**Anton Slizh’s**

**U3M2.LW.Developing Integration Services Solutions**

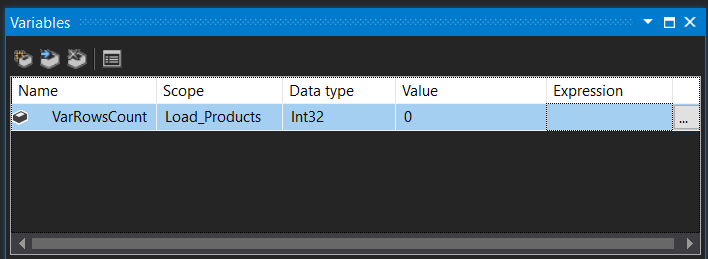
*GitHub:* [*https://github.com/drapejny/DataCamp2022/tree/master*](https://github.com/drapejny/DataCamp2022/tree/master)

*It’s funny that Visual Studio use English and Russian localization at the same time. I even didn’t load Russian Language packet. At some screenshots you will see it.*

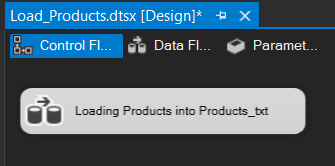
**Task 1**

## 1.1. Exercise 1: Creating an Integration Services Project and implementing a package

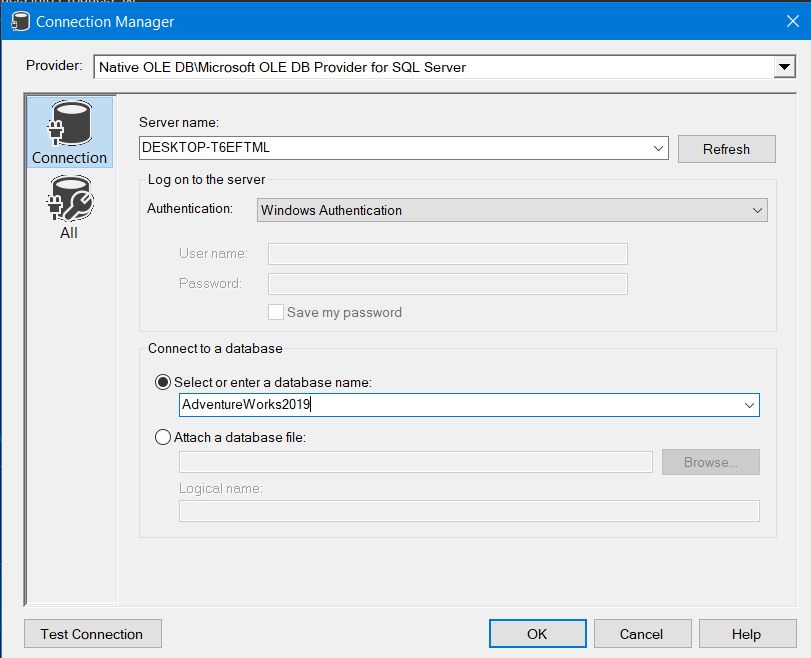
Creating package scope variable, which will contain the number of exported rows.

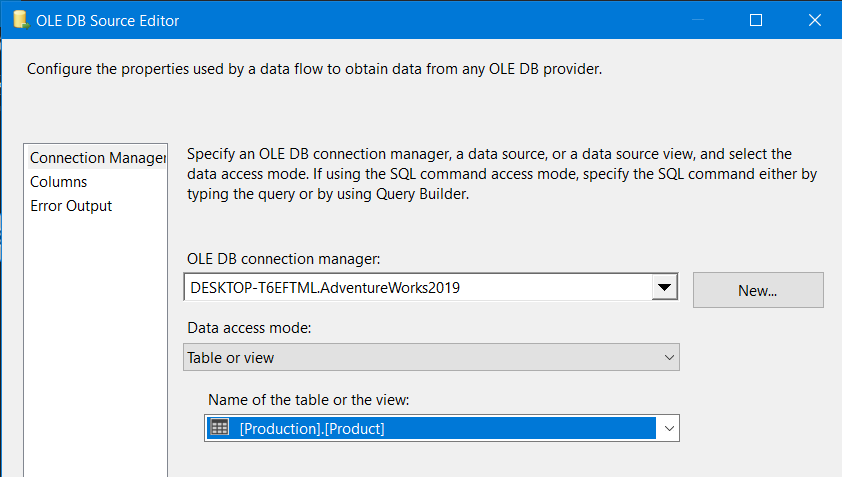


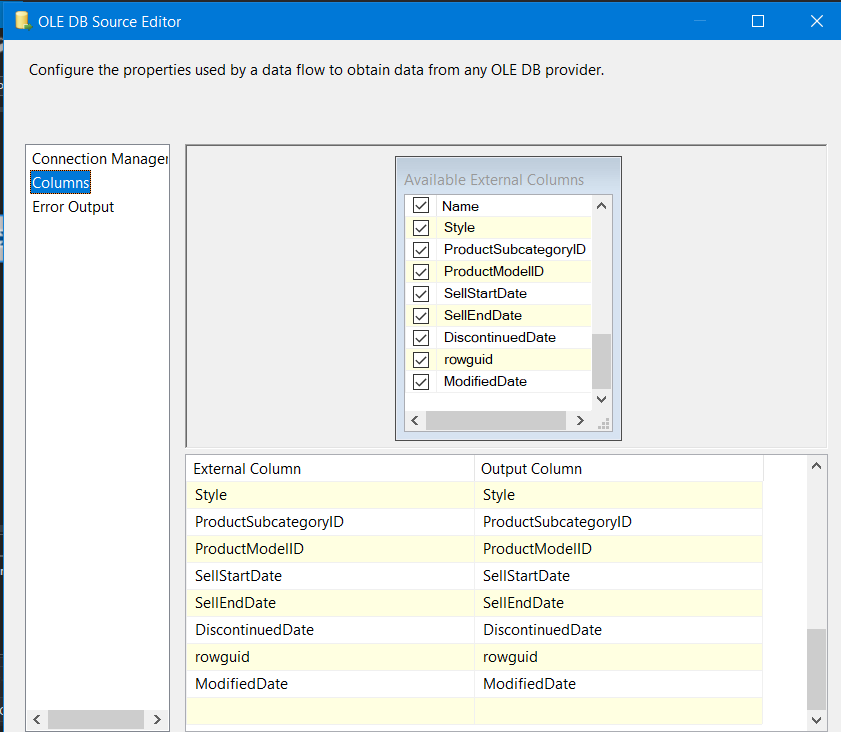
Creating the Data Flow Task to export data about products from Database into the Flat File.



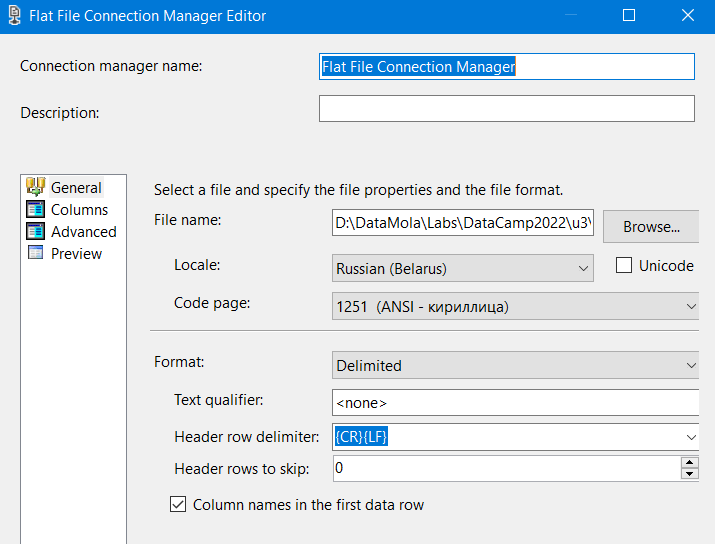
Creating Source OLE DB.

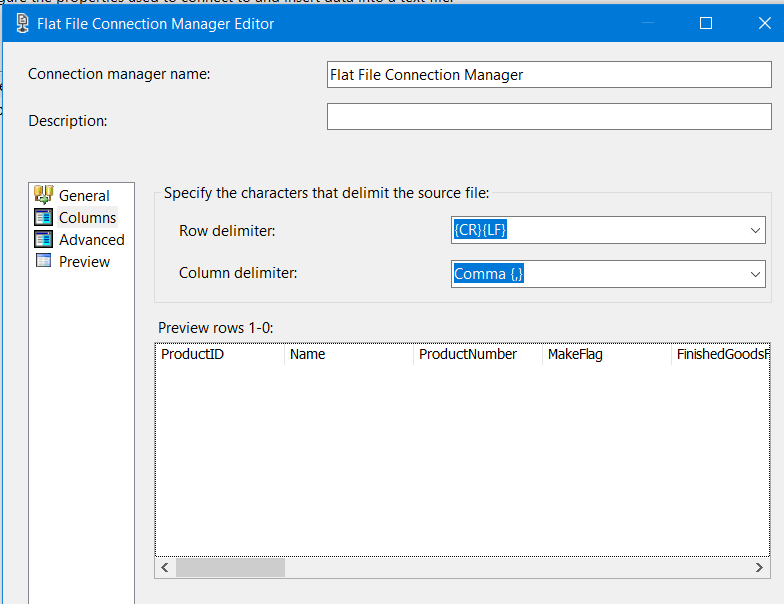


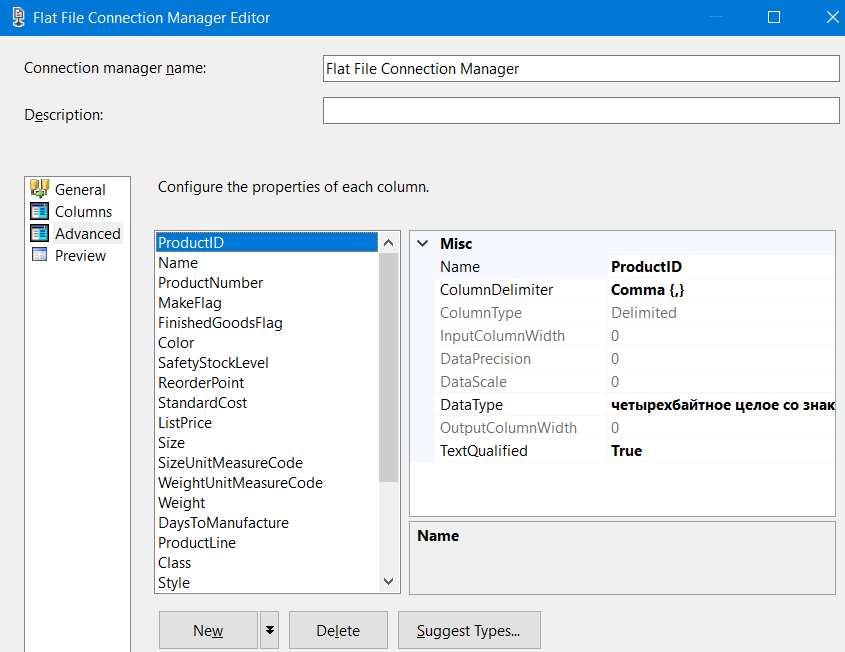


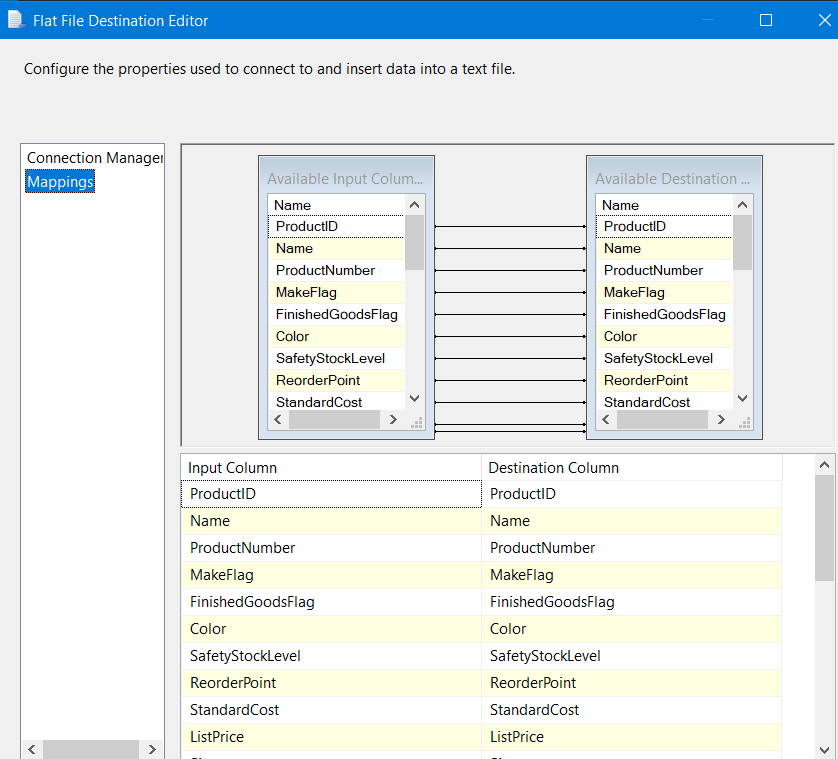


Creating Destination Flat File:

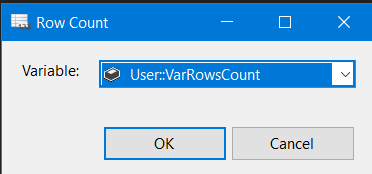
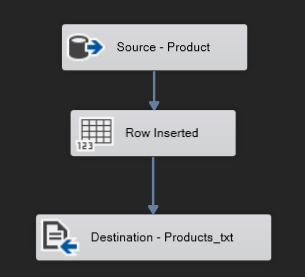




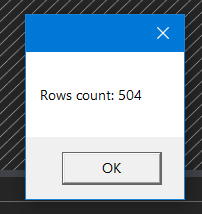
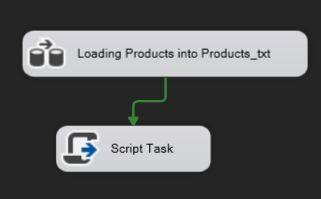




Now let’s create additional task between the Source and Destination which will count all inserted rows. The rows number will be written into recently created *VarRowsCount* variable.



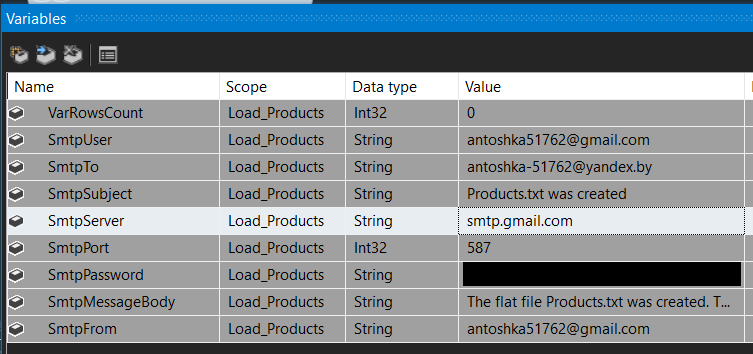
Let’s make sure that all working fine. Create Script Task and print the variable value.

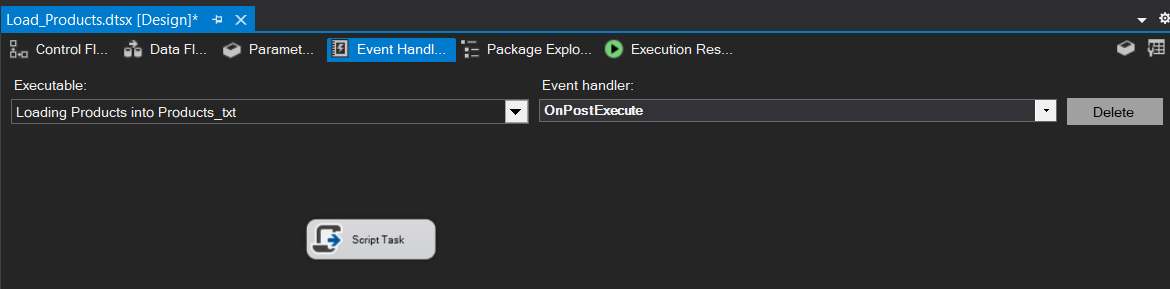


Now let’s create the Event Handler which will send the email after loading data into *Products.txt* file.

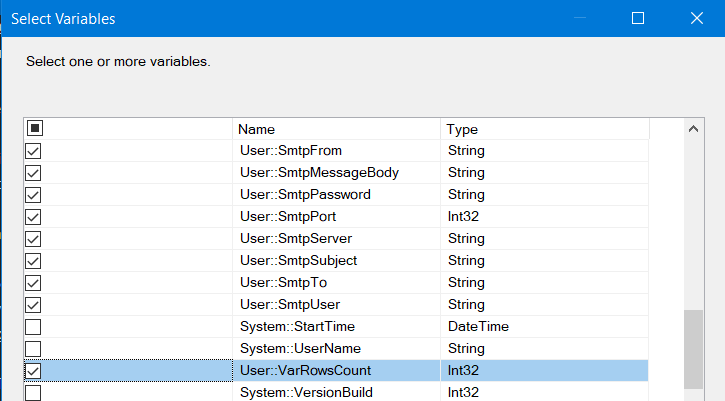
As the Send Mail Task doesn’t work with server which needs the credentials, we will write our own script to send messages.

Before writing a script we should create variables which contain our credentials and other email information.

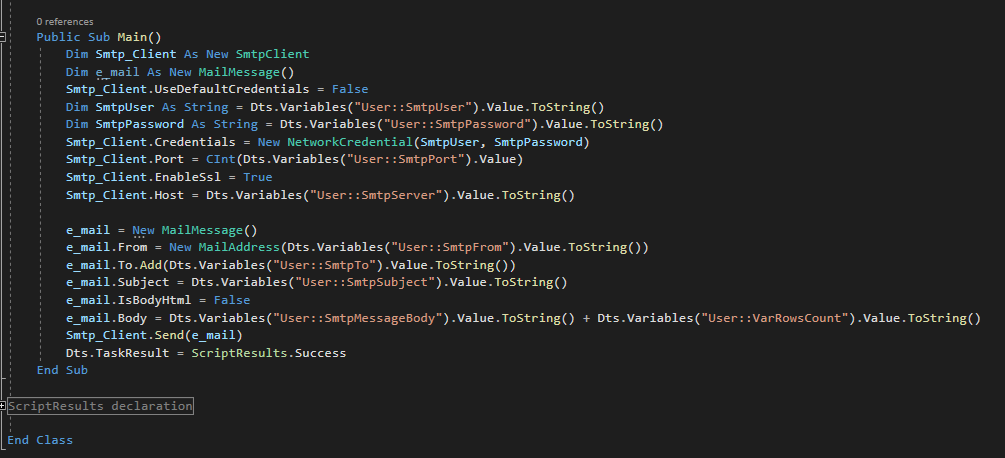


Creating the *OnPostExecute* Handler on *Loading Products into Products\_txt* executable. 

Selecting our variables to use them in the script

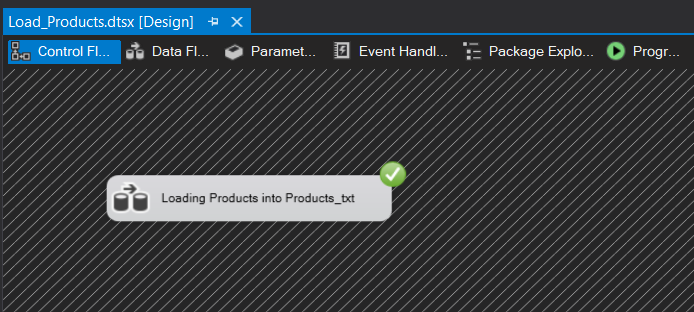


Writing the script code

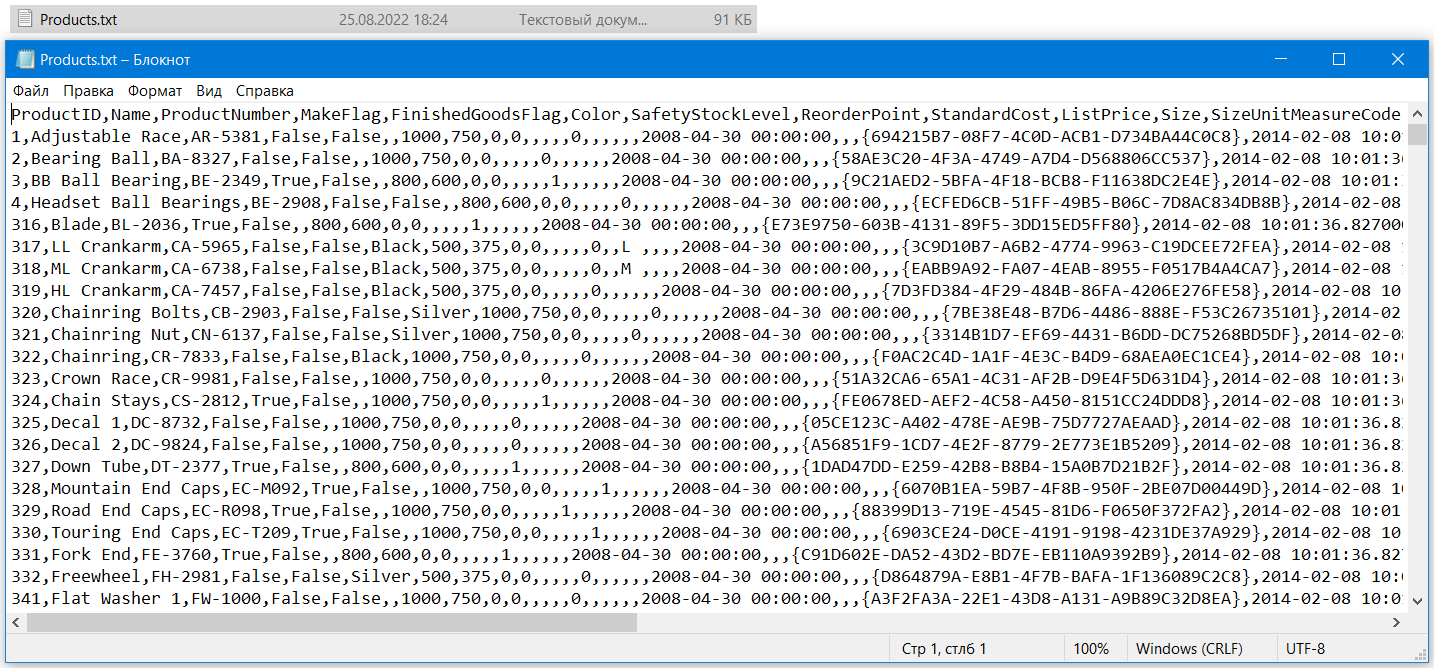


So, the task is completed.

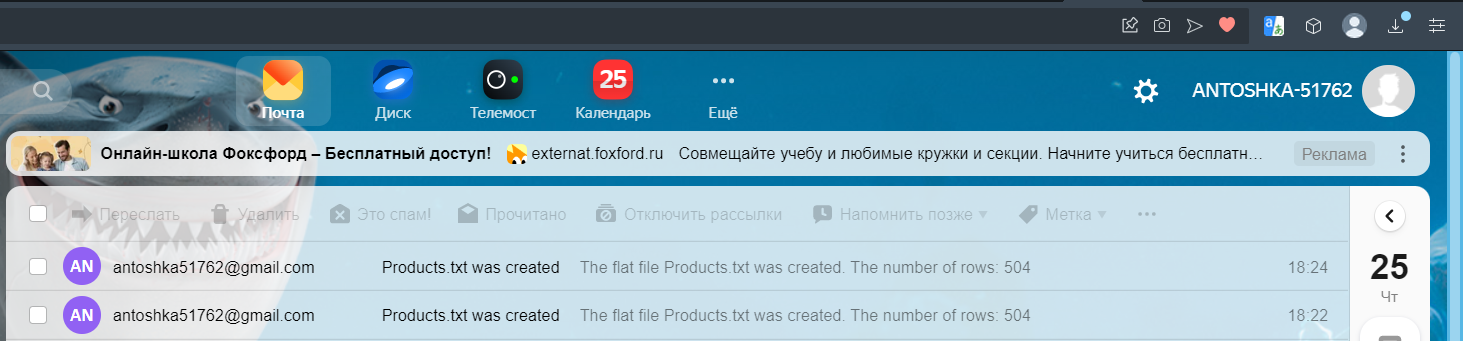
Executing the package:

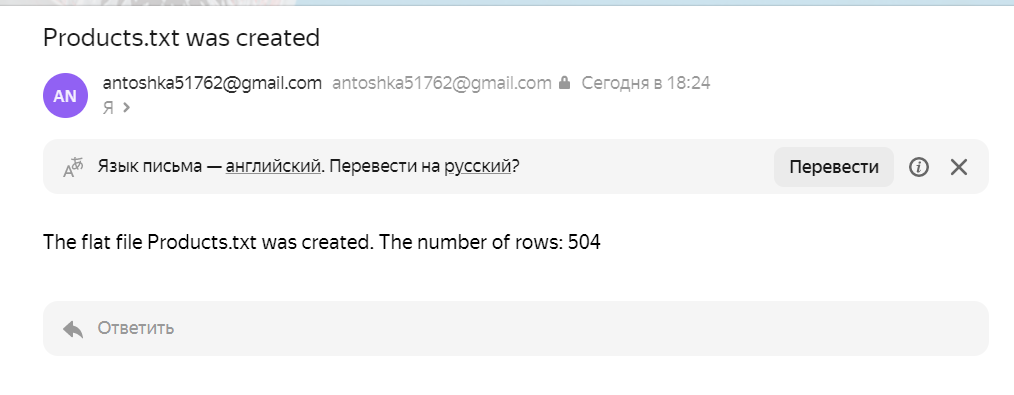


Looking at the data in the *Products.txt*



Looking at the E-Mail

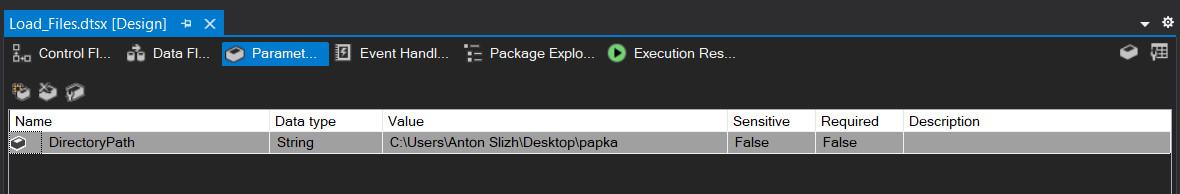




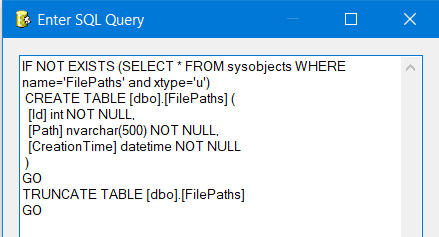
**Task 2**

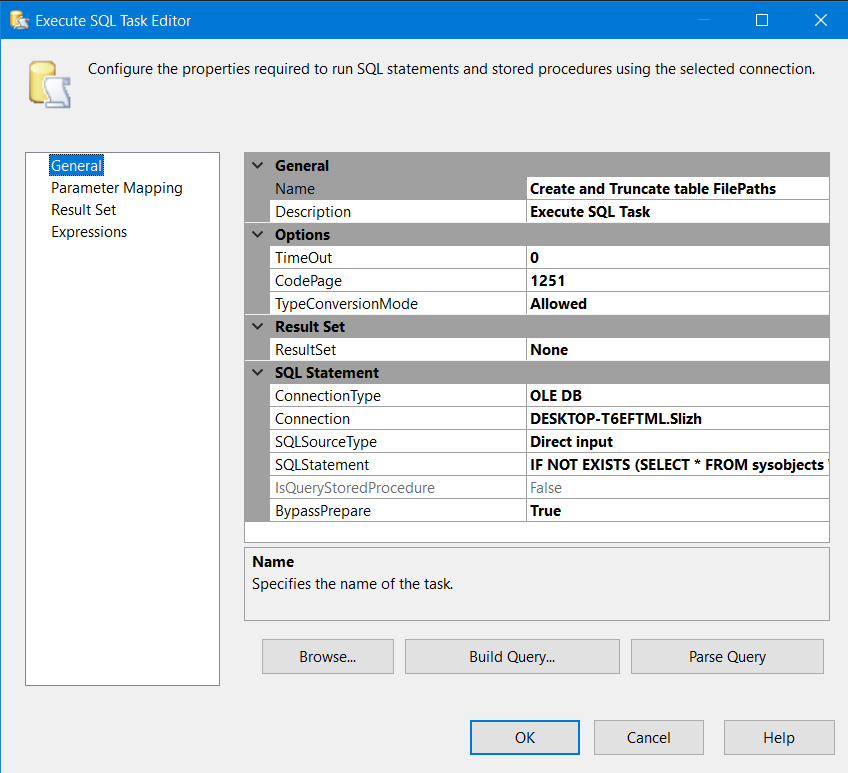
## 1.2. Exercise 2: Creating a package to keep list of files from a directory

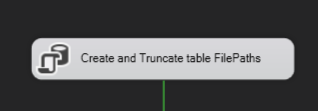
Creating the package parameter which represents the directory for file search.



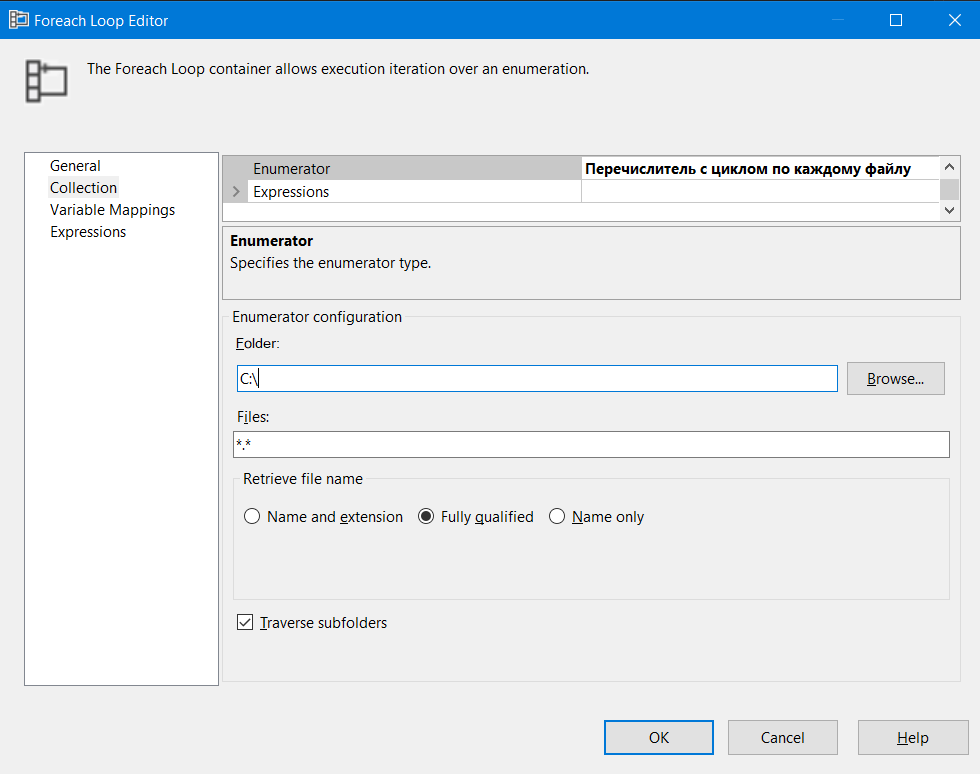
Then preparing the table in the database. Creating SQL Task which will create destination table (if not exist) and truncate it.

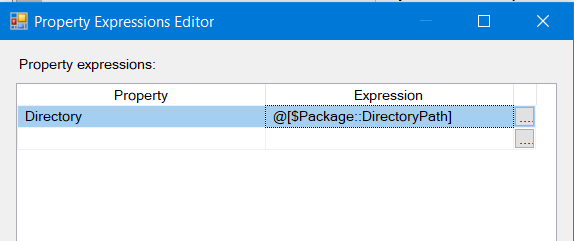




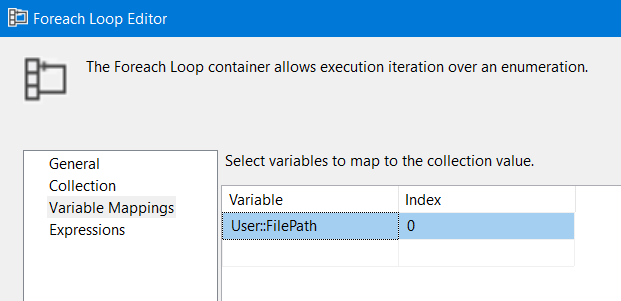


Then creating a Foreach Loop which will iterate through all files in the specified directory. The directory for file search is taken from package parameters.





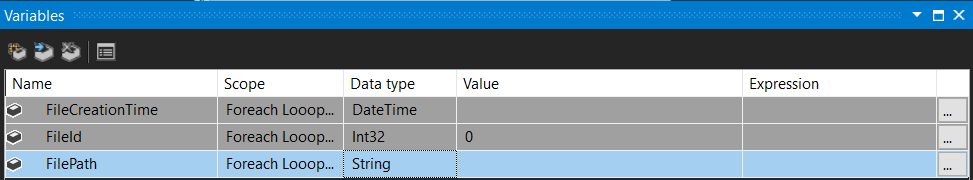
Creating the variable *FilePath* which will store the path for each file while iterating through the loop.



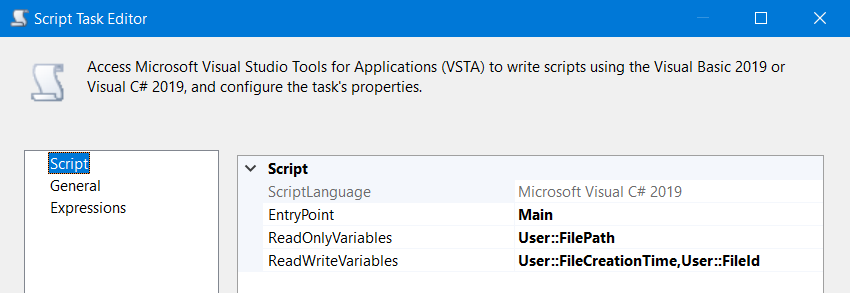
Then defining other variables at the Foreach Loop scope.

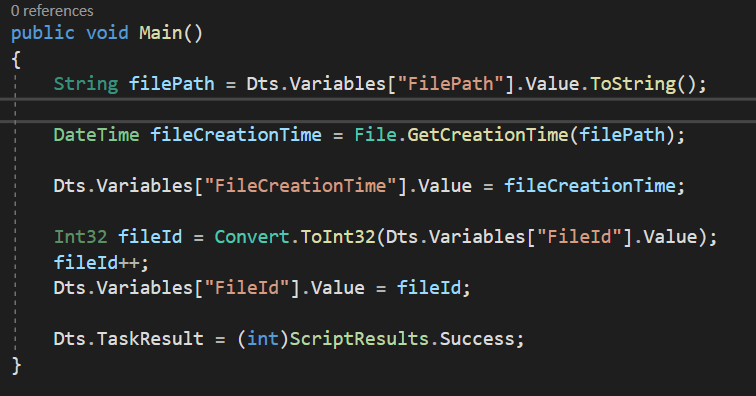
*FileId –* store calculated unique id for each file in the loop

*FileCreationTime –* store creation time for each file in the loop

**

Now, we should write information about file into defined variables. Let’s create the special Script Task which will update variables values.



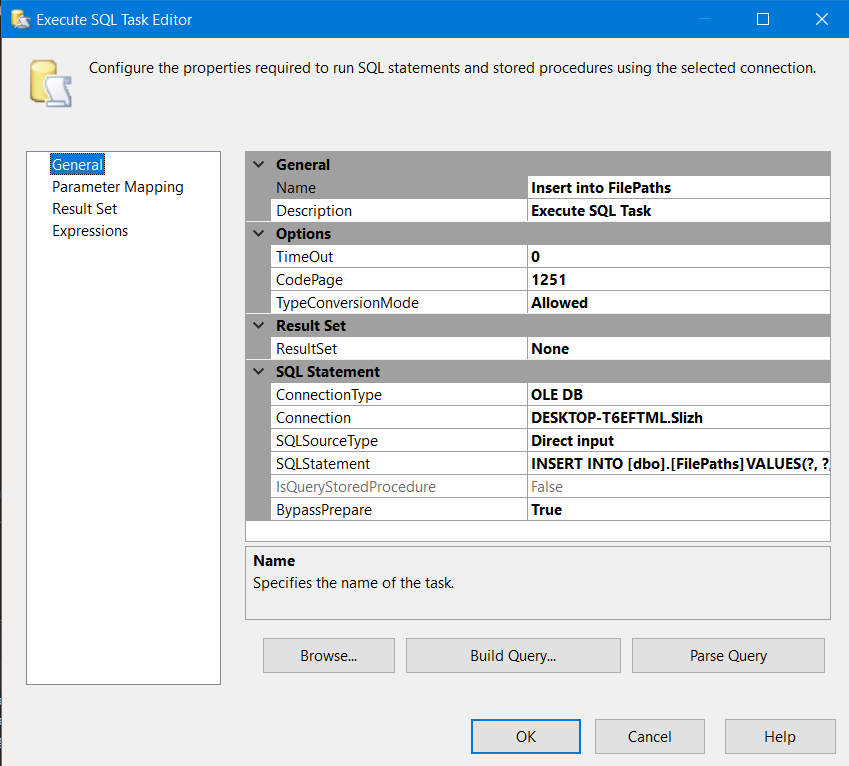


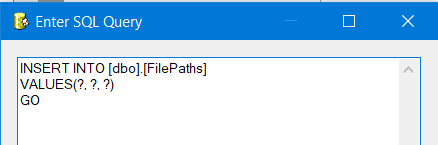
After script execution *FilePath* variable will still contain the path of the file. *FileCreationTime* variable will contain extracted creation time of the file and *FileId* will contain the calculated Int32 Id for the file.

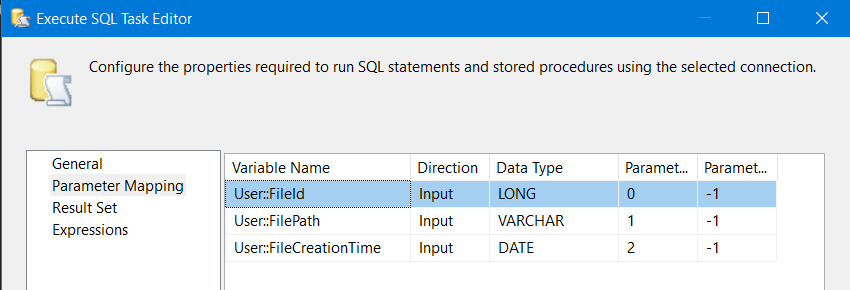
When all necessary data was written into the variables, we can insert data into the prepared table.

Creating the SQL Task which will insert one row into the table *FilePaths.*

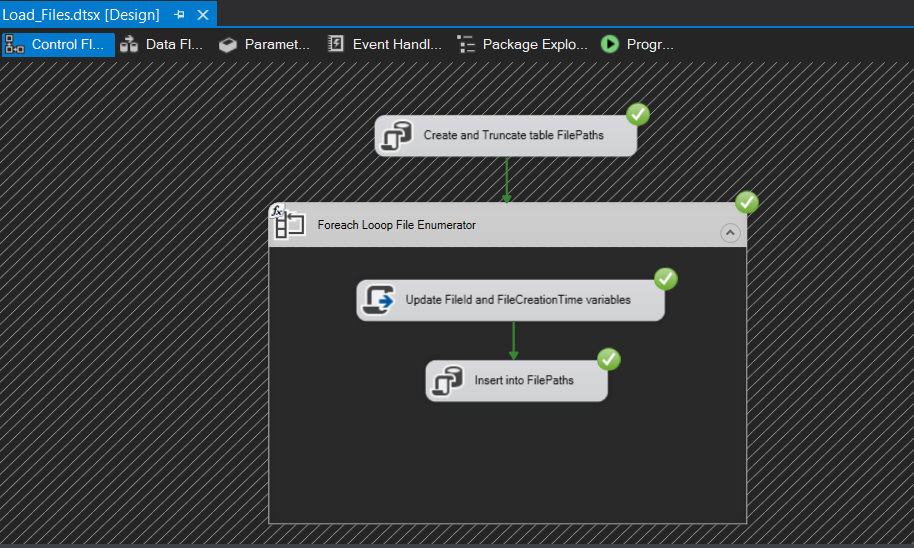
Also we should specify parameters for INSERT statement.





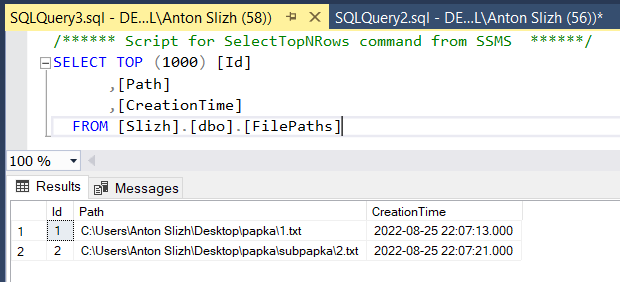


So, our ETL process is done. Let’s execute it.

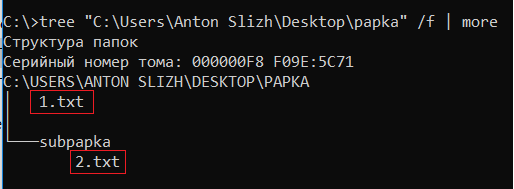


Now, let’s look at the result.

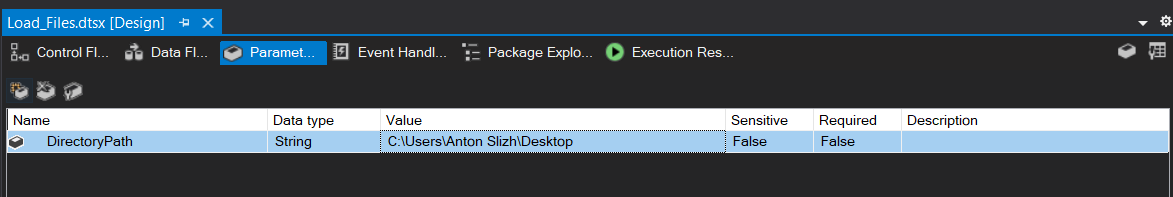
Select result data from the table *FilePaths* in the database:

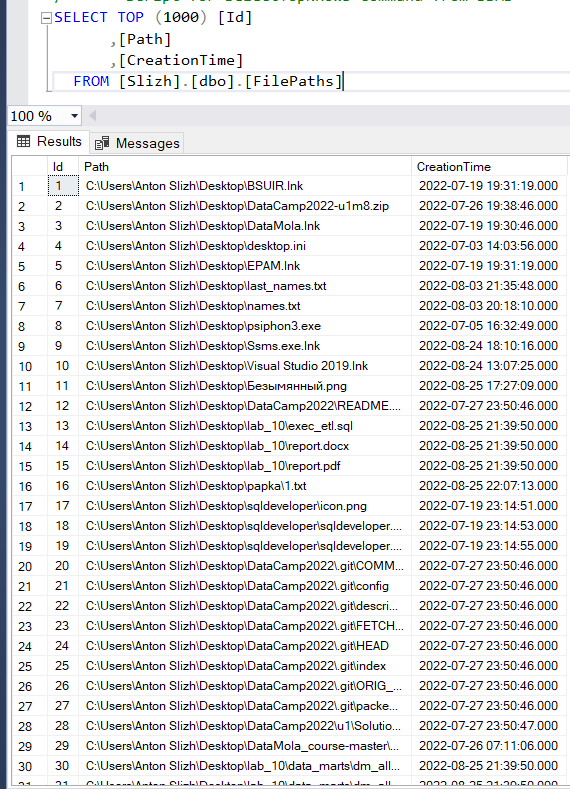


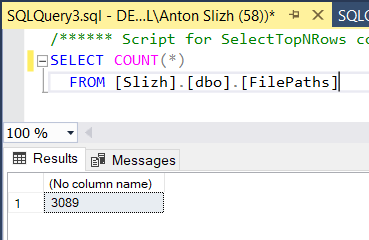
Make sure that specified directory (including subdirectories) contains only 2 files.



And let’s test the directory which contains more than 2 file. For example my Desktop directory.







All works fine.