**Anton Slizh’s**

**U2M10.LW.ETL Overview - Load and Transformation**

**Task 1**

## 2.1. Task 01: Transformation Description

Oracle gives us the following choices for transforming data inside the database:

* Transforming Data Using SQL
* Transforming Data Using PL/SQL
* Transforming Data Using Table Functions

In the building business model, the best solutions are using the SQL and PL/SQL transformations.

The data transforming and loading processes in the preparing DWH often use the small tables and simple transformations. These actions can be successfully completed by using standard functionality of INSERT, UPDATE, MERGE statements. Also, I should note that I have used very often the ‘UPSERT’ functionality to INSERT new rows into the table and UPDATE existing rows. The MERGE statement is really good choice for this action.

Sometimes for realization more complex transformations the standard SQL functionality is not enough. In preparing DWH for more complex and large tables such as sales or products (with SCD2 implementation) more efficient and logically simpler is to use the PL/SQL statements. For example, a PL/SQL procedure could open multiple cursors and read data from multiple source tables, combine this data using complex business rules, and finally insert the transformed data into one or more target table. It would be difficult or impossible to express the same sequence of operations using standard SQL statements.

**Task 2**

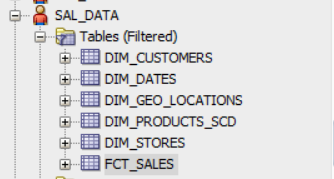
## 3.1. Task 02: Loading to SAL Layer Data

**The Main Task** is to load dimension to SAL layer

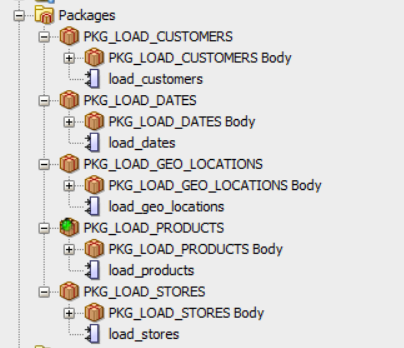
**Required points:**

* Create new package for Load FCT\_\* and DIM\_\* to SAL Layer
* Load Dimension
* Load SCD Dimension
* Load FCT\_\*

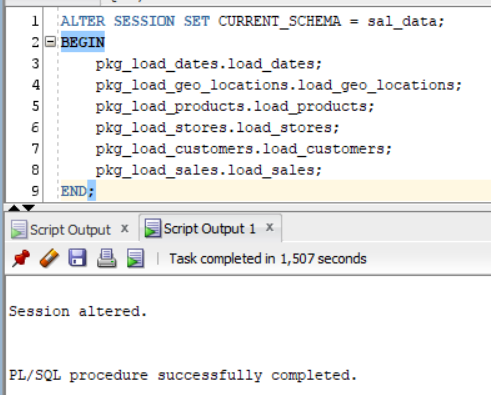
First step is to create SAL layer objects (Dimension and Fact tables) using prepared scripts.



After initializing tables we should to transform data from the DW layer into Star Layer using prepared procedures.

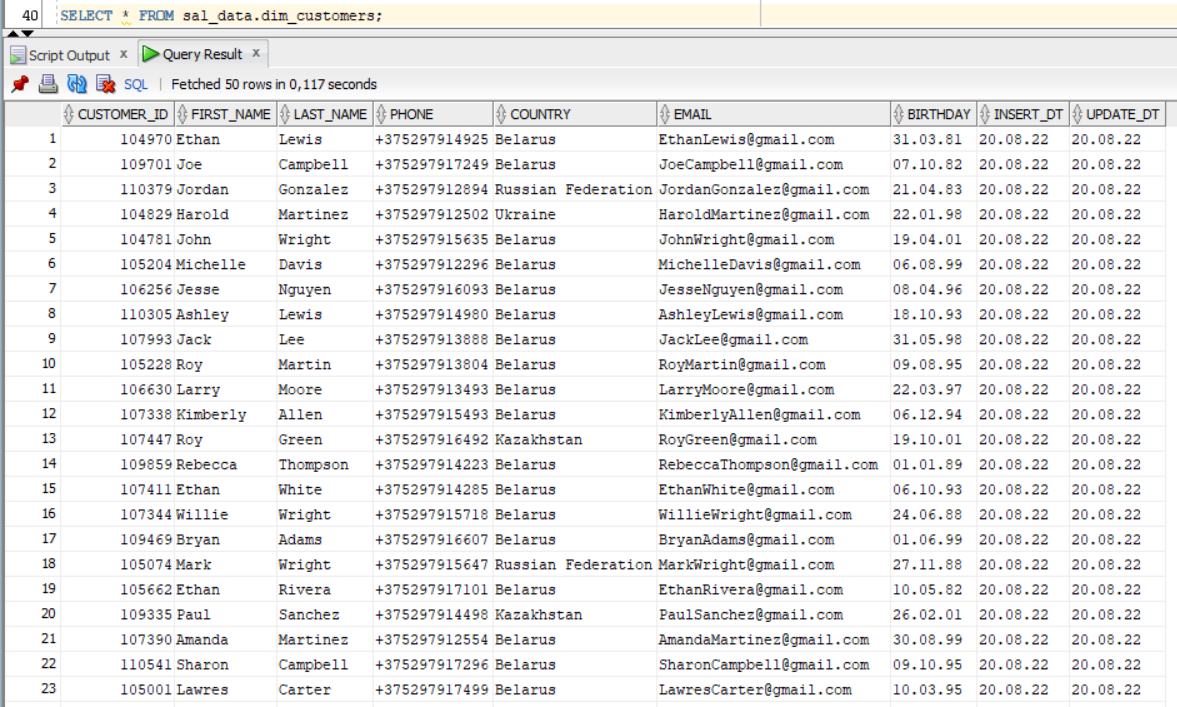


Executing procedures using *load\_data* script:

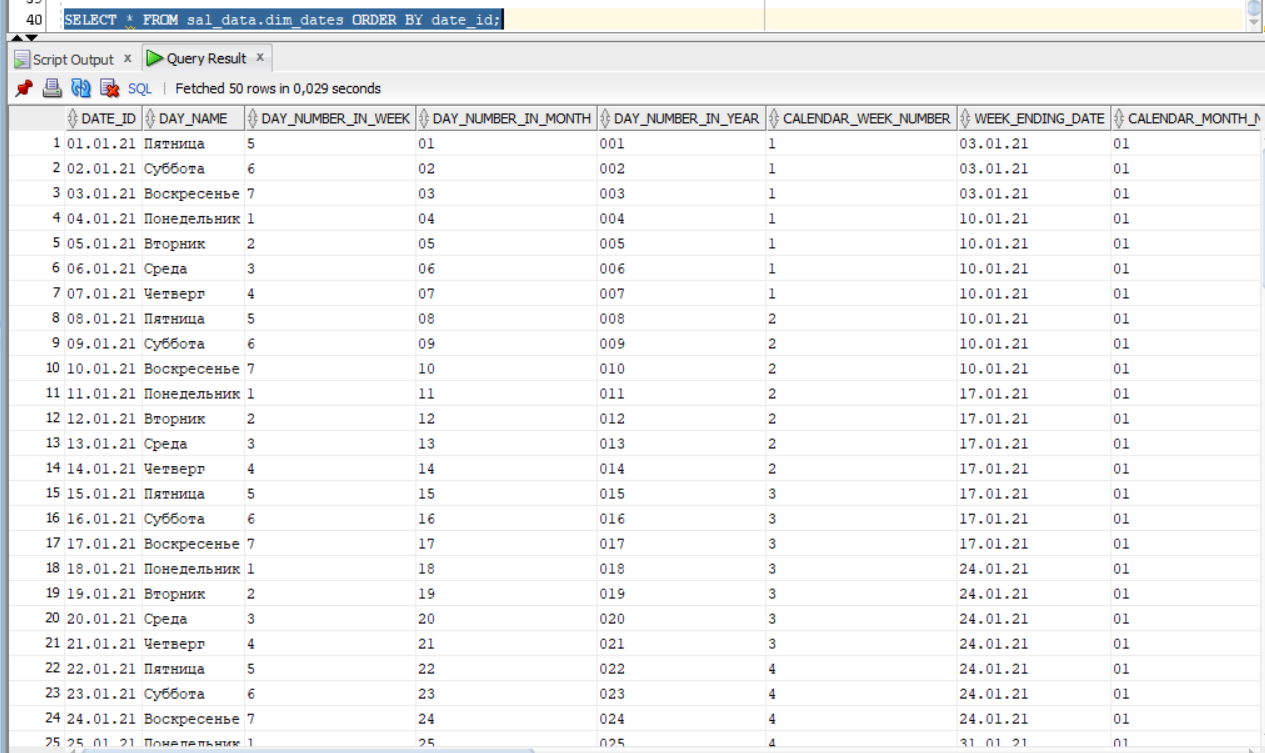


So, let’s look at the data of our final Star Schema:

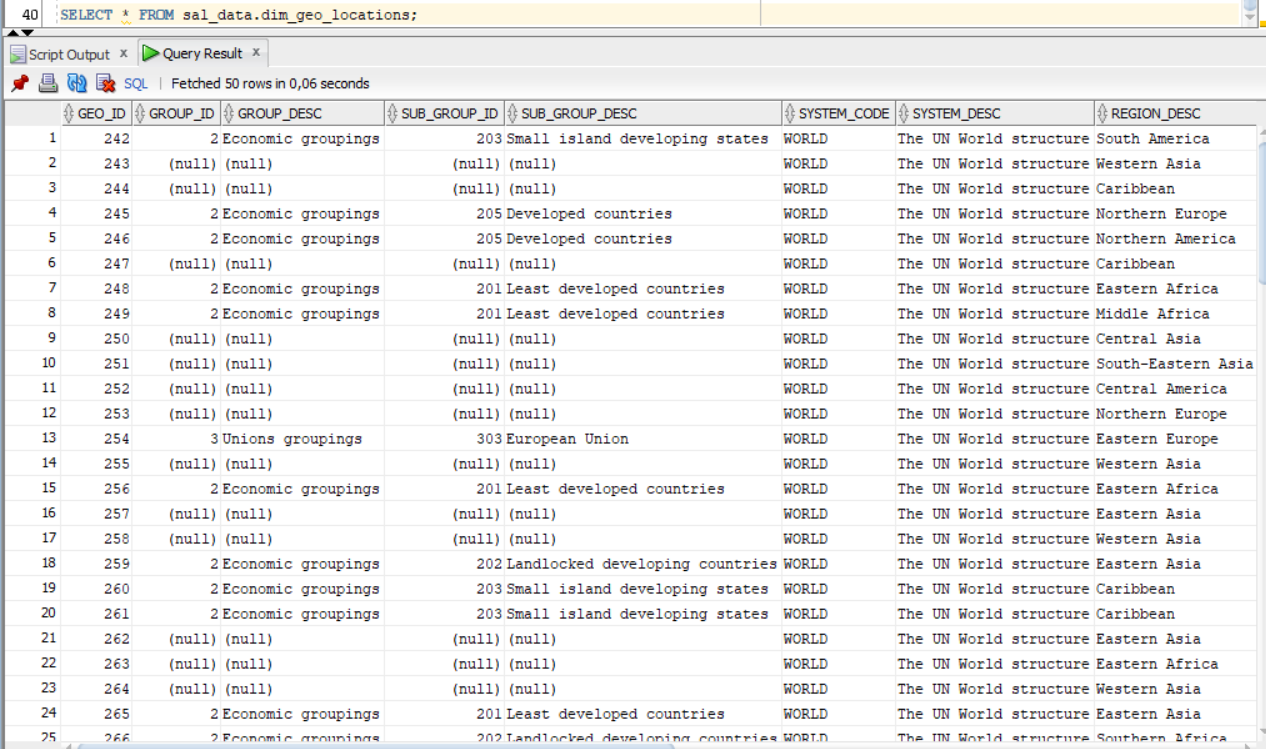
DIM\_CUSTOMERS



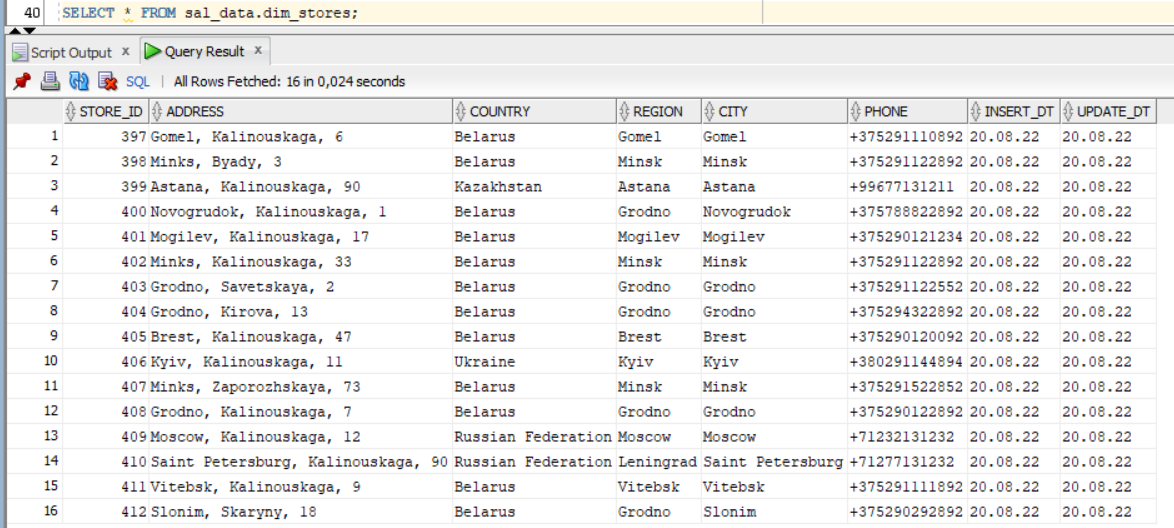
DIM\_DATES



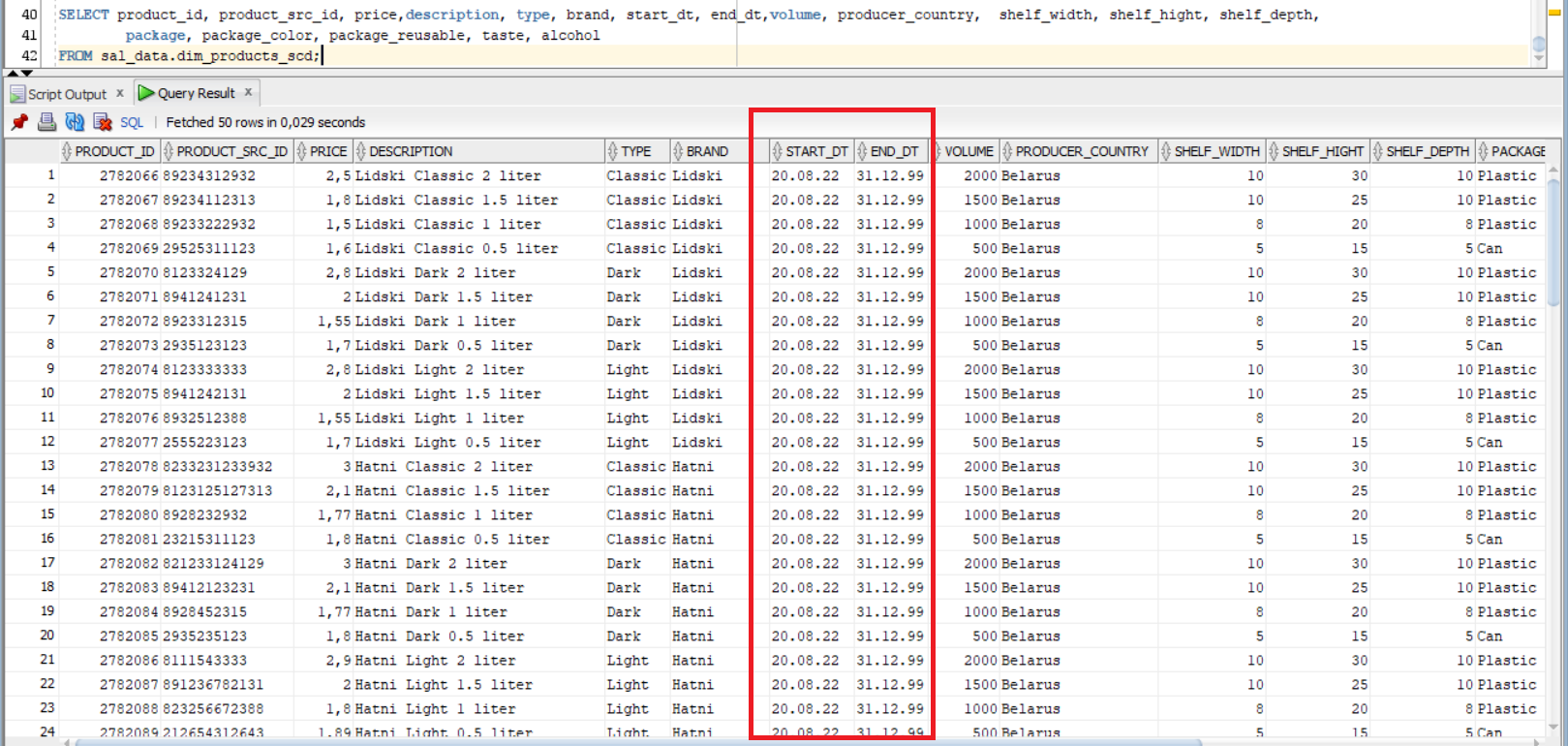
DIM\_GEO\_LOCATIONS



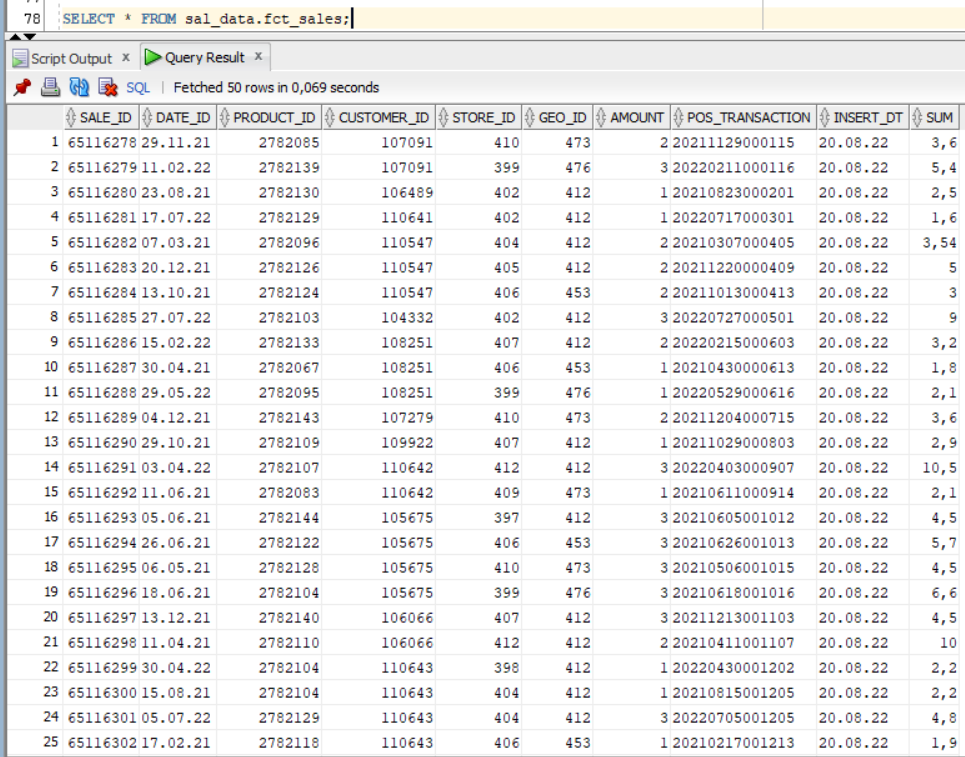
DIM\_STORES



DIM\_PRODUCTS\_SCD



FCT\_SALES

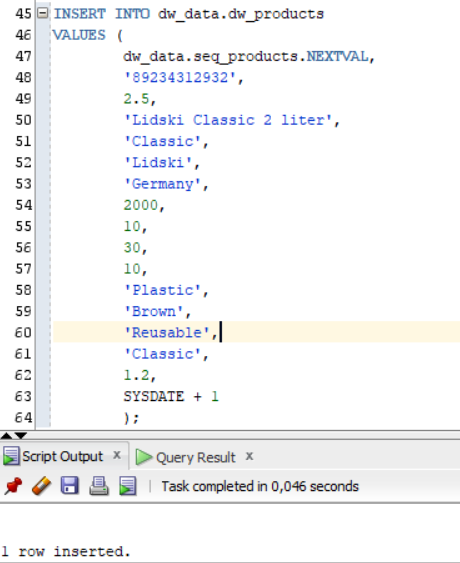


The DIM\_PRODUCTS\_SCD table is SCD2 type dimension. So, for each row we have the start date and end date which represent the time period of being actual. The surrogate key for each product is *product\_id* and the natural key is *product\_src\_id.*

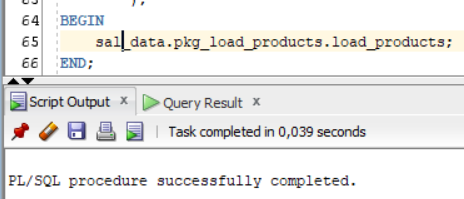
Let’s try to update any product and show the SCD2 mechanism in use. For example, choose the first product in the screen – Lidski Classic 2 liter (good choice by the way).

Product\_id – 278066, Product\_src\_id – 89234312932.

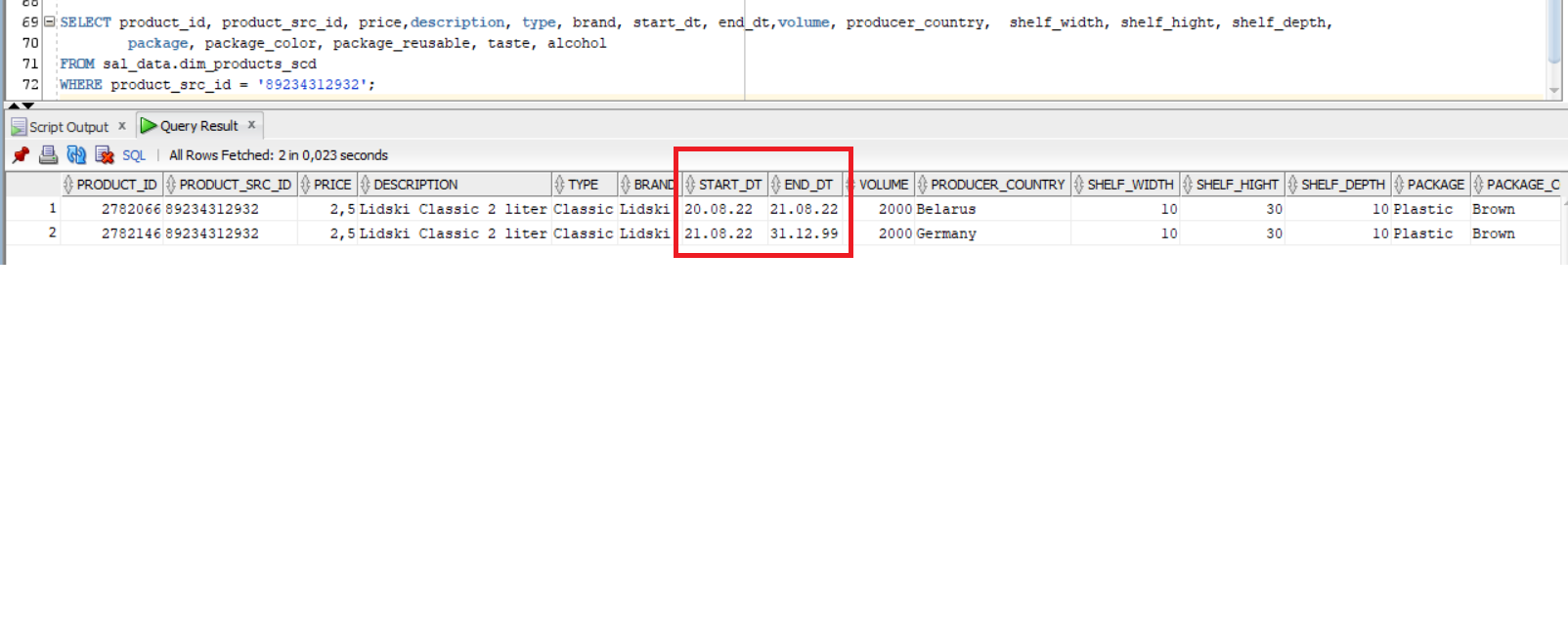
Manually add to the DW layer new row with the same *product\_src\_id* attribute value, but with the updated *producer\_country* attribute value and the tomorrow *insert\_dt.*



Now let’s reload products dimension data.



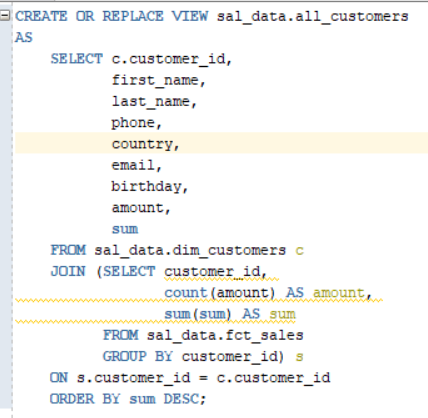
Look at our updated product at the dimension:

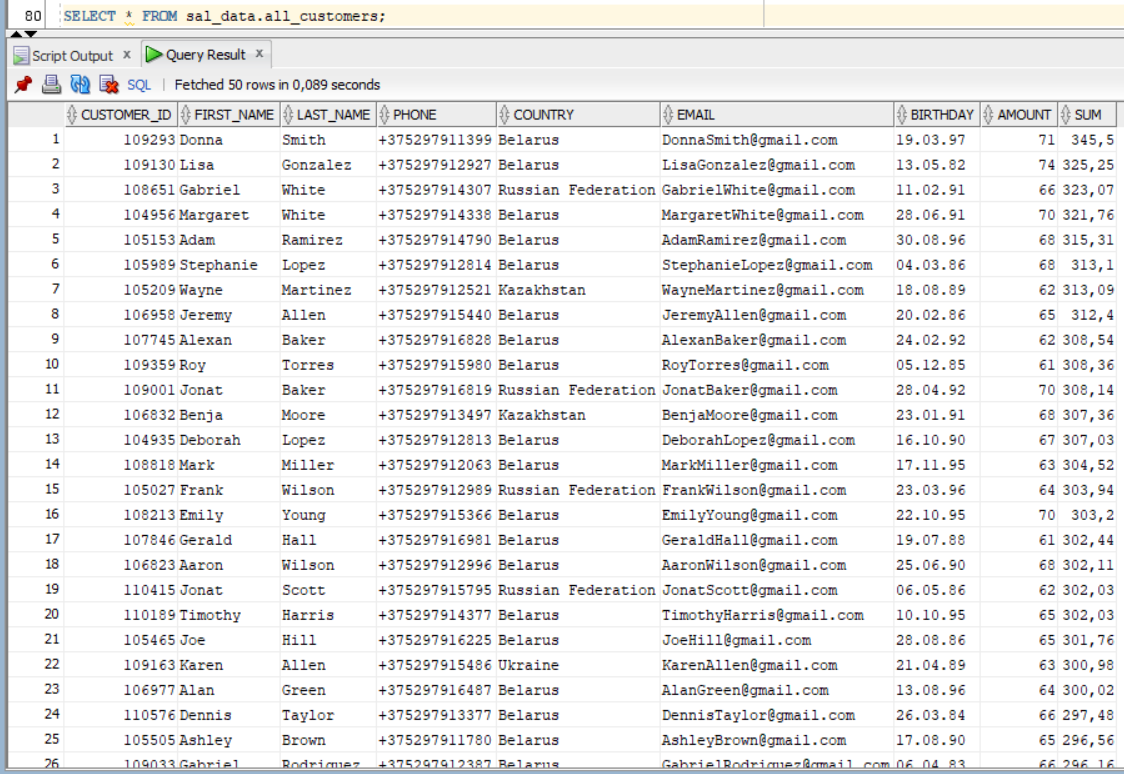


We can see that new row became actual for this product (*product\_src\_id*). So, this way we can store the history of changes in our product dimension.

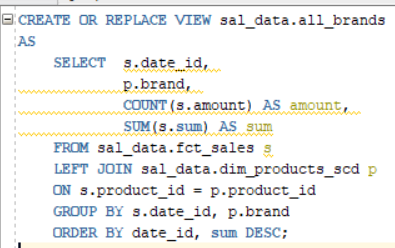
Also let’s create some Data Marts on our Star Scheme. I have used the VIEWs which select all necessary data from prepared Star Scheme.

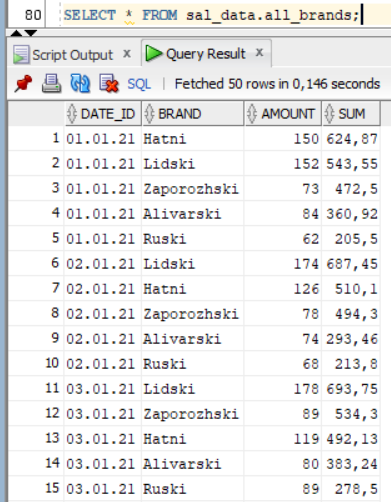
DM\_ALL\_CUSTOMERS



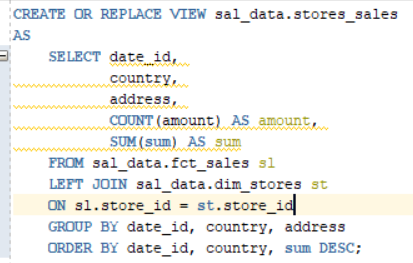


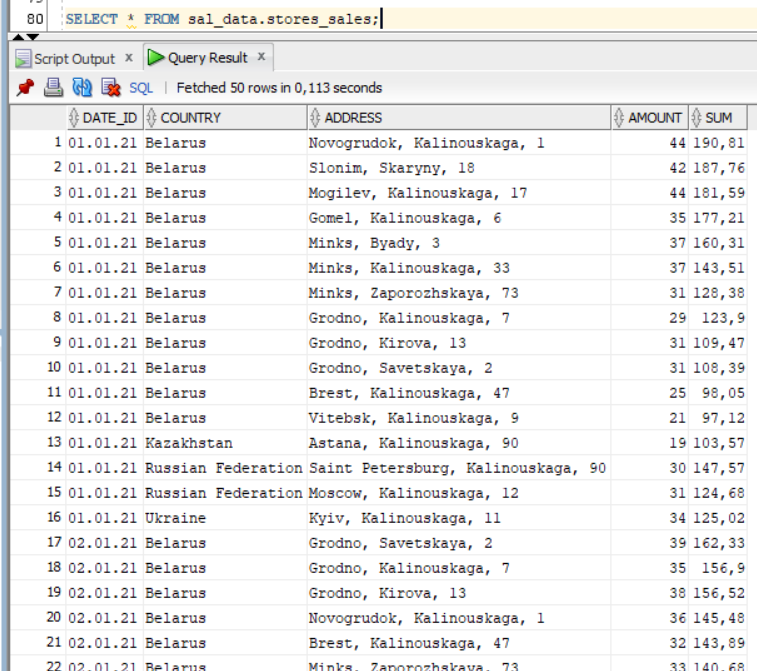
DM\_ALL\_BRANDS





DM\_STORE\_SALES





And at the end let’s create the script which join together all load procedures to execute whole ETL process.

