

## Migrating from OWB 11gR2 (11.2.0.4) to ODI 12c (12.1.3)

---

We all know that OWB (Oracle Warehouse Builder), although still supported by Oracle, is being replaced by ODI (Oracle Data Integrator). The first question that Oracle customers ask is if they will need to recode their ETLs?

Thankfully Oracle provides the OWB to ODI Migration Utility which promises to make the transition from OWB to ODI easy. Full documentation on the utility is available at:

<https://docs.oracle.com/middleware/1213/odi/install-migrate-owb-odi/understanding.htm>

<http://www.oracle.com/technetwork/articles/datawarehouse/bryson-owb-to-odi-2130001.html>

The current version of ODI (currently 12.1.3.0) can be downloaded from:

<http://www.oracle.com/technetwork/middleware/data-integrator/downloads/index.html>

I already had the latest version of OWB 11g 11.2.0.4 Standalone Client for Windows 64bit (p17743124\_112040\_MSWIN-x86-64.zip) and an export of an OWB ETL scenario from a previous project.

Note that you can migrate only from OWB version 11.2.0.3 or 11.2.0.4. If you have a previous version you have first to upgrade to the latest one, before migrate to ODI.

My test environment is a virtual machine with Windows 2012 R2 Standard Edition. First I installed the Oracle Database 12c Enterprise Edition Release 12.1.0.2.0 as per

<http://docs.oracle.com/database/121/NXDQI/toc.htm>

Next step is to installed the OWB 11.2.0.4 and create the workspace.

Note that when running the OWB design centre for the first time you will be asked to create a new workspace. You may receive a java error on this page, if so, you can resolve the issue as follows:

1. Before you run the **<OWB\_HOME>\owb\UnifiedRepos\cat\_owb.sql** script to create the OWBSYS and OWSYS\_AUDIT users be sure that the oracle windows user (if you have create one while installing the database) has full access to the following paths:
  - a. OWB\_HOME/owb/bin/admin
  - b. OWB\_HOME/owb/logNow you can run the **cat\_owb.slq**. (if you already have run it, then first run **<OWB\_HOME>\owb\UnifiedRepos\clean\_owbsys.sql**)
2. Create the new workspace "manually" by calling the Create Repository Assistant **<OWB\_HOME>\owb\bin\win\reposinst.bat** before you run the "Design Center" for first time.

Finally I imported the ETL scenario I had and run it to verify that is working.

So far, so good, but still I only prepared my environment. I haven't touch the Migration tool yet.

Next step is to install the ODI 12c.

It is not necessary to install it in the same machine, but as my purpose is to test, I decided to have everything in one place.

The installation was straight forward without issues (See <http://docs.oracle.com/middleware/1213/core/ODING/install.htm> ).

So now I have the database, the OWB and the ODI installed and I am ready to start the migration.

According to the Oracle Migration Utility document we need to patch OWB with the latest Migration Patch, 18537208 OWB-TO-ODI MIGRATION PATCH (MP2) FOR OWB 11.2.0.4 from Oracle support (p18537208\_112040\_MSWIN-x86-64.zip)

Now I apply the patch.

In the zip file there are instructions how to do it. I extract the zip file usually in the <OWB HOME>\OPatch and the folder 18537208 is created. I navigate into that folder from MS-DOS as administrator and run the following:

```
set ORACLE_HOME=E:\OWB11g
set PATH=%ORACLE_HOME%\bin;%PATH%
set JAVA_HOME=C:\Java\jdk1.7.0_79\jre
set PATH=%JAVA_HOME%\bin;%PATH%
set PATH=%ORACLE_HOME%\OPatch;%PATH%

opatch apply
```

*where E:\OWB11g is the path I have installed the OWB and C:\Java\jdk1.7.0\_71\jre the Java path*

I check that the installation was ok by running in the same DOS session : opatch lsinventory

```
Administrator: Command Prompt

E:\OWB11g\OPatch\18537208>opatch lsinventory
Oracle Interim Patch Installer version 11.2.0.3.4
Copyright (c) 2012, Oracle Corporation. All rights reserved.

Oracle Home      : E:\OWB11g
Central Inventory : C:\Program Files\Oracle\Inventory
   from          : n/a
OPatch version    : 11.2.0.3.4
OUI version       : 11.2.0.4.0
Log file location : E:\OWB11g\cfgtoollogs\opatch\opatch2015-06-30_08-54-59AM_1.log
Lsinventory Output file location : E:\OWB11g\cfgtoollogs\opatch\lsinv\lsinventory2015-06-30_08-54-59AM.txt

-----

Installed Top-level Products (1):

OWB 11.2 Complete Installation                11.2.0.4.0
There are 1 products installed in this Oracle Home.

Interim patches (1) :

Patch 18537208      : applied on Mon Jun 29 11:41:32 BST 2015
Unique Patch ID:    17844448
   Created on 9 Jun 2014, 23:47:10 hrs PST8PDT
   Bugs fixed:
     18537208

-----

OPatch succeeded.

E:\OWB11g\OPatch\18537208>_
```

The Oracle documentation also mentions that we need “ODI 12.1.2 with ODI 12.1.2.0.1 Bundle Patch (patch number 17836908) or ODI 12.1.3”.

Now it's time to start the migration.

1. In the path **E:\OWB11g\owb\bin\admin** I copy the file **migration.config**, paste it in **E:\OWB\_migration** and modify it according to my configuration (We will see later we can have that configuration file at any place in our disk, as we give the full path while running the migration script).

```
ODI_MASTER_USER=ODIREP
ODI_MASTER_URL=jdbc:oracle:thin:@localhost:1521:ora112
ODI_MASTER_DRIVER=oracle.jdbc.OracleDriver
ODI_USERNAME=SUPERVISOR
ODI_WORK_REPOSITORY_NAME=WORK0
OWB_WORKSPACE_OWNER=rep_0
OWB_URL=localhost:1521:ora112.us.oracle.com
OWB_WORKSPACE_NAME=REP_0_WS_0
MIGRATION_LOG_FILE=/tmp/migration.log
MIGRATION_REPORT_INCLUDE=ALL
```

```
MIGRATION_MODE=RUN
MIGRATE_DEPENDENCIES=false
STOP_ON_ERROR=false
SPLIT_JOIN_FOR_ANSI_SYNTAX=true
MIGRATE_UNBOUND_OPERATOR=false
MIGRATION_OBJECTS=PROJECT.MY_RPROJECT; PROJECT.PROJECT_1.MODULE.MODULE_1;
FLUSH_BATCH_SIZE=50
MIGRATION_STRATEGY=CREATE
```

After the changes the new migration file is:

```
ODI_MASTER_USER=DEV_ODI_REPO
ODI_MASTER_URL=jdbc:oracle:thin:@localhost:1521:orcl
ODI_MASTER_DRIVER=oracle.jdbc.OracleDriver
ODI_USERNAME=SUPERVISOR
ODI_WORK_REPOSITORY_NAME=WORKREP
OWB_WORKSPACE_OWNER=OWB_DBA
OWB_URL=localhost:1521:orcl.metalogic.local
OWB_WORKSPACE_NAME=OWB_WORKSPACE
MIGRATION_LOG_FILE=E:\OWB_migration\migration.log
MIGRATION_REPORT_INCLUDE=ALL
#MIGRATION_MODE=RUN
MIGRATION_MODE= FAST_CHECK
MIGRATE_DEPENDENCIES=false
STOP_ON_ERROR=false
SPLIT_JOIN_FOR_ANSI_SYNTAX=true
MIGRATE_UNBOUND_OPERATOR= false
MIGRATION_OBJECTS=PROJECT.DM_PROJECT;
FLUSH_BATCH_SIZE=50
MIGRATION_STRATEGY=CREATE
DEBUG=OFF
```

As you can see in the file there are 3 modes to run the migration script

**a. FAST\_CHECK:**

Under this mode, the migration application performs a check for selected candidates and reports which objects can or cannot be migrated to the target ODI repository.

**b. DRY\_RUN:**

Under this mode, the migration will be performed without committing the migrated objects to the ODI repository to check which objects can or cannot be migrated with more detailed/accurate reasons than the FAST\_CHECK mode.

**c. RUN:**

Under this mode, the real migration will be performed and migrated objects will be written to ODI repository.

First I will run it in FAST\_CHECK mode to check the log file.

2. In the path **E:\OWB11g\owb\bin\win** there is the file **migration.bat** which is the one we have to run. If we open it we can see that it runs as follows:

```
migration.bat <ODI_MASTER_REPOSITORY_PASSWORD> <ODI_SUPERVISOR_PASSWORD>
<OWB_WORKSPACE_PASSWORD> <FULLPATH_OF_CONFIGURATION_FILE>
```

where

**<ODI\_MASTER\_REPOSITORY\_PASSWORD>**: The password of the database use **DEV\_ODI\_REPO** (in my installation)

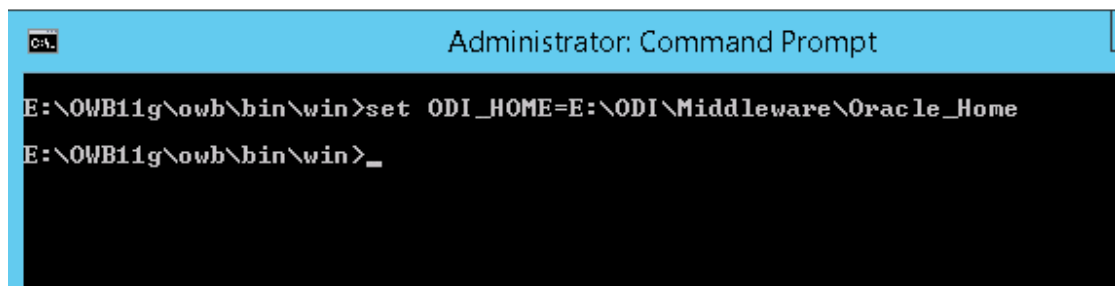
**<ODI\_SUPERVISOR\_PASSWORD>** : The password of the **SUPERVISOR** user at **OWB**

**<OWB\_WORKSPACE\_PASSWORD>** : The **OWB** workspace user (**owb\_dba** in my case)

**<FULLPATH\_OF\_CONFIGURATION\_FILE>** : The full path of the **migration.config** file  
(**E:\OWB\_migration\migration.config** in my case)

I create the file **run\_migration.bat** with the above command and save it in the same path where the **migration.bat** file is.

We open an MS-DOS window as administrator, navigate in **E:\OWB11g\owb\bin\win** .  
Set the **ODI\_HOME** and run the script:



```
Administrator: Command Prompt
E:\OWB11g\owb\bin\win>set ODI_HOME=E:\ODI\Middleware\Oracle_Home
E:\OWB11g\owb\bin\win>_
```

Run the file **run\_migration.bat**

When is done we will see the message "Migration finished".

3. Now we check the 2 log files that have been created at **E:\OWB\_migration** folder
  - a. **migration.log**
  - b. **migration.report**

In the second report we see:

```
...
Statistics
-----

Total Projects Checked: 1

*****
***
PROJECT: DM_PROJECT

Object Types      Can be migrated      Cannot be migrated      Skipped
-----
SEQUENCE:                5                0                0
```

TABLE:	51	0	0
MAPPING_MODULE:	2	0	0
MODULE:	2	0	0
MAPPING:	6	0	0
Details			
-----			
...			

which means that we will have no issues on migration.

4. Modify mode the E:\OWB\_migration\migration.config file and run it again in mode=DRY-MODE.
5. Check the log files again. Now at **migration.report** I see the following:

```

...
Total Projects Checked: 1

*****
***
PROJECT: DM_PROJECT

Object Types          Can be migrated      Cannot be migrated
Skipped
-----
---
                SEQUENCE:                5                0
0
                TABLE:                51                0
0
        MAPPING_MODULE:                2                0
0
                MODULE:                2                0
0
                MAPPING:                2                4
0
MODULE_FOR_LOGICALSCHEMA:                2                0
0

Details
...
MAPPING
  LOAD_DM_COURSES_DIM          SUCCESS
  LOAD_DM_ENROL_FACT          SUCCESS
  LOAD_DM_PROGRAMMES_DIM      [INFO][Migration][MU-5023]Mapping is not
migrated because the mapping operator TABLE:DM_PROGRAMMES_DIM is unbound. Use
the configuration option of migration utility "MIGRATE_UNBOUND_OPERATOR" or
fix the mapping with unbound operators.
  LOAD_DM_STUDENTS_DIM        [INFO][Migration][MU-5023]Mapping is not
migrated because the mapping operator TABLE:DM_STUDENTS_DIM is unbound. Use
the configuration option of migration utility "MIGRATE_UNBOUND_OPERATOR" or
fix the mapping with unbound operators.
  LOAD_DM_STUDENT_PROGRAMME_DIM [INFO][Migration][MU-5023]Mapping is not
migrated because the mapping operator TABLE:DM_STUDENT_PROGRAMME_DIM is
unbound. Use the configuration option of migration utility
"MIGRATE_UNBOUND_OPERATOR" or fix the mapping with unbound operators.
  LOAD_DM_TIME_DIM            [INFO][Migration][MU-5023]Mapping is not
migrated because the mapping operator TABLE:DM_TIME_DIM is unbound. Use the
configuration option of migration utility "MIGRATE_UNBOUND_OPERATOR" or fix
the mapping with unbound operators.
MODULE_FOR_LOGICALSCHEMA

```

SRC_ODS_DBA	SUCCESS
TRG_DM_DBA	SUCCESS
...	

And at **migration.log**:

```

MAPPING[TOTAL:6 CAN_BE_MIGRATED:2 CAN_NOT_BE_MIGRATED:4 SKIPPED:0] .
----PASSED:
PROJECT[DM_PROJECT].MODULE[TRG_DM_DBA].MAPPING[LOAD_DM_COURSES_DIM] .
----PASSED:
PROJECT[DM_PROJECT].MODULE[TRG_DM_DBA].MAPPING[LOAD_DM_ENROL_FACT] .
----FAILED:
PROJECT[DM_PROJECT].MODULE[TRG_DM_DBA].MAPPING[LOAD_DM_PROGRAMMES_DIM] .
----FAILED:
PROJECT[DM_PROJECT].MODULE[TRG_DM_DBA].MAPPING[LOAD_DM_STUDENTS_DIM] .
----FAILED:
PROJECT[DM_PROJECT].MODULE[TRG_DM_DBA].MAPPING[LOAD_DM_STUDENT_PROGRAMME_DIM]
.
----FAILED: PROJECT[DM_PROJECT].MODULE[TRG_DM_DBA].MAPPING[LOAD_DM_TIME_DIM] .
        MODULE_FOR_LOGICALSCHEMA[TOTAL:2 CAN_BE_MIGRATED:2
        CAN_NOT_BE_MIGRATED:0 SKIPPED:0] .
----PASSED: PROJECT[DM_PROJECT].MODULE[SRC_ODS_DBA] .
----PASSED: PROJECT[DM_PROJECT].MODULE[TRG_DM_DBA] .

```

Now we see that 4 out of 6 Modules failed with the error: **Mapping is not migrated because the mapping operator TABLE:<Table\_Name> is unbound**

To fix that issue let's go to the **migration.config** file and change the value of **MIGRATE\_UNBOUND\_OPERATOR** from **false** to **true** and run the script one more in DRY\_RUN mode. Now we see:

```

...
Statistics
-----

Total Projects Checked: 1

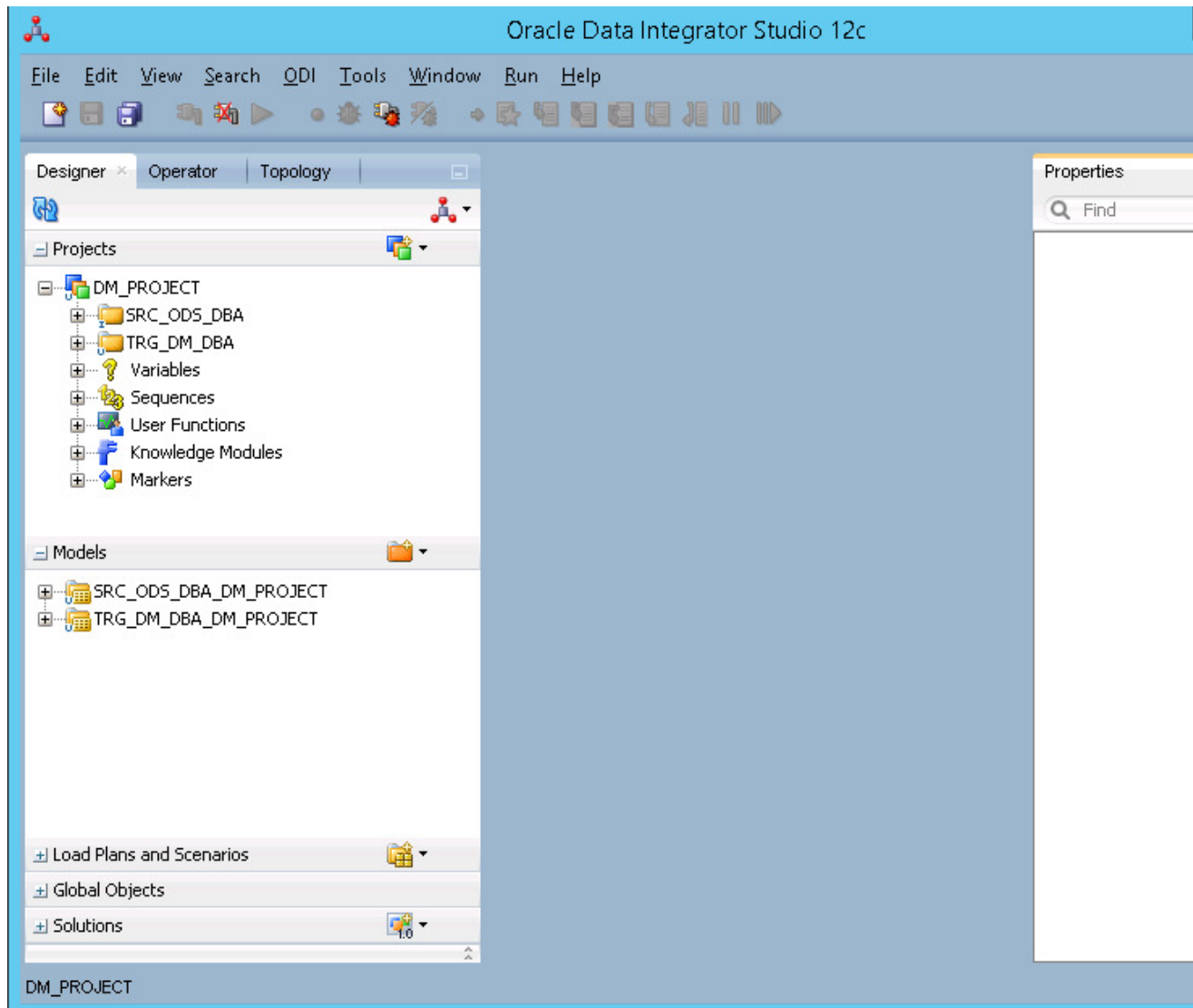
*****
***
PROJECT: DM_PROJECT

Object Types      Can be migrated      Cannot be migrated      Skipped
-----
      SEQUENCE:           5              0              0
      TABLE:            51              0              0
MAPPING_MODULE:      2              0              0
      MODULE:            2              0              0
      MAPPING:           6              0              0

Details
-----
...

```

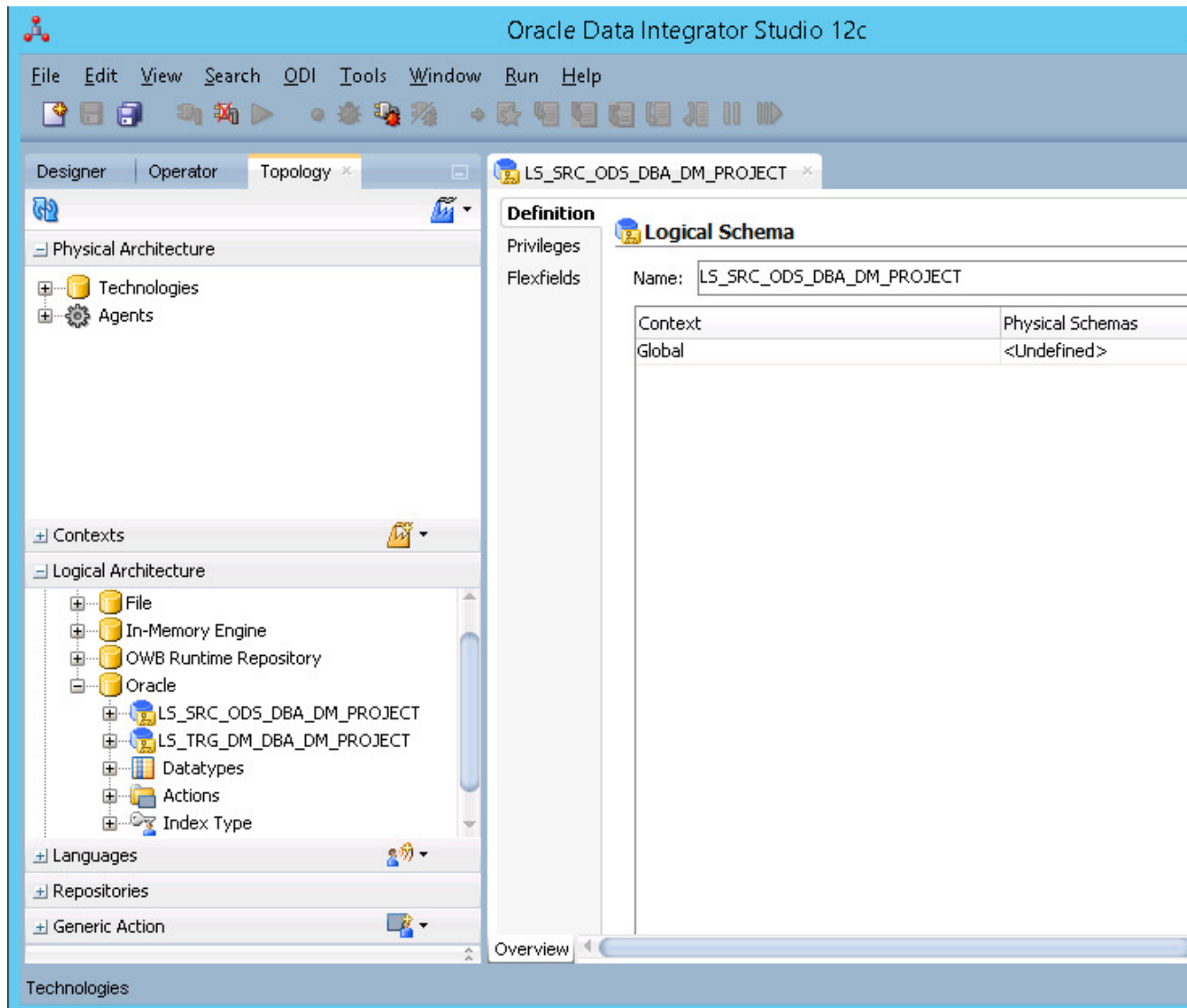
6. So now we can run the script in RUN mode to perform the migration.
7. Open the ODI Studio and connect to the repository.



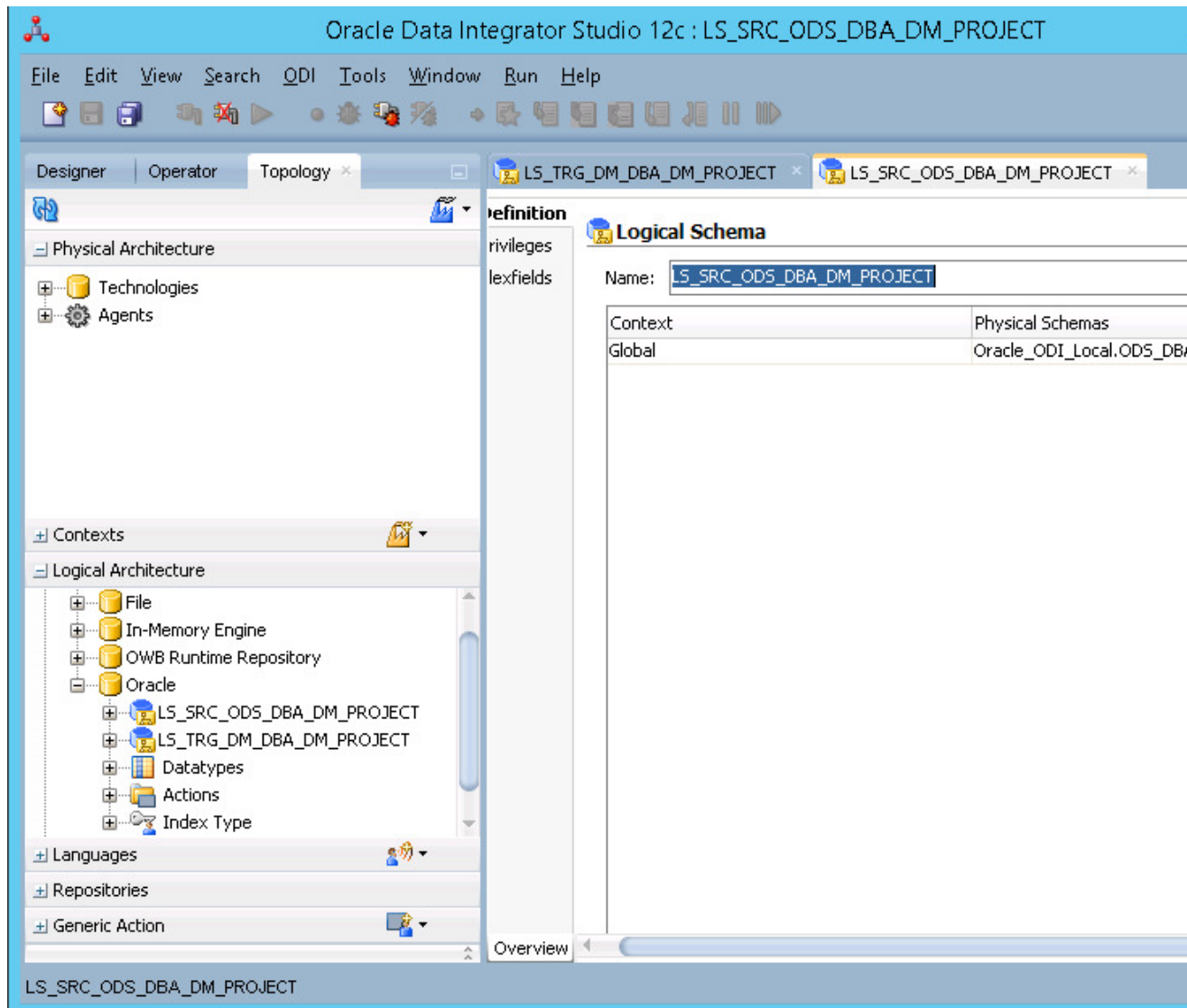
And we see the migrated project.

8. Now check the Topology.

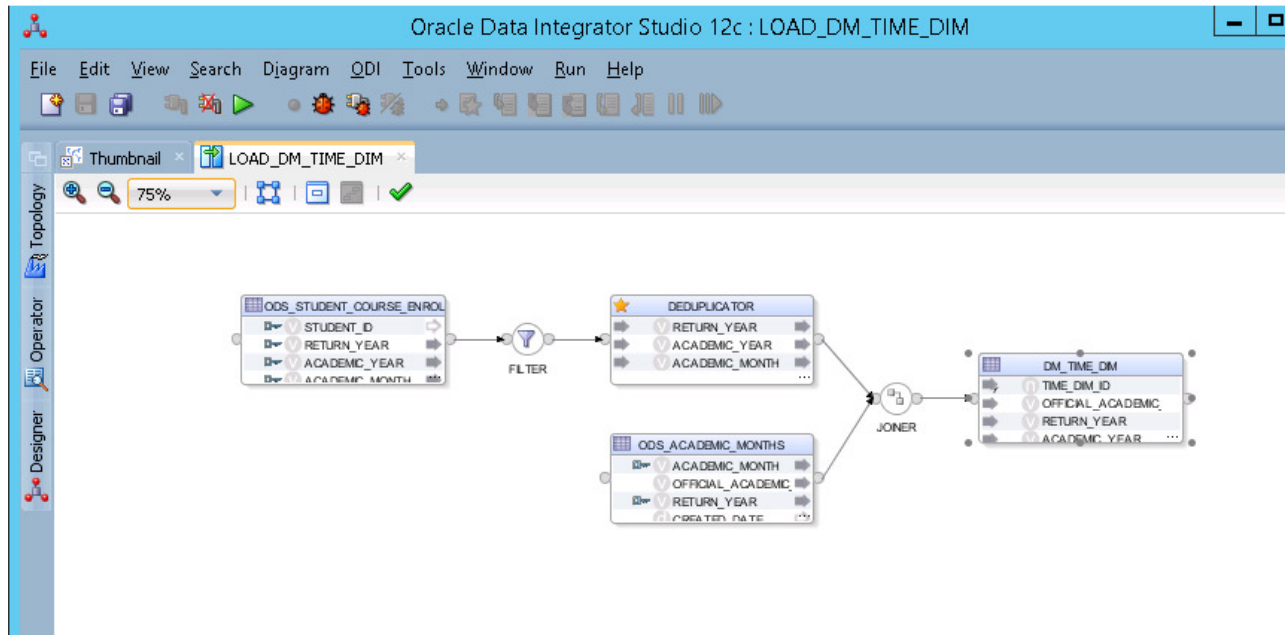




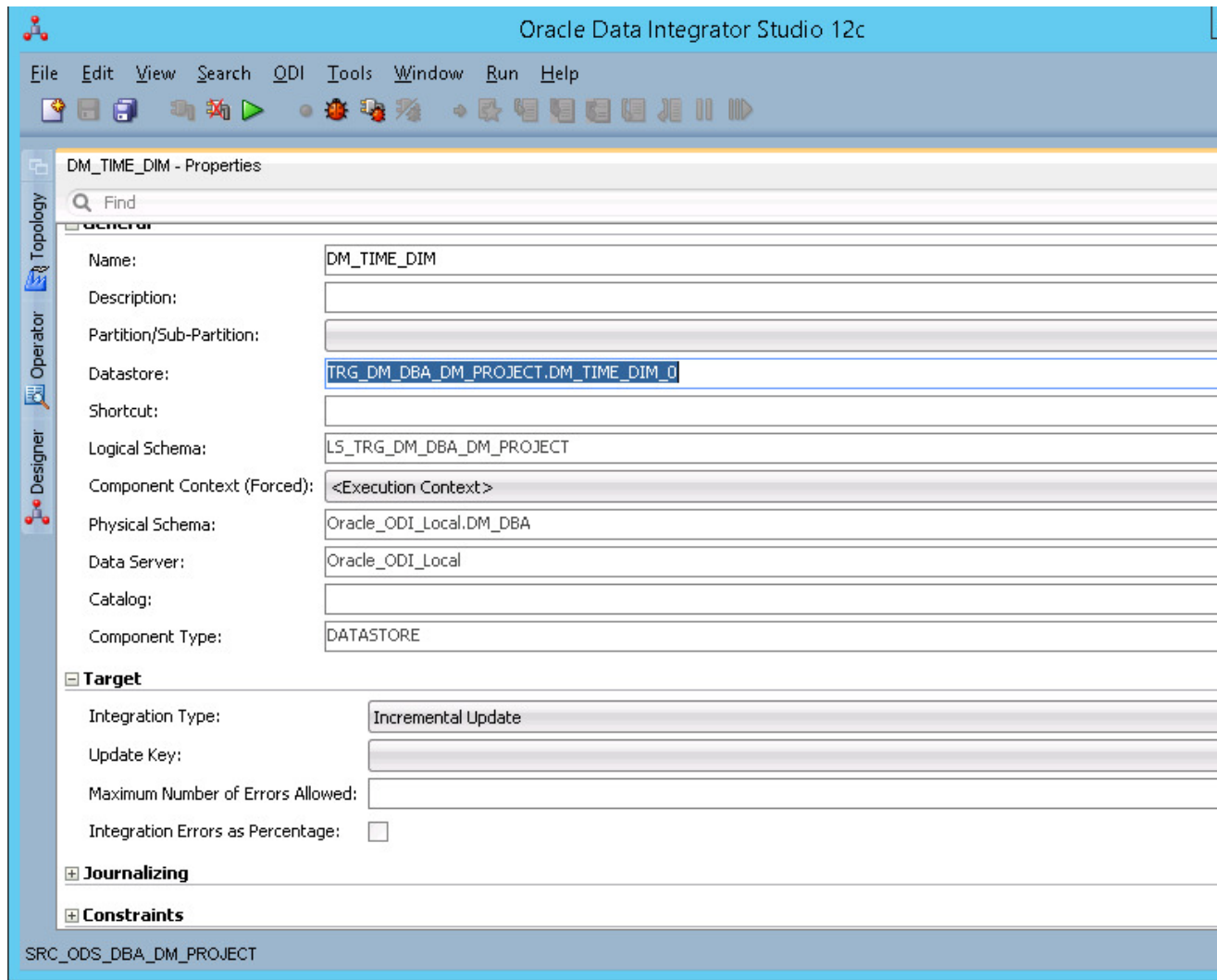
9. We can see the Logical Schemas, but there is no connection with any Physical schema. So now I will create the Physical Schemas to assign them to my logical ones.



10. I check the module LOAD\_DM\_TIME\_DIM.

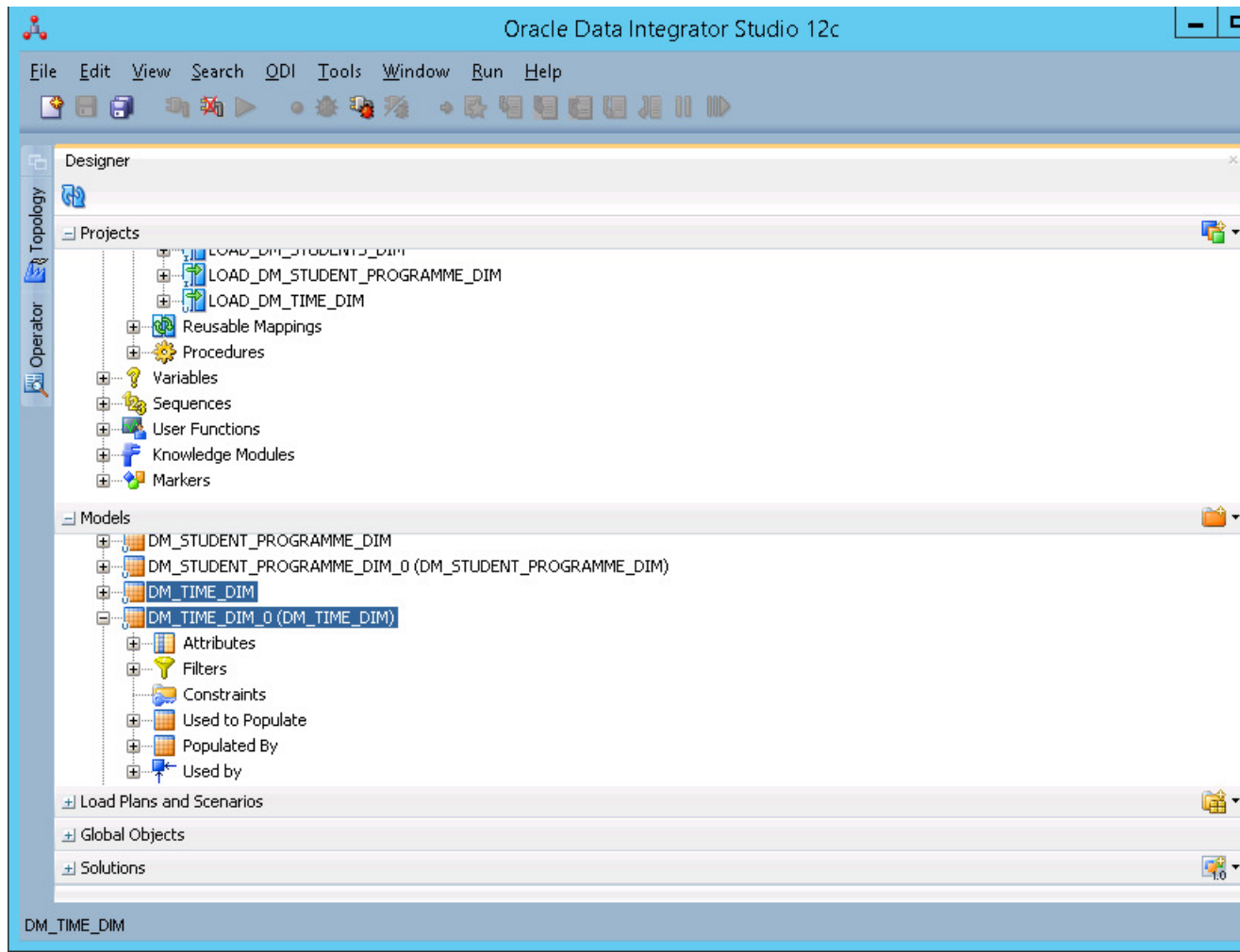


11. As I know that for this module I have set up a different primary key than the one that exists in database (for correct update or insert while loading data) I check the target table's DM\_TIME\_DIM properties

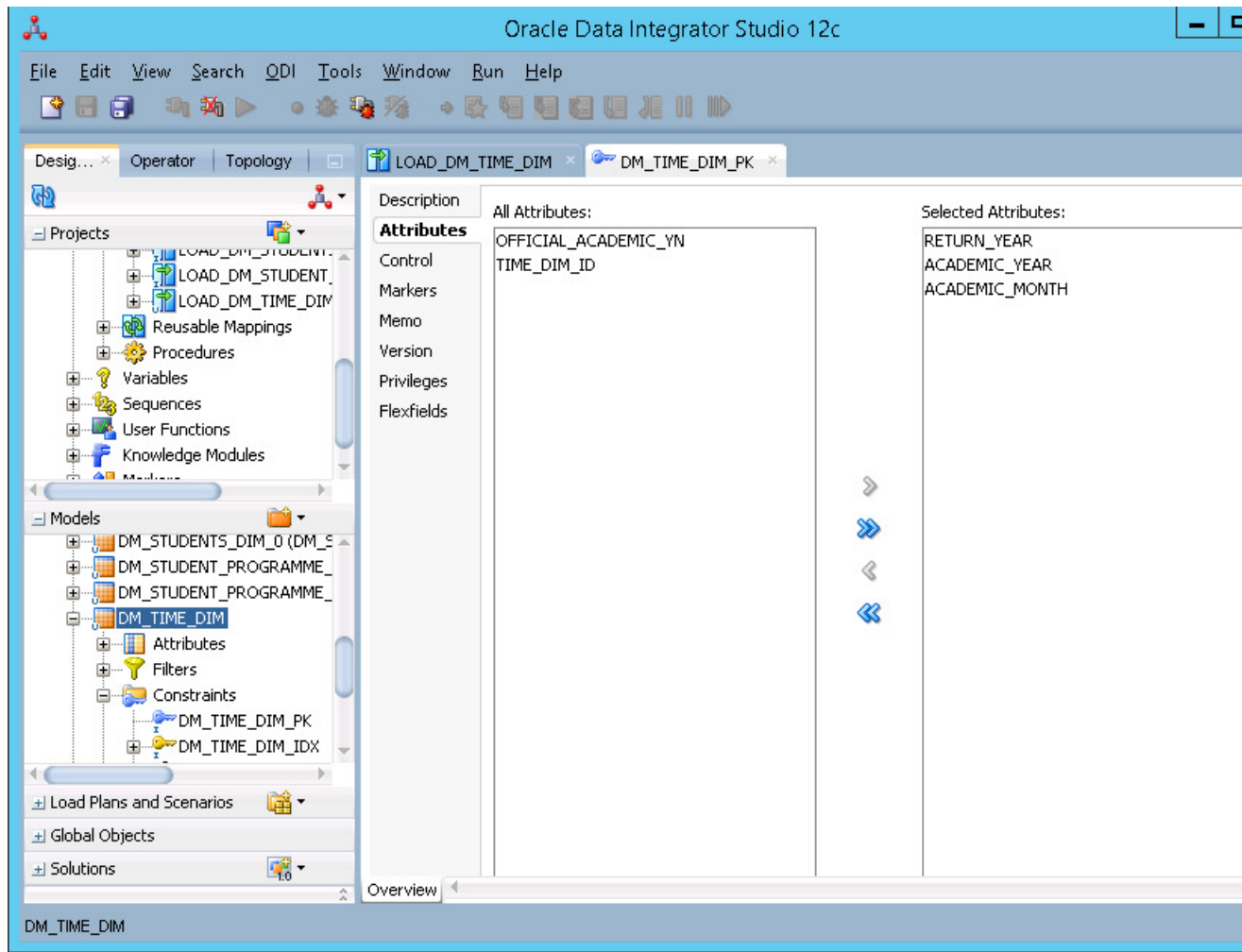


Here there is an issue as I see that the value of Datastore is different than the initial value, and the Update key is Null.

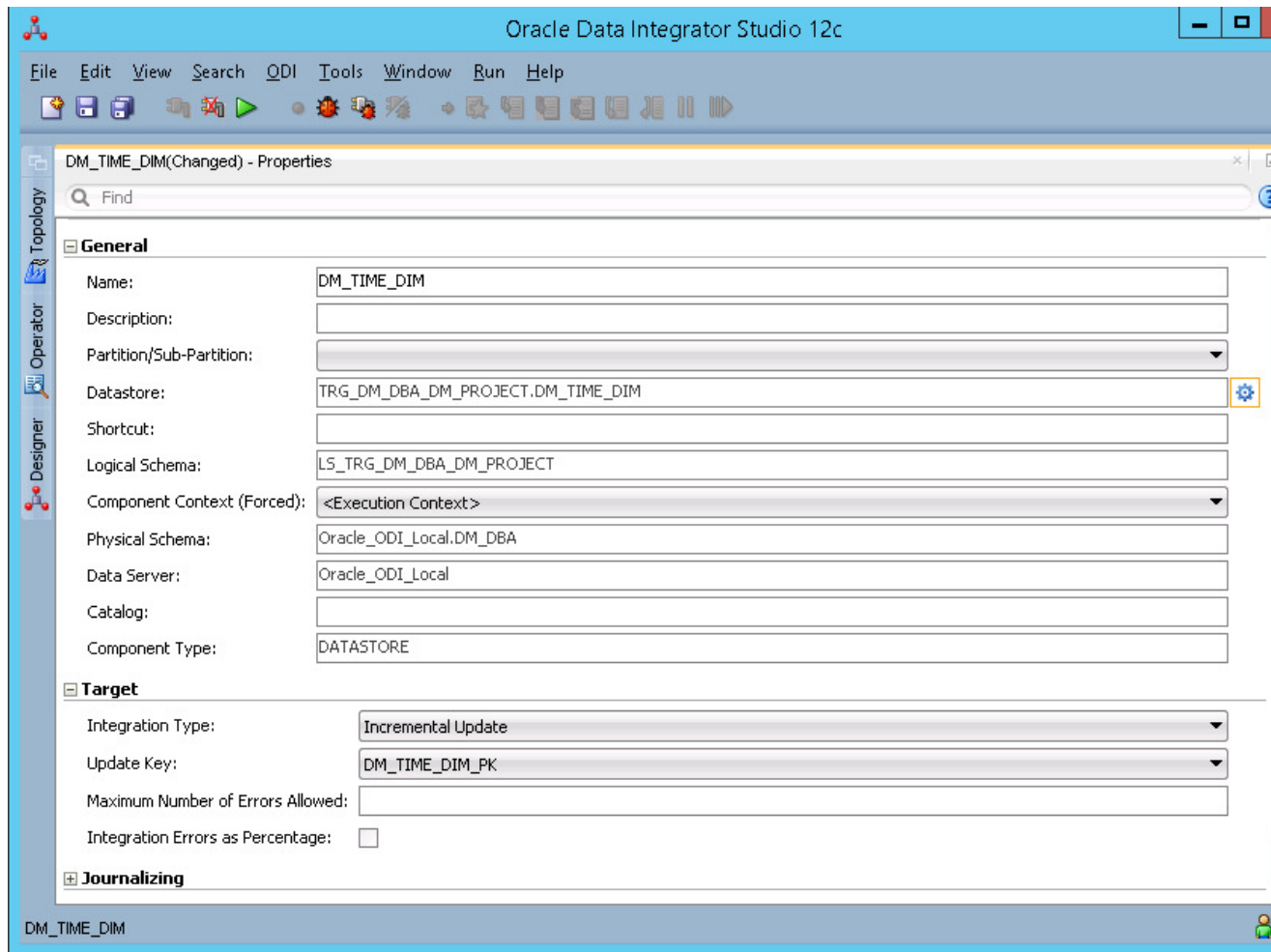
Checking the model I see that a duplicate table has been created for DM\_TIME\_DIM with no constraints:



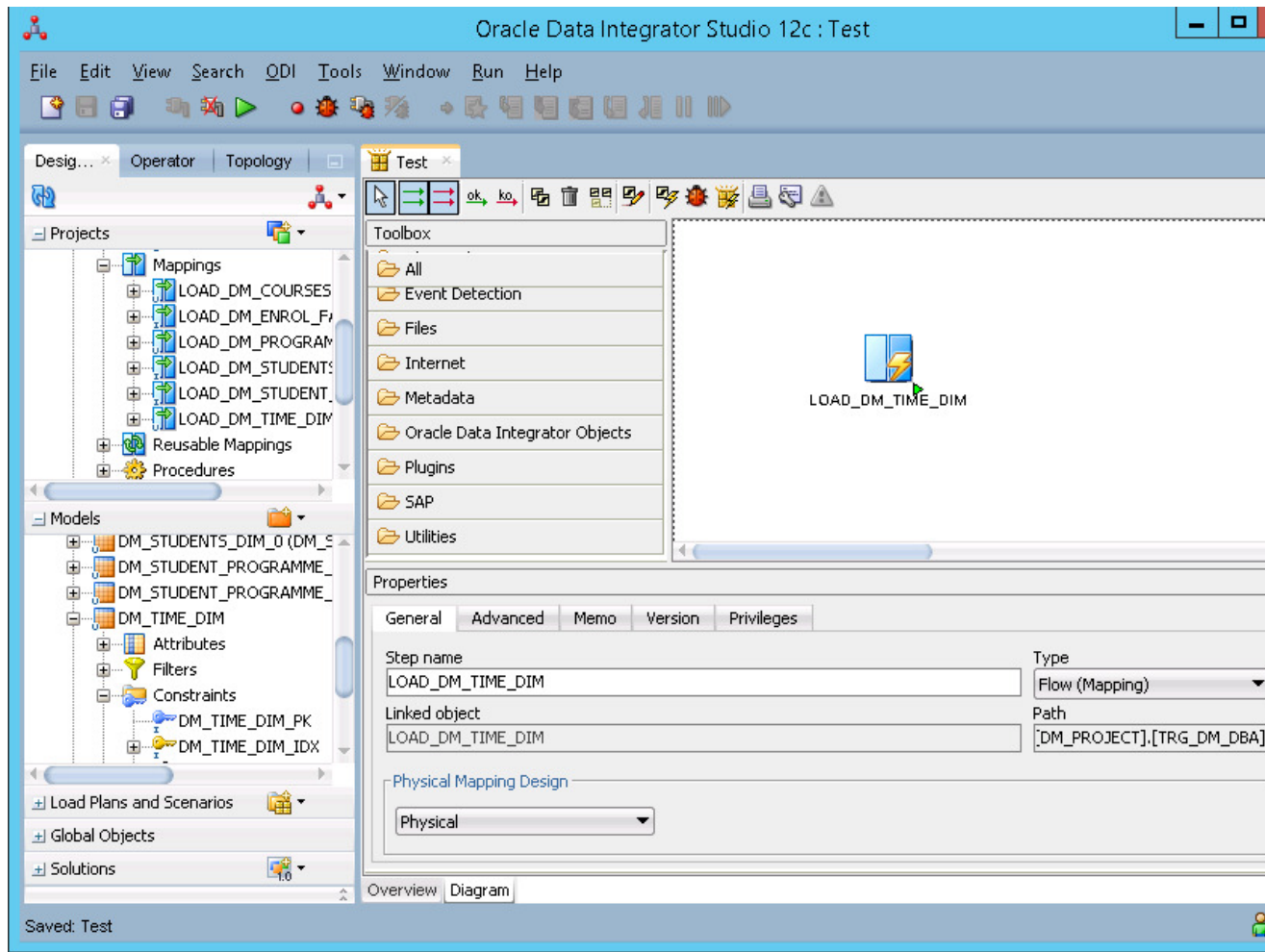
The original one has the primary key I had define in the OWB.



12. I go back to the module and modify the values of datastore and update key at the target table:

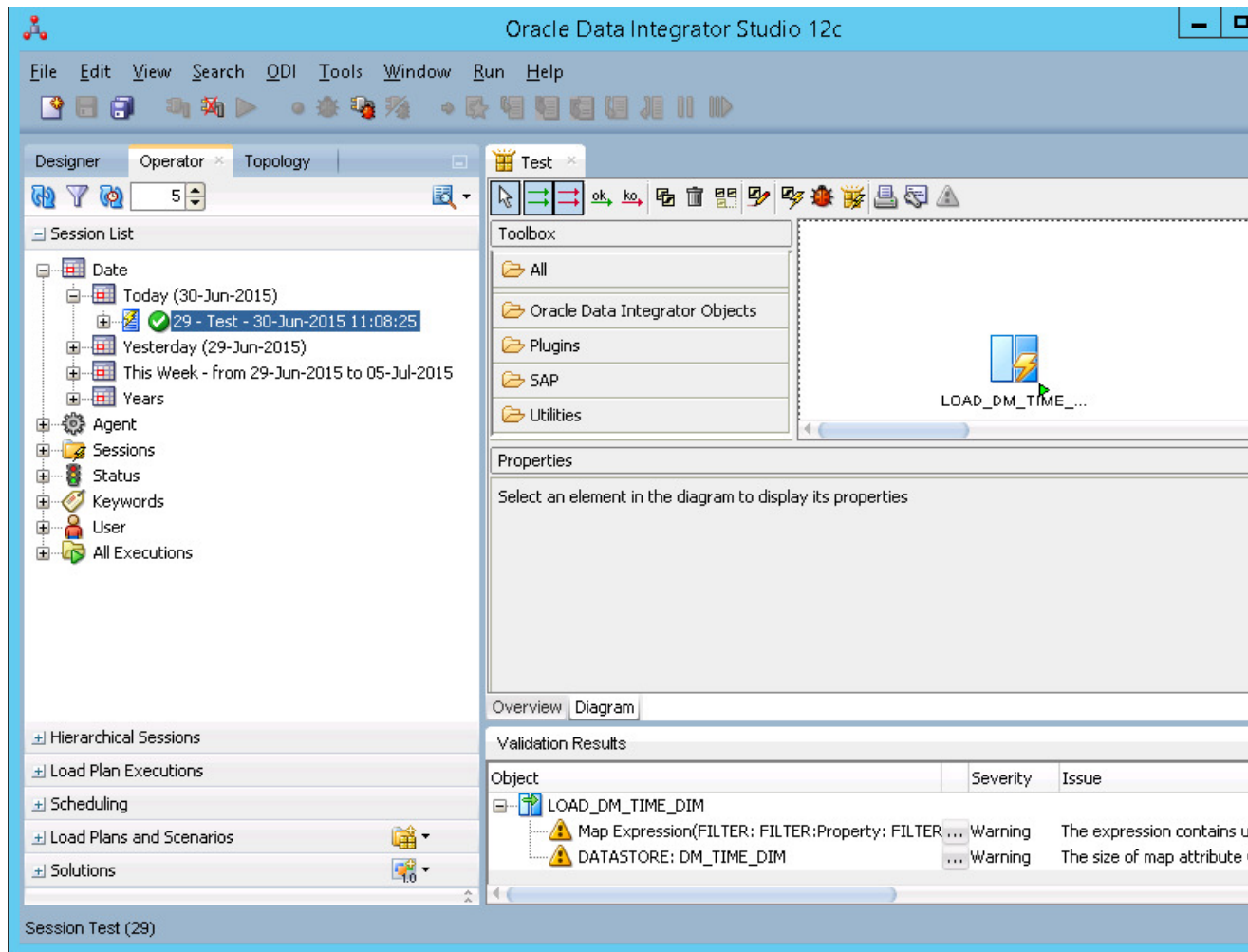


13. Now I will create a package for the specific module and try to run it.



14. And here are the result: SUCCESS





15. I do the same for the rest of the modules.

With the exception of the issue outlined in step 11 above, the OWB to ODI migration process runs smoothly and greatly simplifies the migration process.