTEM	SYSMAN	TSMSYS
WKPROXY	WKSYS	WK_TEST	WMSYS	XDB
XS$NULL				

Note: EDB Postgres Advanced Server doesn't support schema names starting with *pg_* .

Supported object types

- Tables
- Sequences
- Constraints
- Indexes (Except LOB indexes and indexes on materialized views)
- Synonyms
- Views
- Materialized views
- DB links
- Types and type body
- Triggers
- Functions
- Procedures
- Packages
- Users
- Roles
- Profiles
- Role and object grants

Note: COMMENTS on columns, tables, and materialized views are also supported.

Unsupported object types

- Editions
- Operators
- Schedulers
- LOB indexes and indexes on materialized views
- XML schemas
- Tablespaces
- Directories
- RLS policy
- Queues *
- Library

- Indextype

* Even though EDB Postgres Advanced Server provides support for Queue tables, Migration Portal does not currently support it. Queue tables in the source DDL are not uploaded as source and target DDL objects.

Extracting schemas using Oracle Data Pump utilities

Migration Portal requires only the metadata in the SQL dump file to assess the extracted schemas.

Prerequisites

- If you plan on exporting schemas that aren't your own, make sure that you're assigned the `DATAPUMP_IMP_FULL_DATABASE` role. Otherwise, you can export only your own schema.
- Make sure that you have enough tablespace quota to create objects in the tablespace.
- In SQL*Plus, create a directory object that points to a directory on your server file system. For example: **On-Linux** :

```
 create directory DMPDIR as /sample/file/path;
```   **OnWindows** :

```sql
 create directory DMPDIR as c:\sample\file\path\;
```

- In SQL*Plus, grant read-write permissions to the user running the Data Pump utility.

```
grant read,write on directory DMPDIR to <Username>;
```

Perform either of the following procedures:

- *Extract one or more schemas in a database*
- *Extract all schemas in a database*

Extract one or more schemas in a database

1. Before running the `expdp` command, create a file with a `.par` extension (for example, `export.par`) on your server. Add attributes and values to the file:

edb_notranlate_3 The attributes and values in this command specify the following options: - `CONTENT=metadata_only` extracts only the metadata of the schemas. - `INCLUDE=` specifies the database object types to include in the extraction. Extract only database object types that are supported by Migration Portal, see *Supported object types* .

2. From the command line, run the export command to generate a `schemas_metadata.dump` file. For example, to extract metadata information for `<Schema_1>` , `<Schema_2>` , `<Schema_3>` , and so on, run:

```
expdp <Username>@<ConnectIdentifier> DIRECTORY=DMPDIR SCHEMAS=<Schema_1>,<Schema_2>,
↪<Schema_3> DUMPFILE=schemas_metadata.dump parfile=export.par
```

3. To generate a SQL file from the dump file, run the import command.

For example, to generate a `YourSchemas.SQL` file from the `schemas_metadata.dump` file, enter:

```
impdp <Username>@<ConnectIdentifier> DIRECTORY=DMPDIR TRANSFORM=OID:n,SEGMENT_
↪ATTRIBUTES:n EXCLUDE=STATISTICS SQLFILE=YourSchemas.sql DUMPFILE=schemas_metadata.dump
```

See *File encoding* for information about the file encoding format expected by the Migration Portal.

Extract all schemas in a database

Note: Don't perform this procedure from a user account that belongs to the excluded schemas list (see *Unsupported schemas*). The *impdp* command fails if the user account running the command is in the excluded list of schemas.

1. Before running the `expdp` command, create a file with a `.par` extension (for example, `export.par`) on your server. Add attributes and values to the file:

edb_notranlate_6 The attributes and values in this command specify the following options: - `CONTENT=metadata_only` extracts only the metadata of the schemas. - `INCLUDE=` specifies the database object types to include in the extraction. Extract only database object types that are supported by Migration Portal, see *Supported object types* . - `FULL=Y` extracts all schemas in the database.

2. From the command line, run the export command to generate a `schemas_metadata.dump` file:

```
expdp <Username>@<ConnectIdentifier> DIRECTORY=DMPDIR DUMPFILE=schemas_metadata.dump␣
↪parfile=export.par
```

3. Before running the `impdp` command, create a parameter file with a `.par` extension (for example, `import.par`) on your server. Add attributes and values to the file:

```
EXCLUDE=SCHEMA:"IN(select username as schema_name from sys.dba_users where␣
↪LOWER(username) LIKE pg_% OR username IN
(ANONYMOUS,APEX_PUBLIC_USER,APEX_030200,APEX_040000,APEX_040200,APPQOSSYS,AUDSYS,CTXSYS,
↪DMSYS,DBSNMP,
DBSFWUSER,DEMO,DIP,DMSYS,DVF,DVSYS,EXFSYS,FLOWS_FILES,FLOWS_020100, FRANCK,GGSYS,
↪GSMADMIN_INTERNAL,
GSMCATUSER,GSMROOTUSER,GSMUSER,LBACSYS,MDDATA,MDSYS,MGMT_VIEW,OJVMSYS,OLAPSYS,ORDPLUGINS,
↪ORDSYS,
ORDDATA,OUTLN,ORACLE_OCM,OWBSYS,OWBYSS_AUDIT,PDBADMIN,RMAN,REMOTE_SCHEDULER_AGENT,SI_
↪INFORMTN_SCHEMA,
SPATIAL_CSW_ADMIN_USR,SPATIAL_WFS_ADMIN_USR,SQLTXADMIN,SQLTXPLAIN,SYS$UMF,SYS,SYSBACKUP,
↪SYSDG,SYSKM,
SYSRAC,SYSTEM,SYSMAN,TSMSYS,WKPROXY,WKSYS,WK_TEST,WMSYS,XDB,XS$NULL))"
EXCLUDE=STATISTICS
```

The attributes and values in this command specify the following options: - `EXCLUDE` specifies the schemas to exclude. Append a list of the

Unsupported schemas to ensure that the Data Pump utility doesn't extract any

system schemas (including schemas beginning with `pg_`). - `EXCLUDE=STATISTICS` specifies not to include statistics-related information in the SQL output file. By not including them in the SQL file, the size of the file is significantly reduced without affecting Migration Portal's ability to assess the DDL.

4. To generate a SQL file from the dump file, run the import command.

For example, to generate `YourSchemas.SQL` file from the `schemas_metadata.dump` file, enter:

edb_notranlate_9 See *File encoding* for information about the file encoding format expected by the Migration Portal.

4.2 Performing a schema assessment

You can assess an Oracle database schema for compatibility with EDB Postgres Advanced Server. You can also generate a report about the assessment.

4.2.1 Perform a schema assessment

1. Navigate to *Using Migration Portal* .
2. Enter your EDB credentials.
3. Select **New** to create a project.
4. In the dialog box, enter the project name in the **Projectname** field.
5. Specify project details:
   - Select the application interface. - Select a source DB and version for the source. - Select the target DB and Version for the target.
6. To upload the `.SQL` file generated with your schema metadata, select **Choosefile** .

Note: Don't modify the *.SQL* file.

7. To specify adding a suffix to indexes (_IDX) and constraints (_CON) when creating a project, select the **Addsuffixtomakeobjectnamesunique** check box. This selection ensures better assessment results, as EDB Postgres Advanced Server doesn't support the same name for tables and indexes whereas Oracle does.
8. To specify using Oracle default casing when creating a project, select the **UseOracle'sdefaultcaseforschemamigration** check box. Use the Oracle's default case feature to preserve the Oracle's default, uppercase naming convention for all database objects when migrating schema from an Oracle database to an EDB Postgres Advanced Server database. See *Known issues, limitations, and notes* for more information about this feature.
9. Select **Create&assess** to create a new project and to assess the schemas.

The analysis tool reviews every construct and executes repair actions to improve compatibility with EDB Postgres Advanced Server. It flags any remaining errors that require manual intervention.

10. Verify the DDL objects (e.g., TABLES) that don't show a 100% success ratio.
11. Select the objects that aren't compatible with EDB Postgres Advanced Server and view the details. At the bottom of the middle panel, you can view repair action details.
12. Refer to the Knowledge Base information in the right panel to locate the possible workarounds for the objects that aren't immediately compatible with EDB Postgres Advanced Server. You can also view the Knowledge Base information on the Portal wiki page.
13. On the **KnowledgeBase** tab, you can enter the error message for the incompatible objects with EDB Postgres Advanced Server and select **Search** .

The object detail panel displays the workaround or the resolution for the failed object. You can manually make the changes on the **Assessment** tab for that object and select **Reassess** .

Note: If any failed object passes while reassessing, the dependent objects for that object are also reassessed.

Similarly, you can make all incompatible objects compatible.

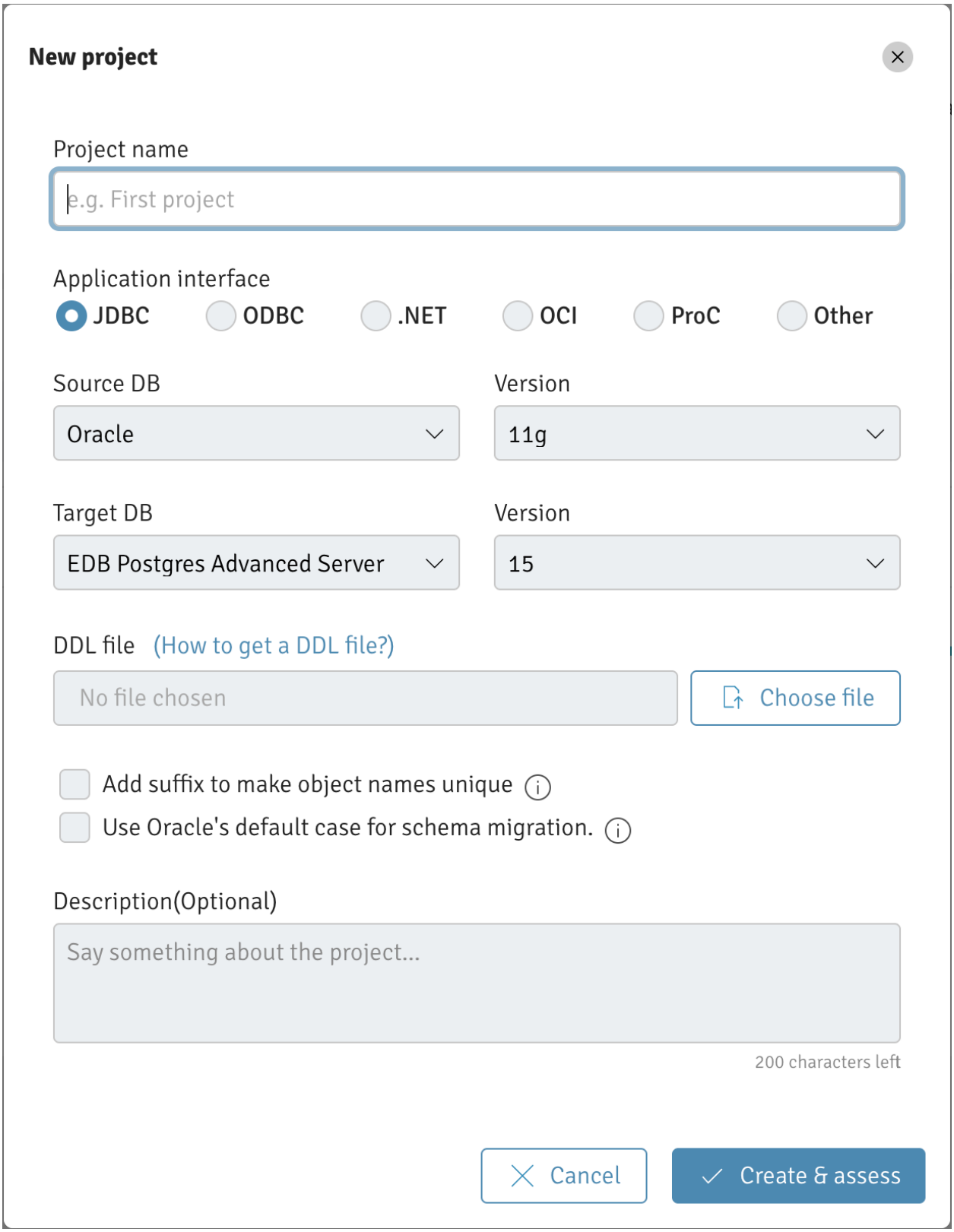


Fig. 1: The Migration Portal New project dialog

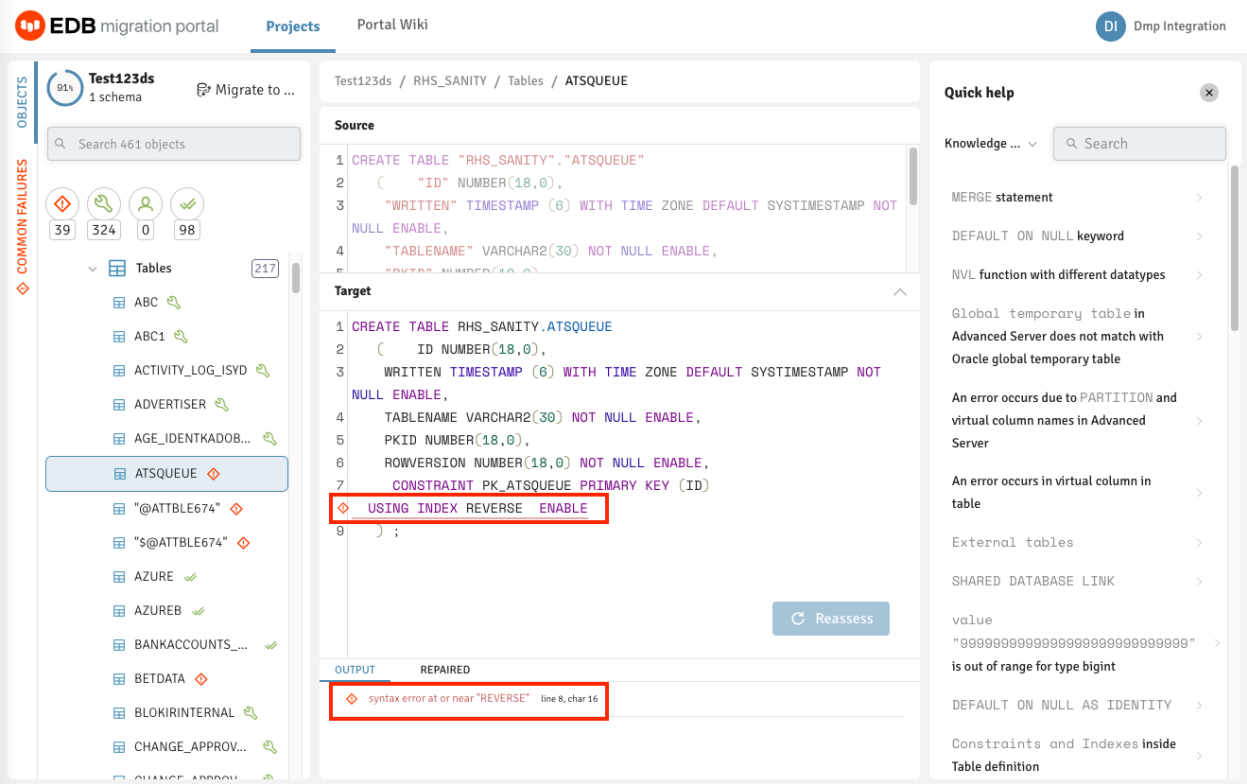


Fig. 2: Incompatible objects are identified

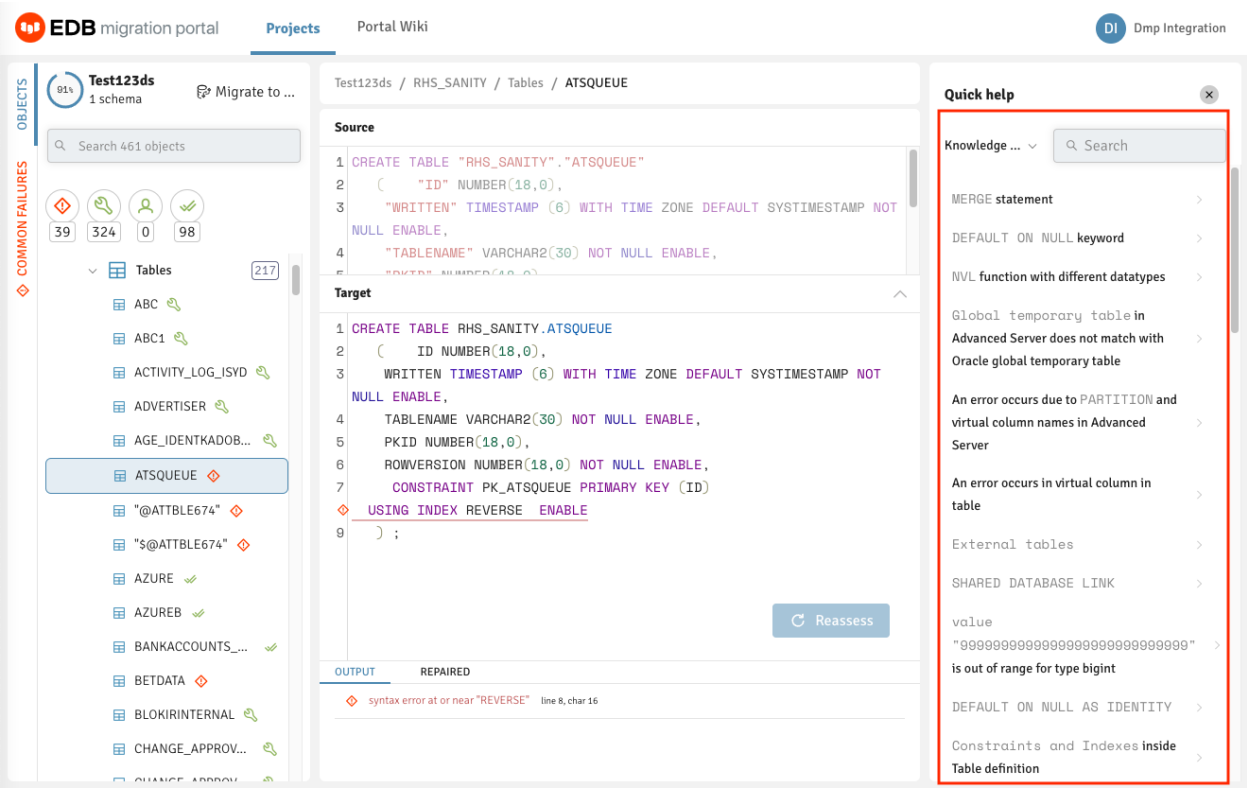


Fig. 3: Incompatible objects are identified

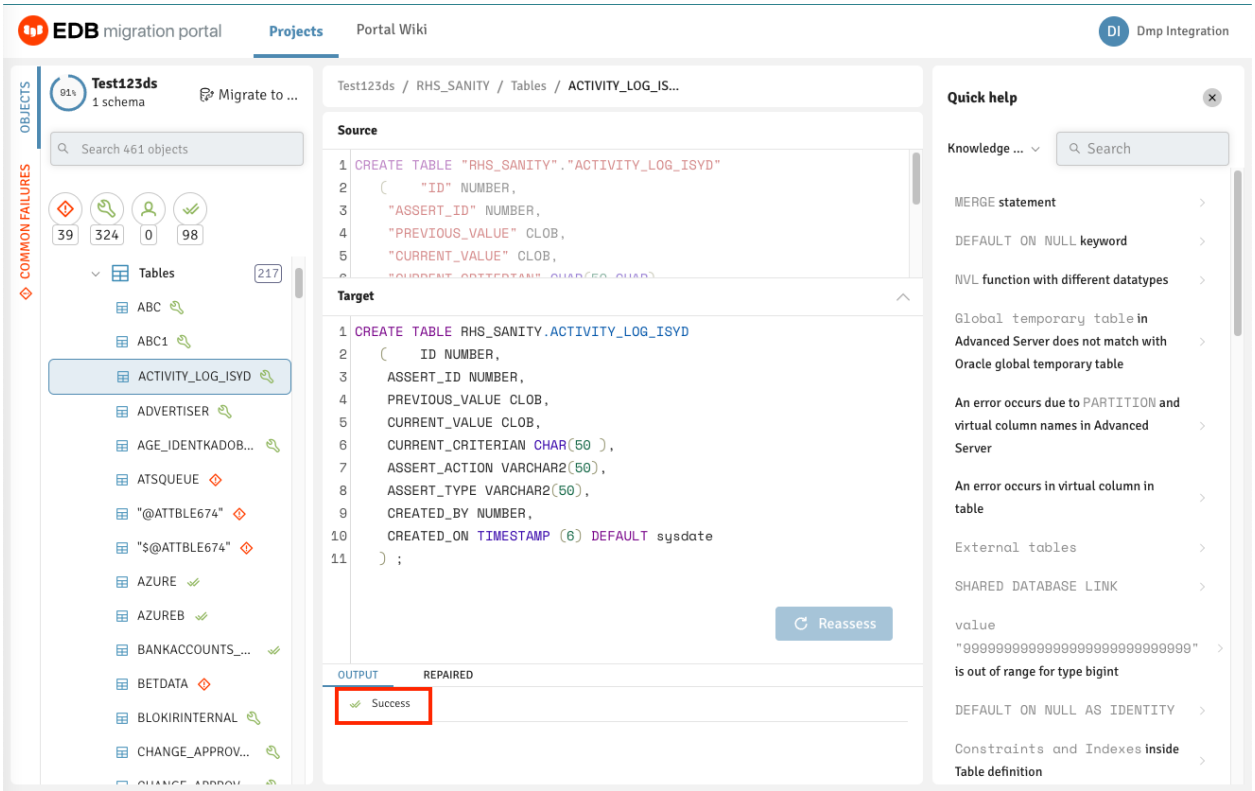


Fig. 4: Workaround or resolution for incompatible objects

Note: If the information or workaround for incompatible objects isn't available in the Knowledge Base, contact the Support team for help.

When you have finished working with the DDL, you can either download the modified EDB-compatible DDL as a `.sql` file or migrate the schemas. See *Performing a schema migration* .

4.2.2 Generate an assessment report

Migration Portal reports provide a high-level assessment summary of the schemas assessed for your project. In addition, the report provides details about the failed objects and the cause of failure.

1. Select **Report** .
2. Select the schemas to include in the report.
3. To generate the onscreen report, select **Generate** .

You can optionally select **GeneratePDF** to save the report in PDF format. You can also view the count of distinct repair handlers applied to the DDLs under the selected schemas.

Note: Migration Portal doesn't assess sensitive PL/SQL source code hidden in Oracle wrapped objects. These wrapped objects aren't included in the assessed objects count, and therefore the true value of compatibility percentage might be different from the value calculated in the assessment report. See the note in Known issues and notes for more information about wrapped objects.

4.2.3 Evaluate an assessment report

The assessment report presents data and information about the compatibility of objects and features in your schemas and the overall project.

- **Object** : Migration Portal generates an object for all CREATE statements it processes. In addition, it generates an object for each ALTER TABLE statement that adds a constraint in the uploaded DDL.
- **Feature** : An Oracle clause or a construct that is used in the creation of an object. A feature can be used multiple times in one or more objects.

The assessment report lists tables for *Incompatible Features* and *Unresolved Objects*.

Object compatibility

An object is compatible if it successfully executes on a target database. In the assessment report, you see a summary of object compatibility based on the following categories:

Object compatibility level	Description
Passed	Executes on a target database without applying any repair handlers.
System repaired	Executes successfully on a target database after applying one or more repair handlers. Repair handlers use transformation logic to automatically make DDLs compatible with EDB Postgres Advanced Server. Ensure that you carefully review the descriptions in the **Repair Handler applied** section of the report in case of an unexpected impact on the behavior of the object.
User repaired	Executes successfully on a target database after being manually repaired and re-assessed by the user.
Failed objects	Does not execute successfully on the target database.

Note: The object analysis presented is based on the existing state of the objects in the target DDL, which might change if the objects are updated and reassessed. For example, objects that are fixed are removed from the *Unresolved Objects* table after reassessment.

Feature compatibility

Some features might not be natively available in EDB Postgres Advanced Server. The assessment report presents a summary of feature occurrences and compatibility levels.

Feature compatibility level	Workaround status
Compatible features	•
Features with existing available workaround	Exists
Features with reasonable effort new workaround	Possible

continues on next page

Table 3 – continued from previous page

Feature compatibility level	Workaround status
Features with no workaround or requiring further investigation	None / Needs Investigation

Note: For features with no workaround, it might be possible to address the feature incompatibility through application-level changes. #### Limitations

The assessment report is updated after a user manually makes changes to the target DDL and selects **Reassess** . However, the *Incompatible Features* table in the report does not get updated and continues to list the incompatible features in the source DDL. The schema migration complexity assessment score is updated as incompatible objects with incompatible features are resolved. Also, after objects are fixed and reassessed they are removed from the *Unresolved Objects* table along with the listing of any incompatible features that were originally found in those objects.

4.3 Performing a schema migration

After resolving errors in your schemas, you can migrate the schemas to an EDB Postgres Advanced Server database.

The Migration Portal provides guided steps for migrating your schemas using one of the following target database options:

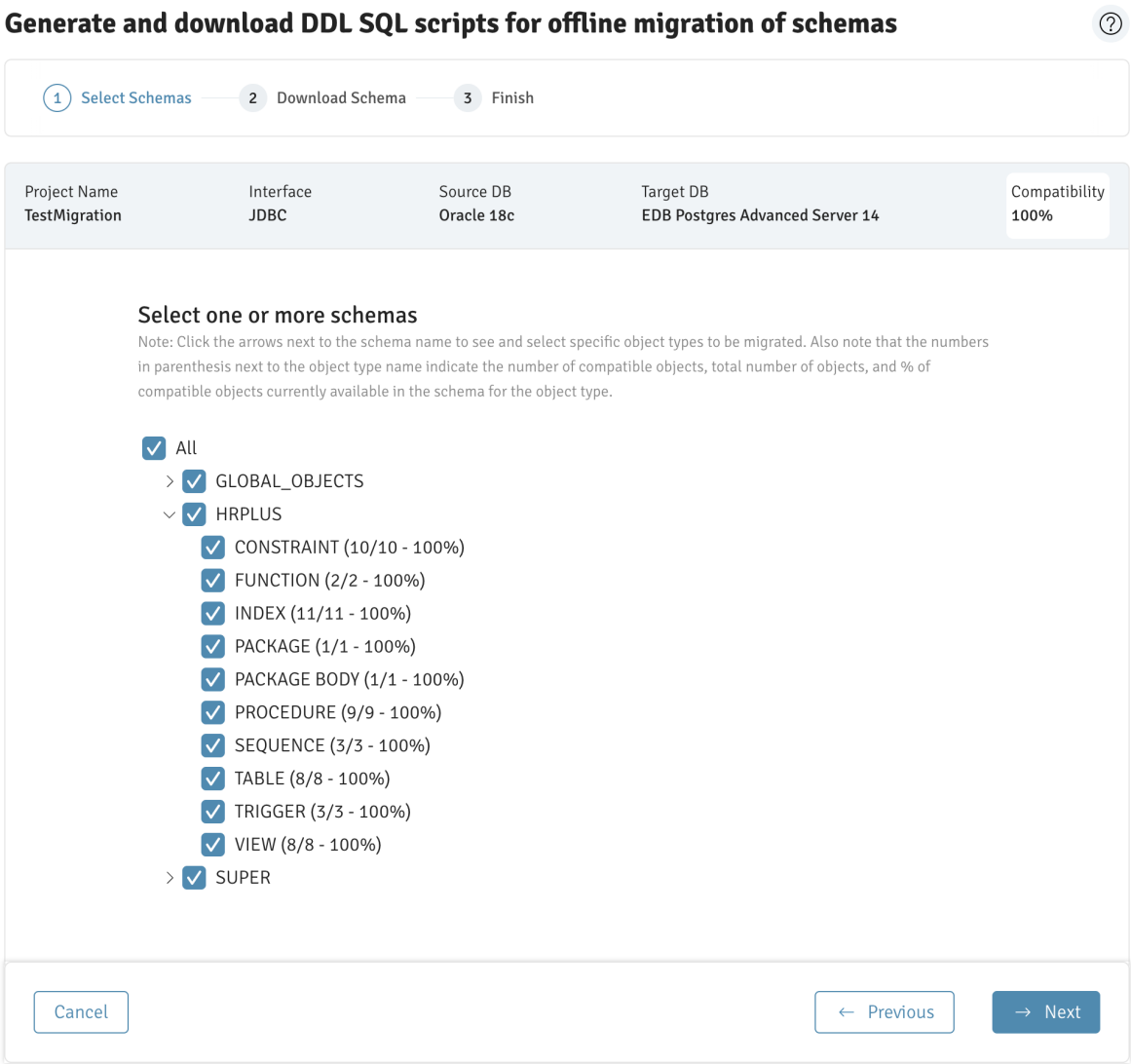
- Offline migration - download a data definition language (DDL) SQL file to create the schema manually in a target EDB Postgres Advanced Server database using a client application such as the psql client or pgAdmin. This option is suitable for both on-premises and cloud based migrations.
- Online migration - connect directly to a target EDB Postgres Advanced Server database and automatically create the schema in it. Generally, this option is most suitable for migrations to databases running in the cloud (either BigAnimal clusters or self-managed instances).

Before migrating a schema, EDB recommends that you identify any configuration requirements for the target database in *Known issues, limitations, and notes* .

4.3.1 Offline migration

Migrate schemas manually to an EDB Postgres Advanced Server on Windows or Linux platforms by running a DDL SQL file using a client.

1. Select **OfflineMigration** .
2. Select one or more schemas to migrate to EDB Postgres Advanced Server. You can also expand a schema and select specific object types under each schema.



Note: If your schemas aren't 100% compatible, a banner appears. Complete the Contact Us form as needed.

1. Select **DownloadSQLfile** , to download the assessed schema. The zip package consists of SQL files with DDLs for your selected schemas and objects. The DDLs are grouped in SQL files by project, schemas, or objects.

File name	Description	Example
<ProjectName> _ALL_OBJECTS.sql	A single consolidated file consisting of DDLs for all your selected schemas and objects.	**AllTestMigration_ALL_OBJECTS.sql** consists of DDLs for all objects of schemas OE, SH, IX, and PM, and only selected objects of schema HR.
<SchemaName> .sql	One or more files, each consisting of DDLs for selected objects in a particular schema.	**HR.sql** consists of DDLs for all the selected objects of the HR schema.

continues on next page

Table 4 – continued from previous page

File name	Description	Example
<SchemaName> _ *<ObjectType>* .sql	One or more files, each consisting of DDLs for a single selected object.	**HR_TABLE.sql** consists of DDLs only for TABLE object of the HR schema.

1. After the zip file finishes downloading, select **Done** to return to the main Migration Portal page.
2. Import the schema into your target database:
3. Use the edb psql command line utility to connect to the EDB Postgres Advanced Server database from where the converted schema is to be migrated.
4. Create a new database:

```
CREATE DATABASE <database_name>;
```

1. Connect to the new database using the following psql command:

```
\connect <database_name>
```

1. Import the required SQL files into the new database using the following psql command:

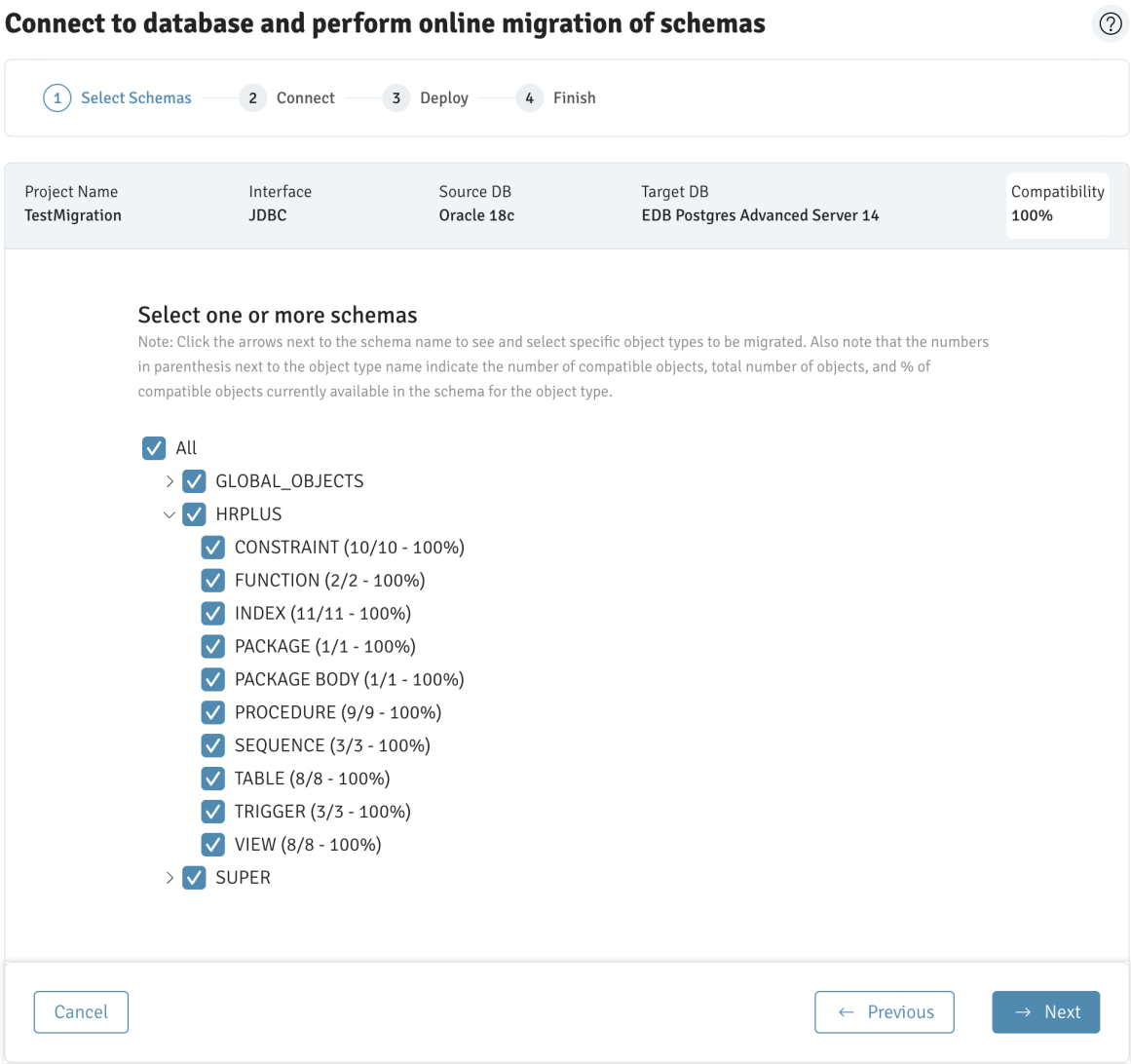
```
\i <path_to_exported_schema_file>.sql
```

Note: You can alternatively use the *pgAdmin* client for the import. Import the schema into your target database:

4.3.2 Online migration

Directly connect to and create schemas in the target EDB Postgres Advanced Server database.

1. Select **OnlineMigration** .
2. Select one or more schemas to migrate to EDB Postgres Advanced Server. You can also expand a schema and select specific object types under each schema.



3. Enter the required connection details on the `Connection` page.
1. To verify the connection details, select **TestConnection** .

Note: You can select **Edit Connection** to make changes to the connection details and retest the connection details.

1. If the test connection is successful, select **Next** to connect to the target database and create the selected schemas in the target server.
2. After the migration completes, a page displays with a message indicating whether the migration completed successfully or with errors. Select the **DownloadSummary** button to download and review a log generated for the migration attempt.
3. Select **Done** to return to the main Migration Portal page.

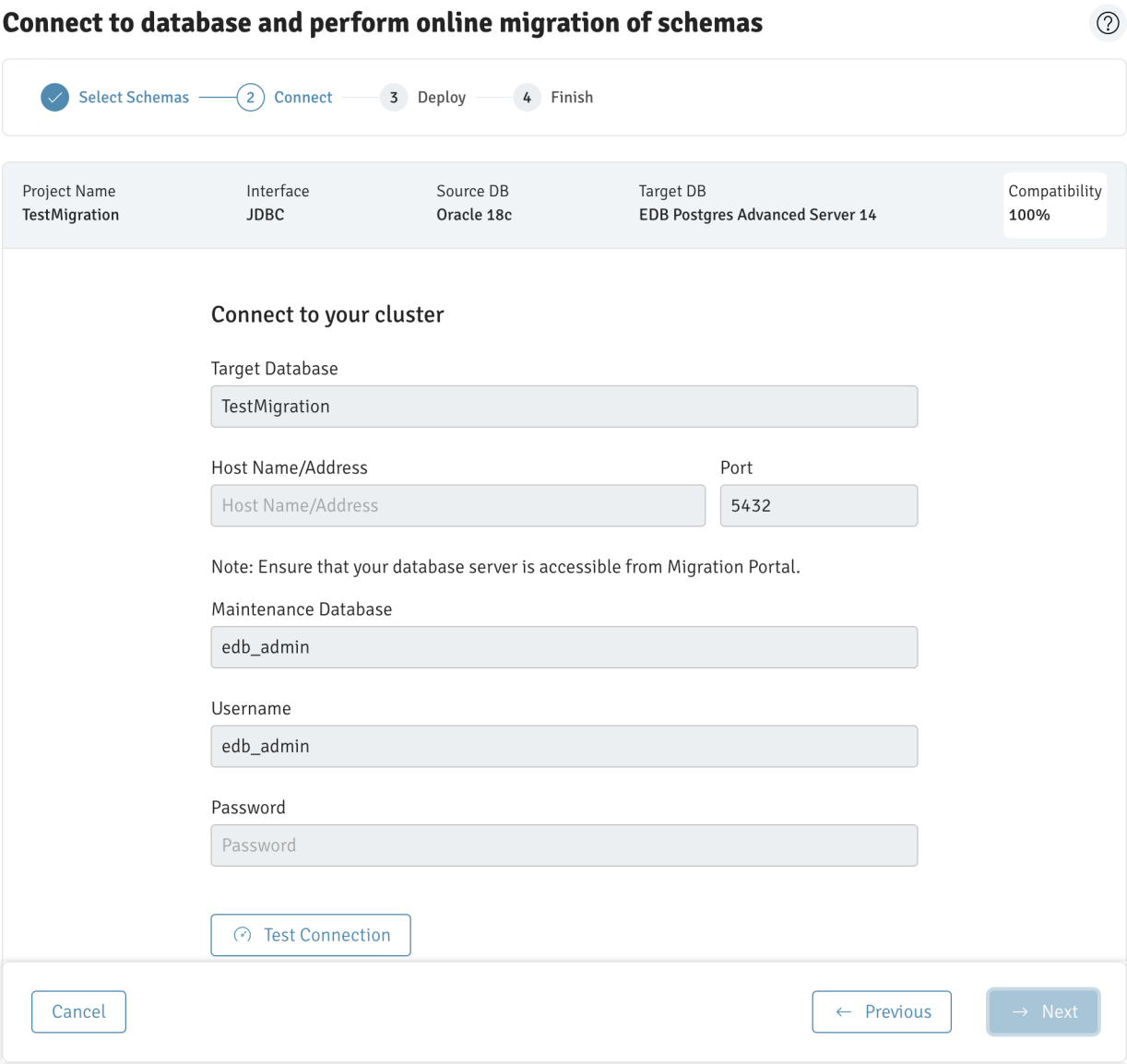


Fig. 5: Connecting to the cloud cluster

4.4 Performing a data migration

After performing the schema migration, migrate the data.

1. Use Migration Toolkit to migrate the data. For detailed information about using Migration Toolkit, see Migration Toolkit .
2. Configure the Migration Toolkit `toolkit.properties` file, ensuring that connection information for the source and target databases is available in the property file:

```
SRC_DB_URL = jdbc: oracle:thin:@localhost:1521:ORCL
SRC_DB_USER = user_name
SRC_DB_PASSWORD = password
TARGET_DB_URL= jdbc:edb://localhost:5444/migration
TARGET_DB_USER = enterprisedb
TARGET_DB-PASSWORD = password
```

For more information, see Building the toolkit.properties File .

3. Invoke Migration Toolkit in `-dataOnly` mode. Include the `-truncLoad` keyword to resolve foreign key dependencies across tables.

For example, the following command migrates data to a schema named `hr` :

```
runMTK.sh -dataOnly -targetSchema hr -truncLoad HR
```

The command migrates the specified `source_schema` to `target_schema` . The data is loaded into the locally installed EDB Postgres Advanced Server instance with a database superuser named `enterprisedb` and the password of `password` .

Note: The tables are truncated before attempting the data load.

4.4.1 Advanced data migration

For larger databases that require a parallel data load, you can use one of the following methods:

- The EDB Postgres Advanced Server database link feature (for compatibility with Oracle databases)
- A Dblink or a database link style migration (if your data contains `CLOB` data)

For more information, see the Migration Toolkit .

CHAPTER

FIVE

KNOWN ISSUES, LIMITATIONS, AND NOTES

This page lists known issues, limitations, and notes for:

- *Using Migration Portal* - *Extracting schemas using the EDB DDL Extractor* - *Extracting schemas using Oracle Data Pump utilities*

5.1 Migration Portal

5.1.1 Wrapped objects

Migration Portal can't assess wrapped objects. If you include them in the extracted DDL, they aren't loaded into Migration Portal and aren't included in the count of objects that are assessed. If you want to assess wrapped objects and migrate them to EDB Postgres Advanced Server, include unwrapped versions of the objects in the DDL file that you upload to Migration Portal. The recommended way of doing this is to replace the wrapped versions in the Oracle database with clear-text versions before performing the schema extraction. After performing the schema extraction, you can replace the objects with the wrapped versions.

5.1.2 File encoding

Migration Portal recommends the `.SQL` output file be in the UTF-8 encoding format. If you upload a `.SQL` file with non-UTF-8 encoding, all the characters that aren't compatible with UTF-8 are converted to the replacement character in the output DDL.

Tip: You can manually convert the extracted file to the UTF-8 format by using the iconv utility on Linux or the LibIconv utility on Windows. For example, if your database character set is in Latin-1 (ISO-8859-1), you can convert the extracted file to the UTF-8 format, as follows: *iconv -f iso-8859-1 -t UTF-8 sample.sql > sample_utf8.sql*

5.1.3 ALTER statements

Except for ALTER TABLE and ALTER TRIGGER, Migration Portal does not process any other ALTER statements in the DDL.

5.1.4 Users, roles, profiles, and grants

5.1.5 Grants

The Migration Portal now supports assessing and migrating the Oracle users for the schemas that have been extracted and uploaded. All the users, roles, and profiles for the schemas associated with a project are grouped together under a `GLOBAL_OBJECTS` pseudo schema in the Migration Portal views. Currently, grants of system privileges, object privileges, and roles are included with the DDL of the user object. Similarly, grants of system and object privileges to roles are included with the DDL of the role objects.

Since the grants are currently attached to the user and role objects, if one or more objects that the user is being granted a permission on fails, the user object itself fails assessment. To resolve this issue, any grant statements on failed objects can be commented out either using a SQL single line comment (`--`) for individual statements or a SQL multi-line comment (`/* */`) for consecutive grant statements and then a reassessment of the object can be performed. After the failed object corresponding to the failed grant statements have been repaired, the previously failing grant statement can be uncommented and the user object reassessed.

5.1.6 Privileges

To migrate a schema to a target EDB Postgres Advanced Server database, the target database user being used to perform the migration needs the following privileges:

- `SUPERUSER` to migrate profiles
- `SUPERUSER` or `CREATE ROLE` to migrate users and roles

5.1.7 Passwords

When migrating users, the Migration Portal does not migrate the users' passwords. The users are created in the target EDB Postgres Advanced Server database without a password. If it is desired or required that the users be configured with a password, this can be done manually in the target database after the users have been migrated.

5.1.8 Profiles

The Migration Portal migrates non-`DEFAULT` profiles assigned to the schema users. Oracle profiles include both password and resource limits. EDB Postgres Advanced Server only supports the Oracle password related limits. Resource limits are extracted with the profiles, but only the password limits are uploaded, assessed, and migrated by the Migration Portal. Also, in the Oracle database, the `DEFAULT` profile may have been overridden with new limit values. The `DEFAULT` profile is not migrated by the Migration Portal. If it is desired that the `DEFAULT` profile in EDB Postgres Advanced Server be configured with any limit overrides that have been applied to the Oracle `DEFAULT` profile, then they have to be applied manually in the target EDB Postgres Advanced Server database by using the `ALTER PROFILE` statement.

Note: Starting with version 4.3.0, the Migration Portal supports assessment of users, roles, profiles, and grants. However for the projects created using an earlier version (earlier than 4.3.0) of the Migration Portal, this capability is not supported. In order to assess the users, roles, profiles, and grants, use the latest version of the EDB DDL Extractor and create a new project.

5.1.9 Using Oracle default case

The option to use the Oracle default case feature preserves Oracle's default uppercase naming convention for all database objects when migrating a schema from an Oracle database to an EDB Postgres Advanced Server database.

The EDB DDL Extractor and Oracle Data Pump utilities extract all the DDLs in the default case of Oracle (that is, uppercase for object names or the respective case for case-sensitive object names) except for object names referenced inside the body of PL/SQL objects unless they are explicitly double-quoted by the user when creating the DDL.

So, the Migration Portal preserves all double quotes from the source schema file for this feature, and in addition, the Migration Portal applies double quotes to:

- The column names inside the `SELECT` statement of views and materialized views.
- The column names inside check constraints.
- The table name and column names inside the trigger header.

When the option to use Oracle casing is not selected, the default behavior of the Migration Portal is that all objects are created in lowercase (the default case of EDB Postgres Advanced Server) or mixed case if an object name is specified using a quoted mixed case in the source DDL.

Find the below source and target DDLs using the useOraCase feature:

```
- - Source DDL
CREATE TABLE "RHS_SANITY"."DEPT"
   (    "DEPTNO" NUMBER(2,0),
    "DNAME" VARCHAR2(14),
    "LOC" VARCHAR2(14),
     CONSTRAINT "DEPTDEPTNO_PK" PRIMARY KEY ("DEPTNO")
  USING INDEX  ENABLE,
     CONSTRAINT "DEPTDNAME_UK" UNIQUE ("DNAME")
  USING INDEX  ENABLE
   ) ;

- - Target DDL
CREATE TABLE "RHS_SANITY"."DEPT"
   (    "DEPTNO" NUMBER(2,0),
    "DNAME" VARCHAR2(14),
    "LOC" VARCHAR2(14),
     CONSTRAINT "DEPTDEPTNO_PK" PRIMARY KEY ("DEPTNO")
  USING INDEX  ENABLE,
     CONSTRAINT "DEPTDNAME_UK" UNIQUE ("DNAME")
  USING INDEX  ENABLE
   ) ;
```

While using the Oracle default case, you may experience a lower compatibility ratio compared to other projects. The following are reasons that might contribute to this lower compatibility ratio:

- The most common reason is that the object/relation doesn't exist. This scenario is caused by the fact that the table/object is in double-quoted upper case to preserve the Oracle case, whereas in the failing object it is referenced without the double-quotes. For example:

```
CREATE TABLE "HR"."EMP"
   (   "EMPNO" NUMBER(4,0),
         "ENAME" VARCHAR2(10),
         "JOB" CLOB,
```

(continues on next page)

(continued from previous page)

```
         "MGR" NUMBER(4,0),
         "HIREDATE" DATE,
         "SAL" NUMBER(7,2),
         "COMM" NUMBER(7,2),
         "DEPTNO" NUMBER(2,0),
       CONSTRAINT "PK_EMP" PRIMARY KEY ("EMPNO")
  USING INDEX  ENABLE
   ) ;

CREATE OR REPLACE  PROCEDURE "HR"."PROC"
IS
 CURSOR emp_cur IS SELECT hiredate, CAST(hiredate AS timestamp(6)) AS timestamped FROM␣
↪emp;
 REC emp_cur%rowtype;
BEGIN
   OPEN emp_cur;
   LOOP
   FETCH emp_cur INTO rec;
       EXIT WHEN emp_cur%NOTFOUND ;
               DBMS_OUTPUT.PUT_LINE(rec.hiredate ||  ||rec.timestamped);
         END LOOP;
    CLOSE EMP_CUR;
   END ;
```

The error displays as follows:

```
   relation "emp" does not exist
```  In the example above, the table name and its column names are specified with quoted␣
↪uppercase names in the `CREATE TABLE`  statement; however, in the `CREATE OR REPLACE␣
↪PROCEDURE`  statement, the table and its columns are referenced using using unquoted␣
↪lowercase identifiers.

-  The variables defined using `%type`  are failing in code objects as preceding `table.
↪column-name`  are in lower case, like `employees.employee_id%type`  whereas the table␣
↪name and the column names for the actual table are in double-quoted uppercase. For␣
↪example:

```sql
   CREATE TABLE "HR"."EMPLOYEES"
      (    "EMPLOYEE_ID" NUMBER(6,0),
             "FIRST_NAME" VARCHAR2(20),
             "LAST_NAME" VARCHAR2(25) CONSTRAINT "EMP_LAST_NAME_NN" NOT NULL ENABLE,
             "EMAIL" VARCHAR2(25) CONSTRAINT "EMP_EMAIL_NN" NOT NULL ENABLE,
             "PHONE_NUMBER" VARCHAR2(20),
             "HIRE_DATE" DATE CONSTRAINT "EMP_HIRE_DATE_NN" NOT NULL ENABLE,
             "JOB_ID" VARCHAR2(10) CONSTRAINT "EMP_JOB_NN" NOT NULL ENABLE,
             "SALARY" NUMBER(8,2),
             "COMMISSION_PCT" NUMBER(2,2),
             "MANAGER_ID" NUMBER(6,0),
             "DEPARTMENT_ID" NUMBER(4,0),
              CONSTRAINT "EMP_SALARY_MIN" CHECK ("SALARY" > 0) ENABLE,
              CONSTRAINT "EMP_EMP_ID_PK" PRIMARY KEY ("EMPLOYEE_ID")
```

(continues on next page)

(continued from previous page)

```
    USING INDEX  ENABLE,
             CONSTRAINT "EMP_EMAIL_UK" UNIQUE ("EMAIL")
    USING INDEX  ENABLE
     ) ;
  CREATE OR REPLACE  PROCEDURE "HR"."PKG" AS
  -- declare variables for data fetched from cursor
    empid       employees.employee_id%TYPE; -- variable for employee_id
    hiredate    employees.hire_date%TYPE;   -- variable for hire_date
    firstname   employees.first_name%TYPE;  -- variable for first_name
    lastname    employees.last_name%TYPE;   -- variable for last_name
    rowcount    NUMBER;
    bonusamount NUMBER;
    yearsworked NUMBER;
  -- declare the cursor with a parameter
    CURSOR cursor1 (thismonth NUMBER)IS
      SELECT employee_id, first_name, last_name, hire_date FROM employees
         WHERE EXTRACT(MONTH FROM hire_date) = thismonth;
  BEGIN
    null;
  END;
```

The error displays as follows:

```
syntax error at or near "%" (line 3, char 37)
```

- When tables have TYPE reference in column data type they fail due to the difference in case. For example:

```
CREATE OR REPLACE TYPE "HR"."USER_TYPE" IS OBJECT(
    "ID" NUMBER,
    "NAME" VARCHAR2(256)
);

CREATE TABLE "HR"."MY_TABLE"
(
    "ID" NUMBER,
    "MY_NAME" user_type
);
```

The error displays as follows:

```
type "user_type" does not exist
LINE 4:     "MY_NAME" user_type
              ^
SQL state: 42704
Character: 58
```

5.2 EDB DDL Extractor

- The EDB DDL Extractor script doesn't extract objects restored using `Flashback` that still have names like `BIN$b54+4XlEYwPgUAB/AQBWwA==$0` . If you want to extract these objects, you must change the names of the objects and rerun the extraction process.
- The EDB DDL Extractor extracts `nologging` tables as normal tables. Once these tables are migrated to EDB Postgres Advanced Server, WAL log files are created.
- The EDB DDL Extractor extracts objects only with `VALID` status. For any objects that have `INVALID` status that you want Migration Portal to assess, first update them to `VALID` .
- The EDB DDL Extractor doesn't extract objects that were obfuscated using the Oracle wrap feature. As such, these objects aren't included in the set of DDL assessed by the Migration Portal. If you want to assess these objects and migrate them to EDB Postgres Advanced Server, replace the wrapped versions of the objects with nonwrapped versions. See *Wrapped objects* for more information.
- The EDB DDL Extractor creates Global Temporary tables to store the schema names and their dependency information. These tables are dropped at the end of successful extraction.
- The EDB DDL Extractor script doesn't extract schemas whose name starts with `PG_` because PostgreSQL doesn't support it. If you want to extract these schemas, you must change the name of the schema before extraction.
- The EDB DDL Extractor automatically extracts the information for the profiles, roles, and grants.

5.3 Oracle Data Pump utilities

- Migration Portal might fail to parse your SQL file if you create a database link using the `IDENTIFIED BY` clause with Oracle's quote operator; for example, `IDENTIFIED BY VALUES q'[:1]'` . To parse your file successfully, try using an actual password; for example, `IDENTIFIED BY my_password` .
- The DDL generated by Oracle Data Pump utilities might contain `ALTER STATEMENTS` such as `ALTER FUNCTION` , `ALTER PACKAGE` , and `ALTER TYPE` , which are not processed by Migration Portal.
- Profiles and roles are not extracted when the Oracle Data Pump is executed in schema mode - that is, when the `SCHEMAS` parameter is used when executing the `expdp` command. If the schema user being exported has a profile assigned to it, the assessment of the schema user object fails in Migration Portal since the profile is not part of the exported `impdp` file. To correct the issue, remove the profile assignment in the user object target DDL and reassess the user creation object.