SUP'COM
Ecole Supérieure des Communications de Tunis

P2M Project Report

Smart ETL Tool

Realized by:

Amani Haouachi

Houssem Yousfi

Supervised by:

Mr. Maher Heni

Academic Year: 2021 - 2022

Content:

List of figures:

I. General introduction

1- Context:

It is common knowledge that we are nowadays evolving very fast in all fields. Consequently the amount of data in the world increases at the same pace reaching 64.2 zettabytes in 2020 and it's predicted to go up to 180 zettabytes in 2025.

Ever since data started growing and changing forms people started looking for ways to store and organize it. The ultimate solution was ETL tools which refer to Extract Transform Load. They became a popular concept in the 1970s and were often used in data warehousing.

In fact ETL allows businesses to consolidate data from multiple databases and other sources into a single repository with data that has been properly formatted and qualified in preparation for analysis. This unified data repository allows for simplified access for analysis and additional processing.

2- Objective:

With this project we aim to conceive a smart ETL tool that after accessing the wanted data will do the necessary transformation and load it into a data warehouse. This tool will be the backend of a user friendly web platform that we will design from scratch.

This way a user will make sure their data is clean and organized therefore ready for analysis.

3- State of art:

There are numerous places where one can store data going from a CSV file containing columns to a full database with numerous tables.

One of the most used database management systems is MySQL. It is an open source system. As with other relational databases, MySQL stores data in tables made up of rows and columns. Users can define, manipulate, control, and query data using Structured Query Language, more commonly known as SQL.

There are plenty of other database management systems such as Firebird, Microsoft SQL Server, and IBM Informix etc... And also plenty of already developed ETL tools such as

HEVO, Airbyte etc... But most of them are not free and are made to be mainly used by big companies. In other words not everyone can have access to the services offered by these ETL tools unless they verify certain requirements.

II. Technologies used:

1- **Python:** was mainly used for backend in our project.it was the adequate programming language for us to use since it is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly procedural), object-oriented and functional programming.

2- **Flask:** this is basically a micro web framework written in Python. It is classified as a micro framework because it does not require particular tools or libraries. It also supports extensions that can add application features as if they were implemented in Flask itself.

3- **Bootstrap:** to help with the front End development bootstrap provided us with multiple HTML CSS user interface elements.

4- **MongoDB:** a document-oriented NoSQL database used for high volume data storage. Instead of using tables and rows as in the traditional relational databases, MongoDB makes use of collections and documents. We used this system to store users 'data like their passwords.

5- **Visual studio code:** a lightweight but powerful source code editor that we used to put everything together and link all the files and put everything together. With the right extensions downloaded Vscode seemed to the best candidate for this project.

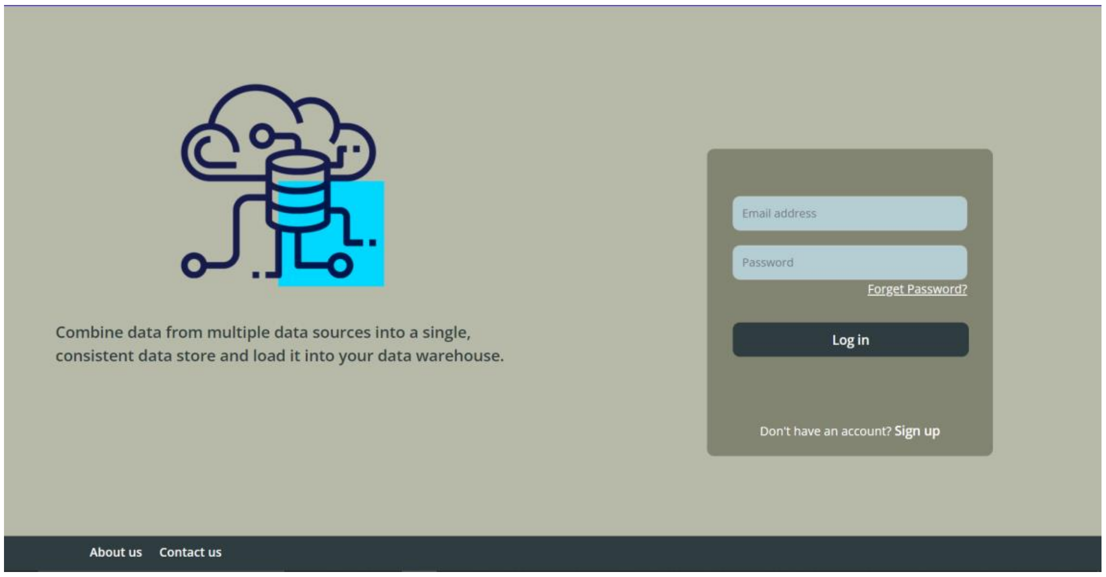
III. Work done:

1- Front end:

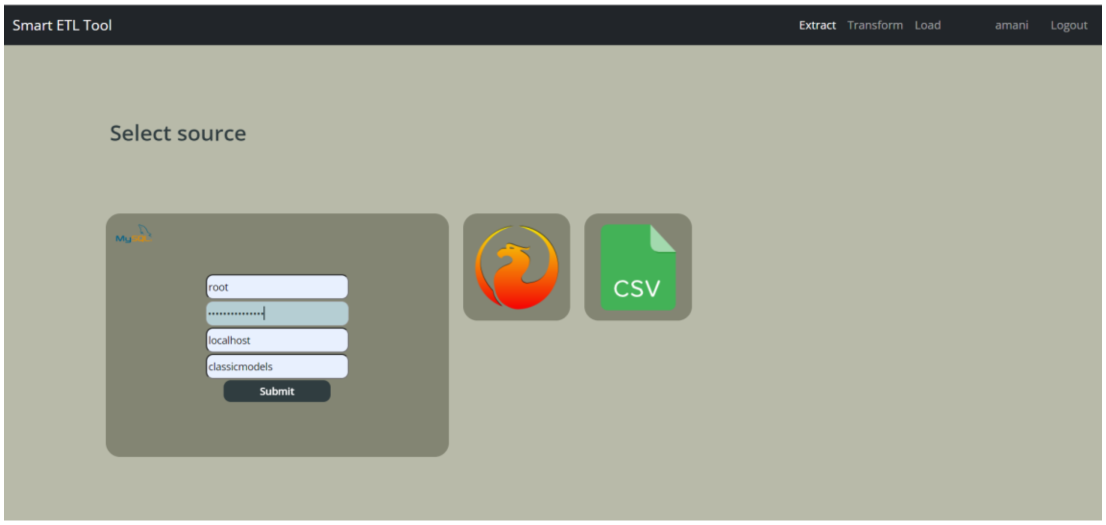
Our website’s interfaces were mainly conceived using HTML and CSS which are the basics of web development and we also used Bootstrap which is a potent front-end framework used to create modern websites and web apps. It's open-source and free to use, yet features numerous HTML and CSS templates for UI interface elements such as buttons and forms. Bootstrap also supports JavaScript extensions.

Front end development is very important and crucial in conveying this branding to customers. It needs to feature the same or similar colors, language, and graphics to help customers easily associate a business's website with their products elsewhere. That’s actually what we tried to accomplish with our design.

Here are a few samples:


Combine data from multiple data sources into a single, consistent data store and load it into your data warehouse.

-Figure1: Sign in interface -



root
localhost
classicmodels

-*Figure 2: Source selection interface* -

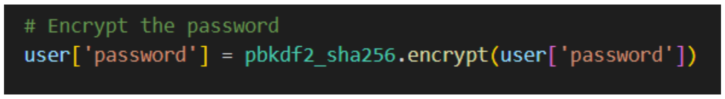
2- Back end:

Back end Development refers to the server side of development where you are primarily focused on how the site works and everything that communicates between the database and the browser.

The back end side of our web site is divided into two big sections:

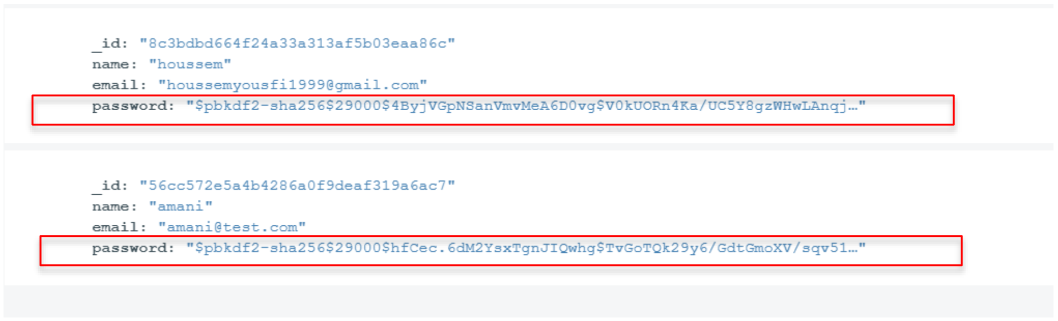
a- Back end of a regular website:

While working with data it's always better to make everything secured. With our project data is everything and sometimes it can be very personal that's why we made sure every user has an account that they create when first interacting with our website then the credentials are stored in our mongoDB database and the password is encrypted using a specific code



```
# Encrypt the password
user['password'] = pbkdf2_sha256.encrypt(user['password'])
```

-*Figure 3 Encryption code*-

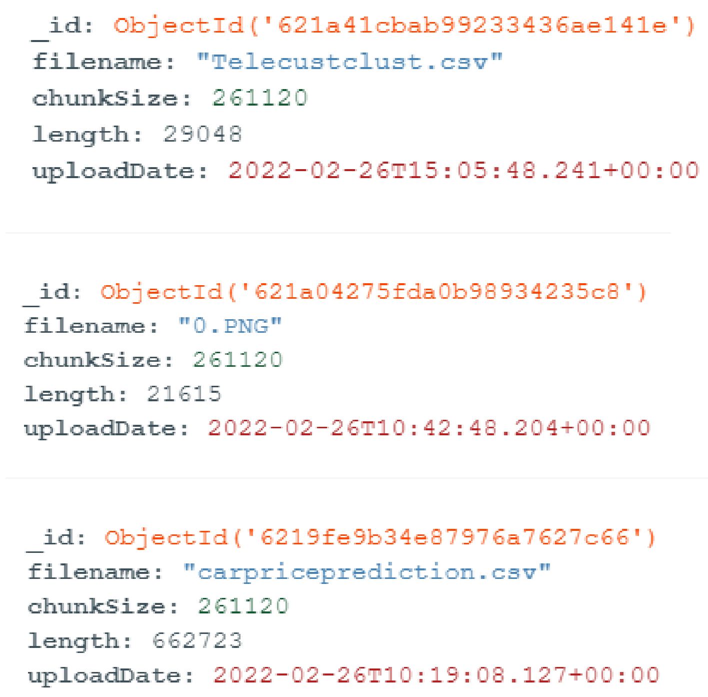


```
_id: "8c3bdbd664f24a33a313af5b03eaa86c"
name: "houssem"
email: "houssemyousfi1999@gmail.com"
password: "$pbkdf2-sha256$29000$4ByjVGpNSanVmvMeA6D0vg$V0kUORn4Ka/UC5Y8gzWHwLAnqj…"

_id: "56cc572e5a4b4286a0f9deaf319a6ac7"
name: "amani"
email: "amani@test.com"
password: "$pbkdf2-sha256$29000$hfCec.6dM2YsxTgnJIQwhg$TvGoTQk29y6/GdtGmoXV/sqv51…"
```

-*Figure 4 Encrypted passwords*-

Since we have CSV as a data source we need to store the uploaded files in our data base since that is basically the extract phase. Here are a few of them:



```
_id: ObjectId('621a41cbab99233436ae141e')
filename: "Telecustclust.csv"
chunkSize: 261120
length: 29048
uploadDate: 2022-02-26T15:05:48.241+00:00

_id: ObjectId('621a04275fda0b98934235c8')
filename: "0.PNG"
chunkSize: 261120
length: 21615
uploadDate: 2022-02-26T10:42:48.204+00:00

_id: ObjectId('6219fe9b34e87976a7627c66')
filename: "carpriceprediction.csv"
chunkSize: 261120
length: 662723
uploadDate: 2022-02-26T10:19:08.127+00:00
```

-Figure 5 uploaded csv files-

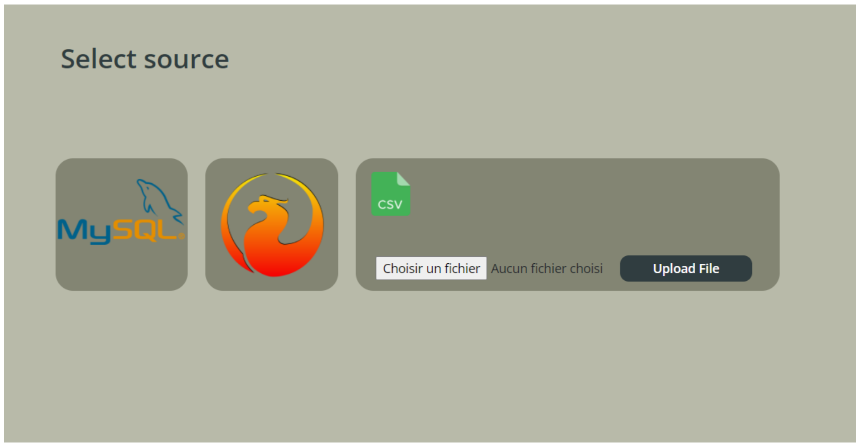
These pieces make our website ready to be used by both users who will only see the user interfaces and the admin who will consult the database for the management.

b- ETL process:

While having a website that looks good and works fine is a success it's still not the base of our project.

We were able to extract data from a csv file uploaded by the user and from a MySQL database that the user owns.

i. Csv file as a source:



-Figure 6:CSV as a source-

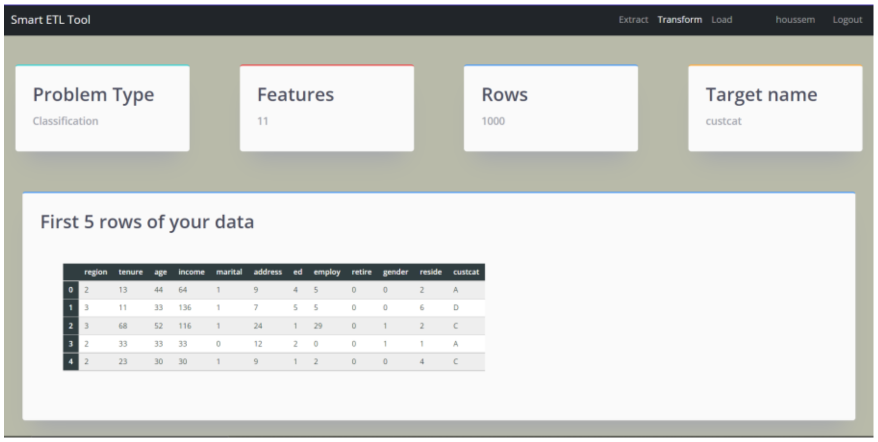
With a click on the “Upload File” button the user will be able to choose any csv file from their computer that will be immediately saved in our mongoDB database. Then a descriptive interface will appear to give the user some information about their file.

The user then will have the possibility to make transformations on the file by pressing a “Filter you data” button and the transformation will be done.

Our ETL like many others is made to be used for business intelligence so with each data we enter there is a target that we want to track. For example a market will enter the data of its sales and the target will be its profit.

With a csv file the target will be a column and after running the code written in python we made we’ll be able to know the problem type whether it be classification or regression then it will measure the “mutual information score” of each column which is the amount of information that column provides about the target.

Here’s an example



Problem Type
Classification

Features
11

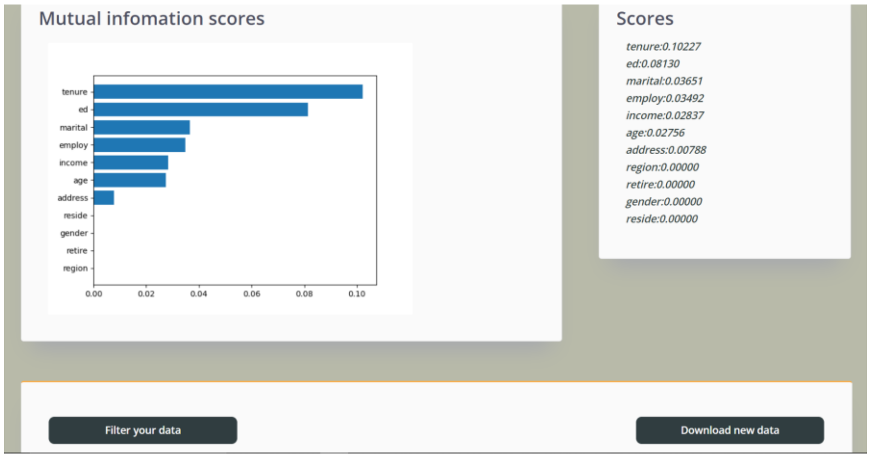
Rows
1000

Target name
custcat

First 5 rows of your data

	region	tenure	age	income	marital	address	ed	employ	retire	gender	reside	custcat
0	2	13	44	64	1	9	4	5	0	0	2	A
1	3	11	33	136	1	7	5	5	0	0	6	D
2	3	68	52	116	1	24	1	29	0	1	2	C
3	2	33	33	33	0	12	2	0	0	1	1	A
4	2	23	30	30	1	9	1	2	0	0	4	C

-Figure 7: Data description-



Scores

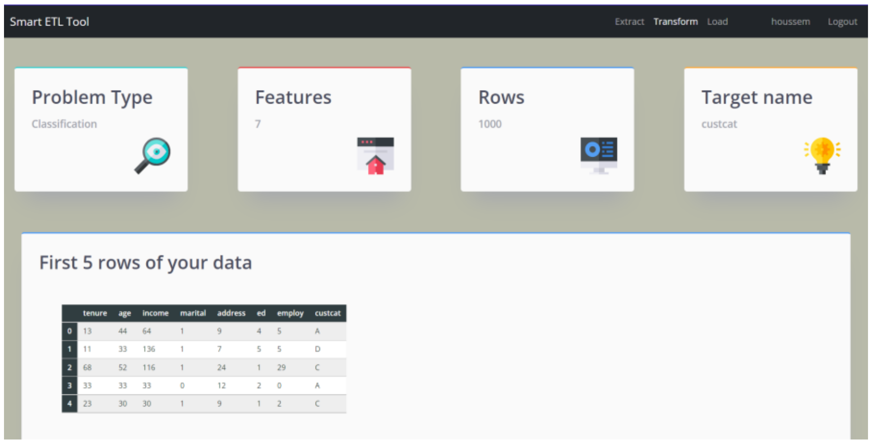
tenure:0.10227
ed:0.08130
marital:0.03651
employ:0.03492
income:0.02837
age:0.02756
address:0.00788
region:0.00000
retire:0.00000
gender:0.00000
reside:0.00000

-Figure 8: Mutual Information Score-

The figure above shows the mutual information score of multiple features on customers’ classification in a telecommunications company. We notice that some features are very important and give a good amount of information about the classification like the tenure since it has a great impact on one’s EMI (Equated Monthly Instalment) whereas region and gender give 0 information. It is then fair enough that we get rid of the columns that give no information if we want to have a good prediction of a customer's classification because this useless data can disturb the efficiency of a forecasting model for example.

After clicking on the transformation button a new file will be created after the necessary modifications and the user will be able to download it on their computer.

Here's a verification picture of the customer classification data:



Problem Type: Classification

Features: 7

Rows: 1000

Target name: custcat

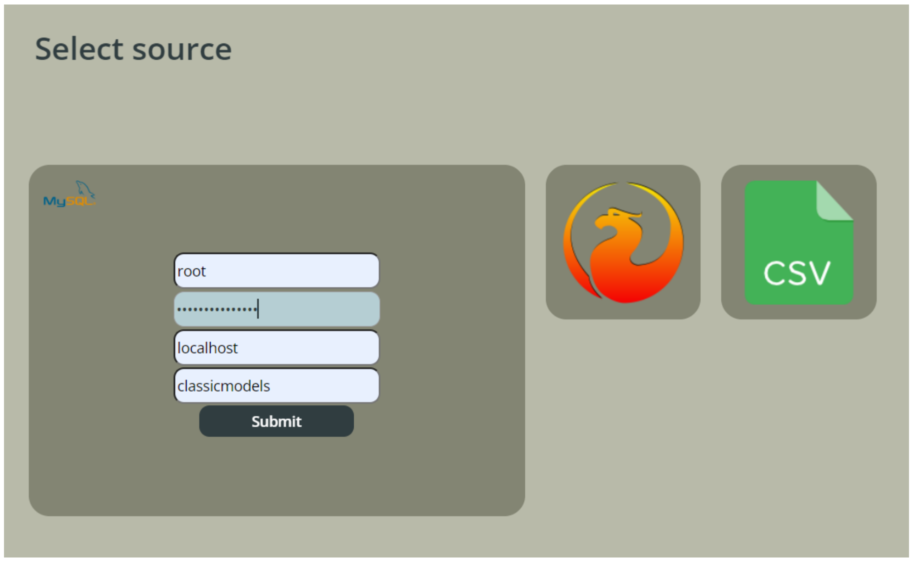
First 5 rows of your data

	tenure	age	income	marital	address	ed	employ	custcat
0	13	44	64	1	9	4	5	A
1	11	33	136	1	7	5	5	D
2	68	52	116	1	24	1	29	C
3	33	33	33	0	12	2	0	A
4	23	30	30	1	9	1	2	C

-Figure 9: Verification of the transformation-

After comparison with the before picture we notice that 4 columns are now gone and those are the features that gave no information about the classification.

ii. MySQL as a source:



root

•••••••••••••••

localhost

classicmodels

-Figure 10: MySQL as a source-

With the change in the source we had to change our approach because a python code is no longer that efficient.
A lot of searching made it clear that Airflow was our best candidate .it is as an open source tool to programmatically author, schedule and monitor workflows. Using this tool we have the ability to better develop and organize our ETL.
In fact there are two important terms that make Airflow very interesting: Task and Dag.
A Task is the basic unit of execution in Airflow. Tasks are arranged into DAGs, and then have upstream and downstream dependencies set between them into order to express the order they should run in.
The step of working with Airflow are the following:

- Defining each task separately (extract, transform, load) then linking them using y coming with airflow which will run these tasks in the hierarchical order we impose in the architecture of the DAG.
- Linking the DAG to our source and destination databases with airflow giving us the possibility to add connections and test them by providing the connection details like: Host, Login, password, Port...
- Controlling the process of the DAG from the beginning till its end by indicating whether each task is made successfully or not and detecting every possible error.

With the CSV file the user could easily retrieve the transformed file by downloading it however with a whole database that won't be possible as a result the loading step will take place in a PostgreSQL database.

PostgreSQL is a powerful, open source object-relational database system with over 30 years of active development that has earned it a strong reputation for reliability, feature robustness, and performance.

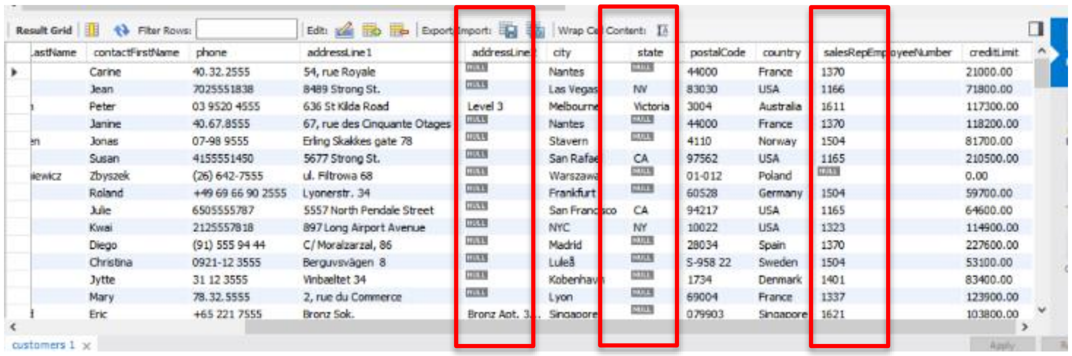
For everything to go smoothly, the airflow components are running on Docker that way we can implement the exterior python scripts in its environment.

- **Example:**

We worked with a database called 'classicmodels' that has multiple tables one of which being called 'Customers'. This table has many NULL recordings. So as a first transformation we wanted to fill these cases and get rid of some columns that were almost empty throughout the database then load the table in the PostgreSQL destination database.

It should be mentioned that no one can limit the transformations a user may want or need. In fact, even well-developed ETL tools like HEVO don’t have specific transformations that one can choose from. The transformation phase is actually a script that the user writes. In our case we wanted to try the procedure with the transformation mentioned earlier.

Here are the databases: the first is the mysql database with no transformation and the second is the postgreSQL database after the transformation.

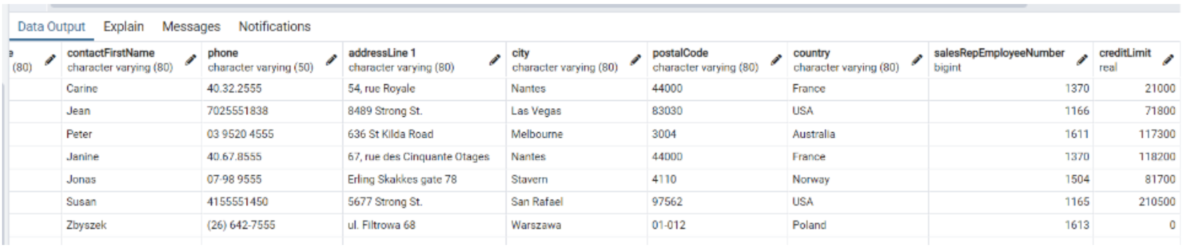


Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

astName	contactFirstName	phone	addressLine1	addressLine2	city	state	postalCode	country	salesRepEmployeeNumber	creditLimit
	Carine	40.32.2555	54, rue Royale	NULL	Nantes	NULL	44000	France	1370	21000.00
	Jean	7025551838	8489 Strong St.	NULL	Las Vegas	NV	83030	USA	1166	71800.00
	Peter	03 9520 4555	636 St Kilda Road	Level 3	Melbourne	Victoria	3004	Australia	1611	117300.00
	Janine	40.67.8555	67, rue des Cinquante Otages	NULL	Nantes	NULL	44000	France	1370	118200.00
en	Jonas	07-98 9555	Erling Skakkes gate 78	NULL	Stavern	NULL	4110	Norway	1504	81700.00
	Susan	4155551450	5677 Strong St.	NULL	San Rafael	CA	97562	USA	1165	210500.00
iewicz	Zbyszek	(26) 642-7555	ul. Filtrowa 68	NULL	Warszawa	NULL	01-012	Poland	NULL	0.00
	Roland	+49 69 66 90 2555	Lyonerstr. 34	NULL	Frankfurt	NULL	60528	Germany	1504	59700.00
	Julie	6505555787	5557 North Pendale Street	NULL	San Francisco	CA	94217	USA	1165	64600.00
	Kwai	2125557818	897 Long Airport Avenue	NULL	NYC	NY	10022	USA	1323	114900.00
	Diego	(91) 555 94 44	C/ Moralzarzal, 86	NULL	Madrid	NULL	28034	Spain	1370	227600.00
	Christina	0921-12 3555	Berguvsvägen 8	NULL	Luleå	NULL	S-958 22	Sweden	1504	53100.00
	Jytte	31 12 3555	Vinbæltet 34	NULL	Kobenhavn	NULL	1734	Denmark	1401	83400.00
	Mary	78.32.5555	2, rue du Commerce	NULL	Lyon	NULL	69004	France	1337	123900.00
	Eric	+65 221 7555	Bronz Sok.	Bronz Apt. 3...	Singapore	NULL	079903	Singapore	1621	103800.00

customers 1

-Figure 11: The before data base in MySQL-



Data Output | Explain | Messages | Notifications

(80)	contactFirstName character varying (80)	phone character varying (50)	addressLine 1 character varying (80)	city character varying (80)	postalCode character varying (80)	country character varying (80)	salesRepEmployeeNumber bigint	creditLimit real
	Carine	40.32.2555	54, rue Royale	Nantes	44000	France	1370	21000
	Jean	7025551838	8489 Strong St.	Las Vegas	83030	USA	1166	71800
	Peter	03 9520 4555	636 St Kilda Road	Melbourne	3004	Australia	1611	117300
	Janine	40.67.8555	67, rue des Cinquante Otages	Nantes	44000	France	1370	118200
	Jonas	07-98 9555	Erling Skakkes gate 78	Stavern	4110	Norway	1504	81700
	Susan	4155551450	5677 Strong St.	San Rafael	97562	USA	1165	210500
	Zbyszek	(26) 642-7555	ul. Filtrowa 68	Warszawa	01-012	Poland	1613	0

-Figure 12: The transformed database loaded in PostgreSQL-

The ETL works successfully and we can change the queries to our liking. What is yet to be done is finishing up our user interfaces with Airflow as an API.

IV. Conclusion:

ETL tools are very important and are crucial in our modern world it was a subject that really caught our attention and we tried or best to make things work on the interfaces.in this project we were able to perform an ETL process with two sources : a csv file and a MySQL database.

It would have been easier if enough documentation was available but the whole concept is still considered new and we were lucky enough to receive the help of our tutor Mr Maher Heni who didn’t hesitate to give us the necessary assistance.

Our work is not completely done but that does not take anything away from the great experience we had making it and the fact that we had to look up things that were not in our academic program gave us strength in the coding field.

References:

Hevo: **https://hevodata.com/**

Airflow: **https://airflow.apache.org/**

Docker: **https://www.docker.com/**