Introduction

In this project we shall use python to extract data from CSV files and make inserts into a PostgreSQL database. we will be utilizing the convenient interface provided by the psycopg2 library that is used to conduct PostgreSQL database operations in Python.

This automation of the data-load process is similar to the way that timely CSV files are often processed in a data pipeline via some software programs and then further data analysis is done on these data inside a database.

Creation of a python script which is able to discover CSV files within a given direction, read before and do data cleansing was a core target. In the end the results of data will be inserted to the PostgreSQL database. This code will automate the data pilling process, making it more efficient and easier to scale as the volumes of data increase.

We intend to describe this pipeline for a demonstration of a workable method of the flow of CSV files to a PostgreSQL base, manifesting an undisputed effort of Python integration and PostgreSQL database for the effective data management process. The below are the screenshots showing of the process of flowing data usage from CSV file to PostgreSQL database.

Figure 1 Installation of necessary packages.

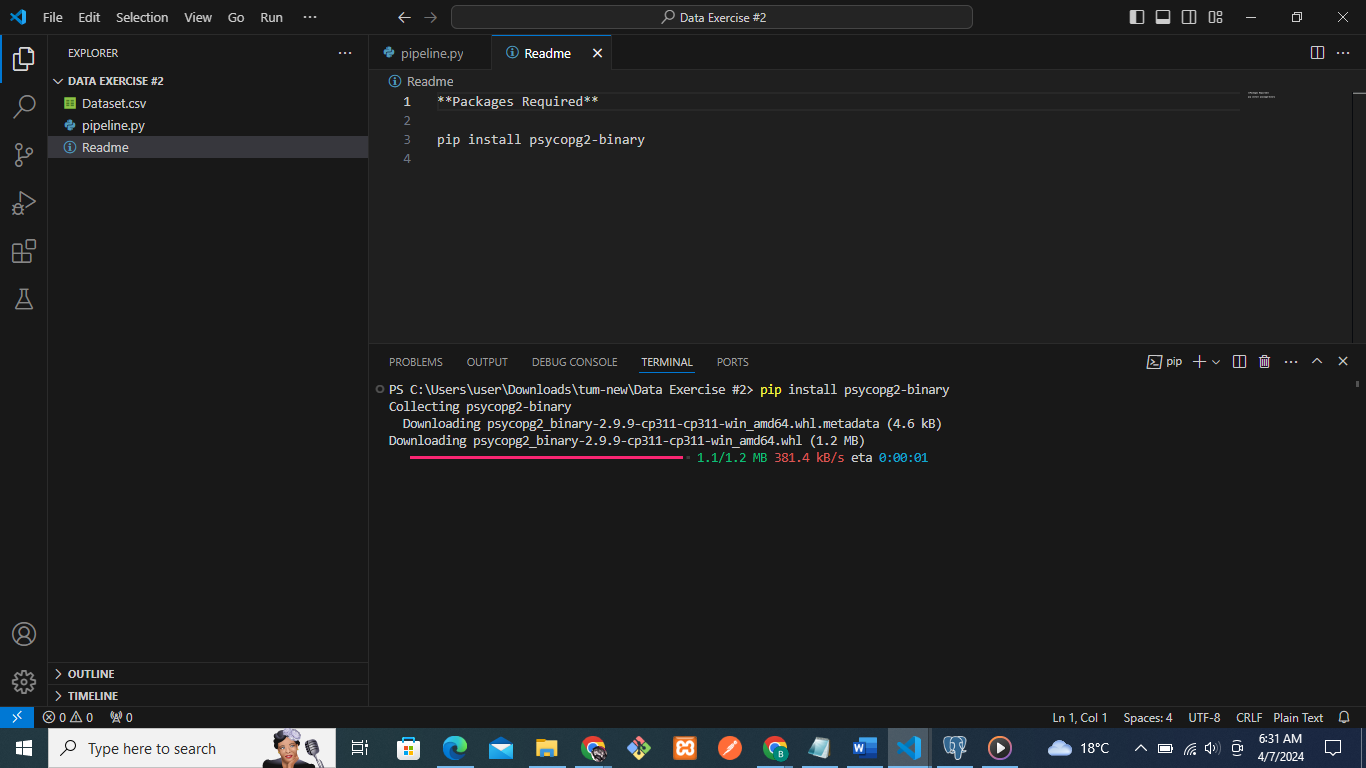


Figure 2 Logging in to the PgAdmin.

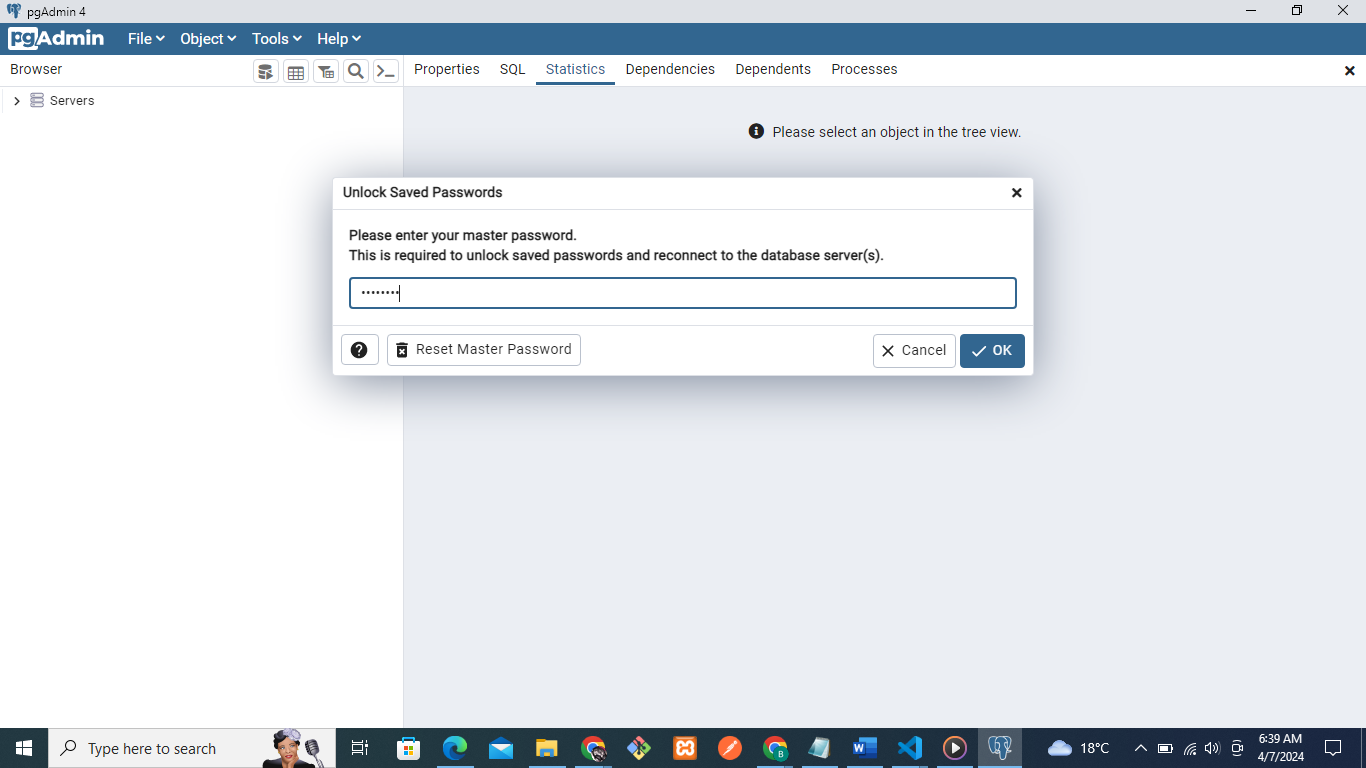


Figure 3 Logging in to database server.

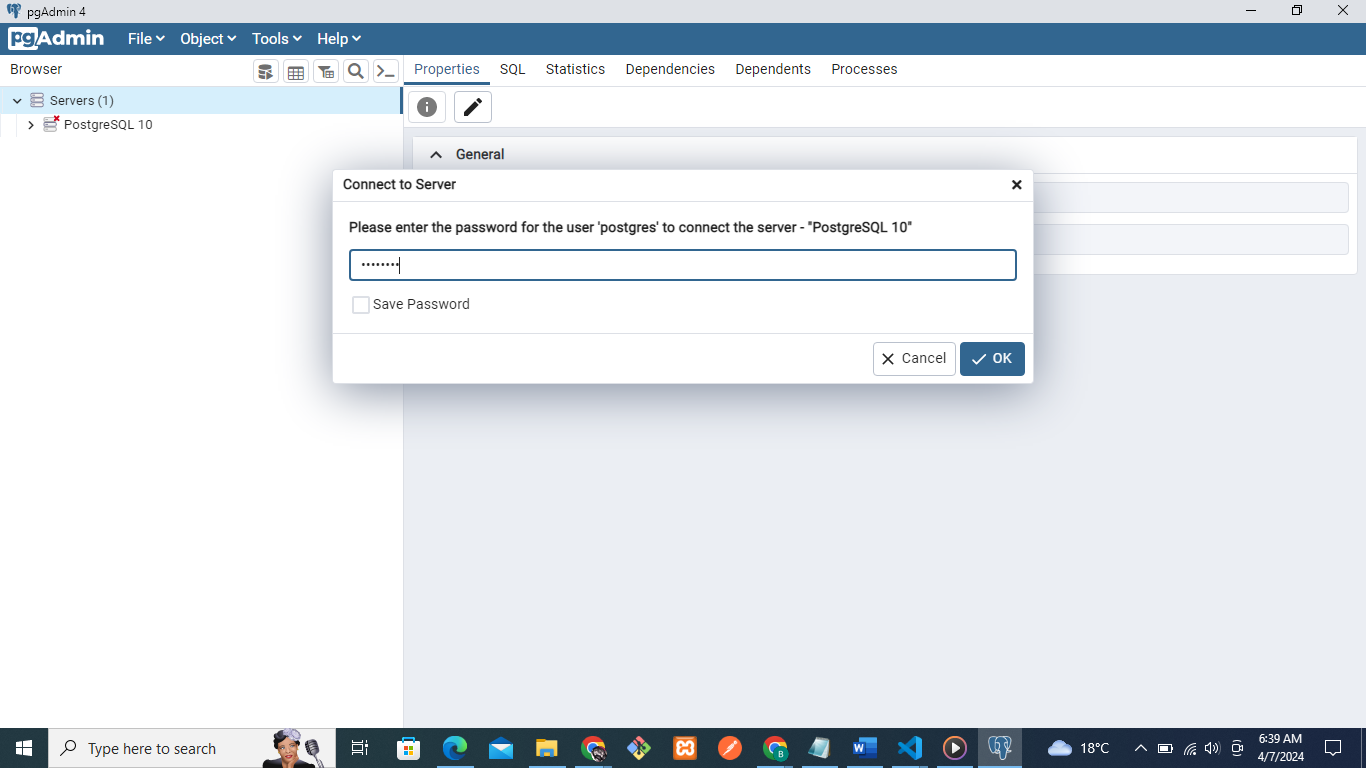


Figure 4 Postgres Database Management System.

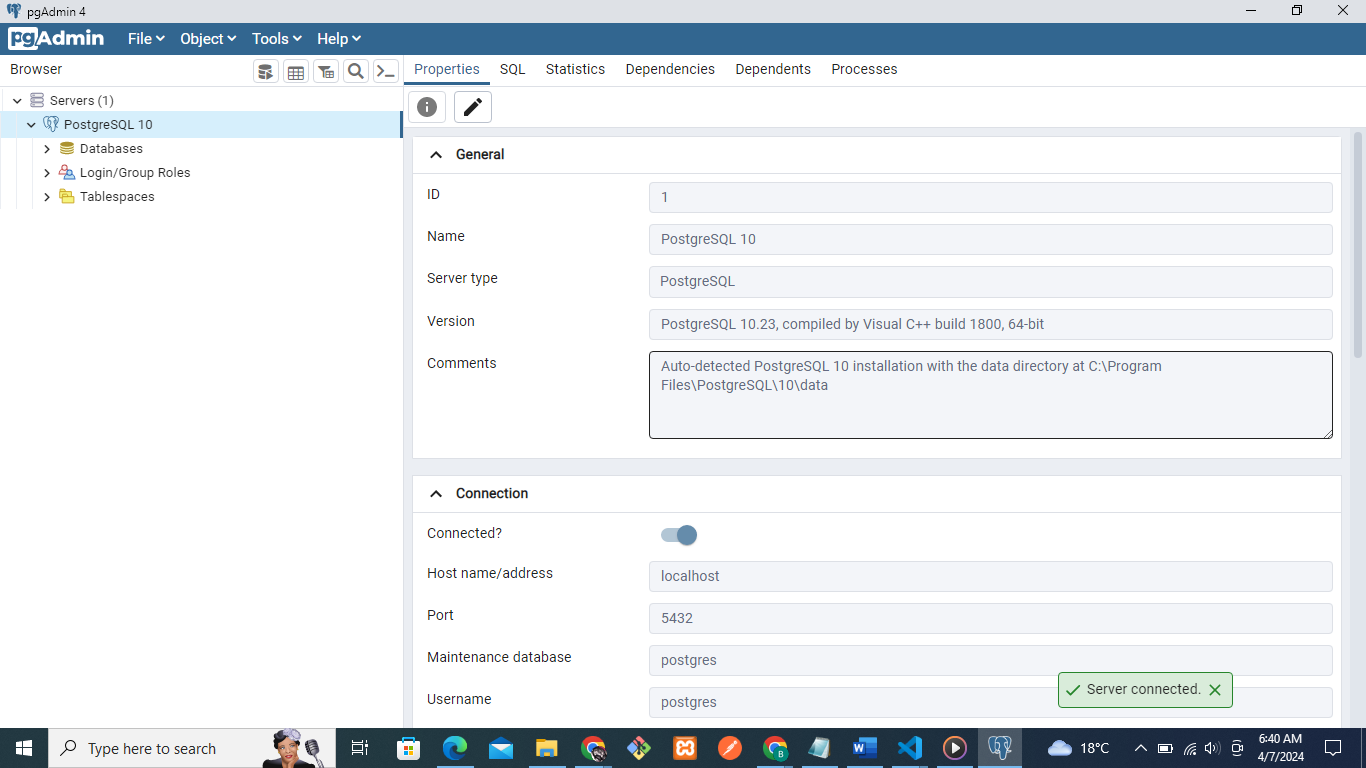


Figure 5 Server connected to the PgAdmin.

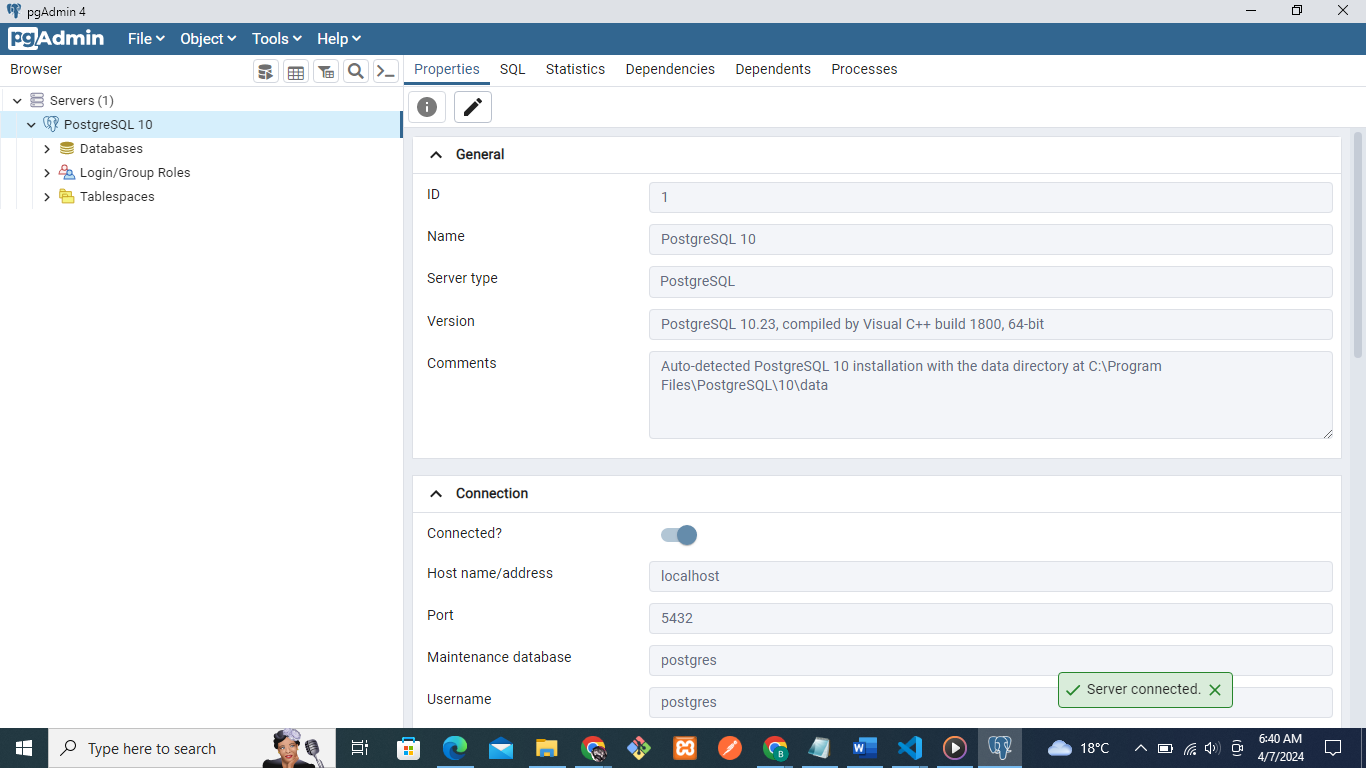


Figure 6 Creating database in the DBMS.

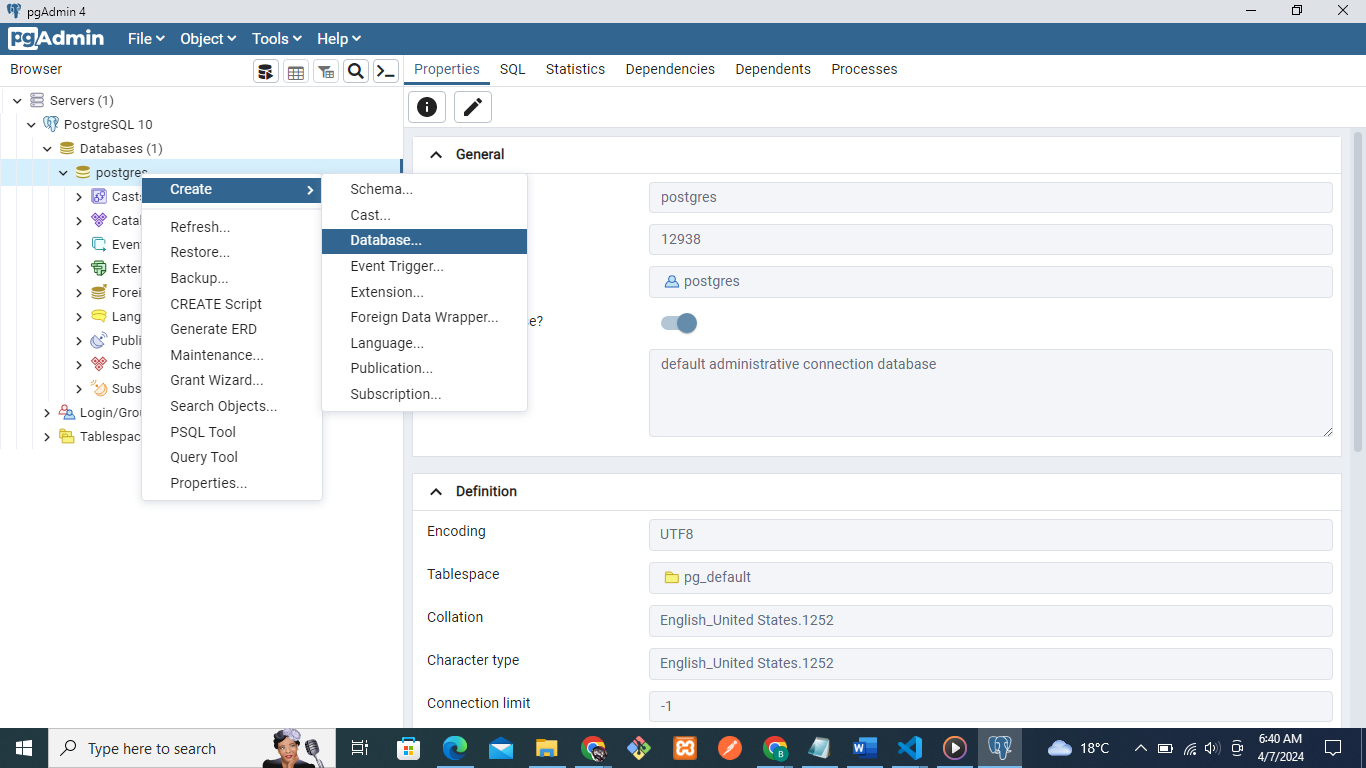


Figure 7 Give the database name - mypipelinedatabase.

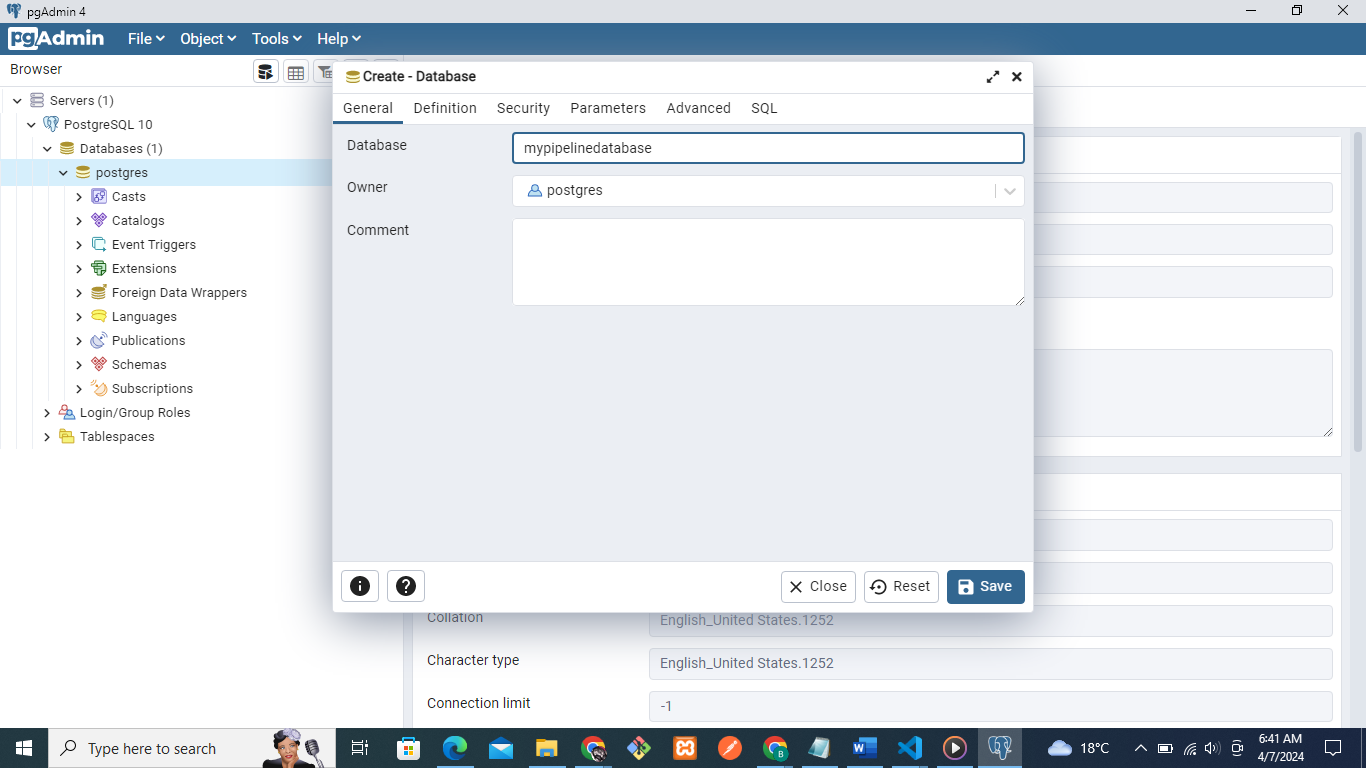


Figure 8 Preview of mypipelinedatabase.

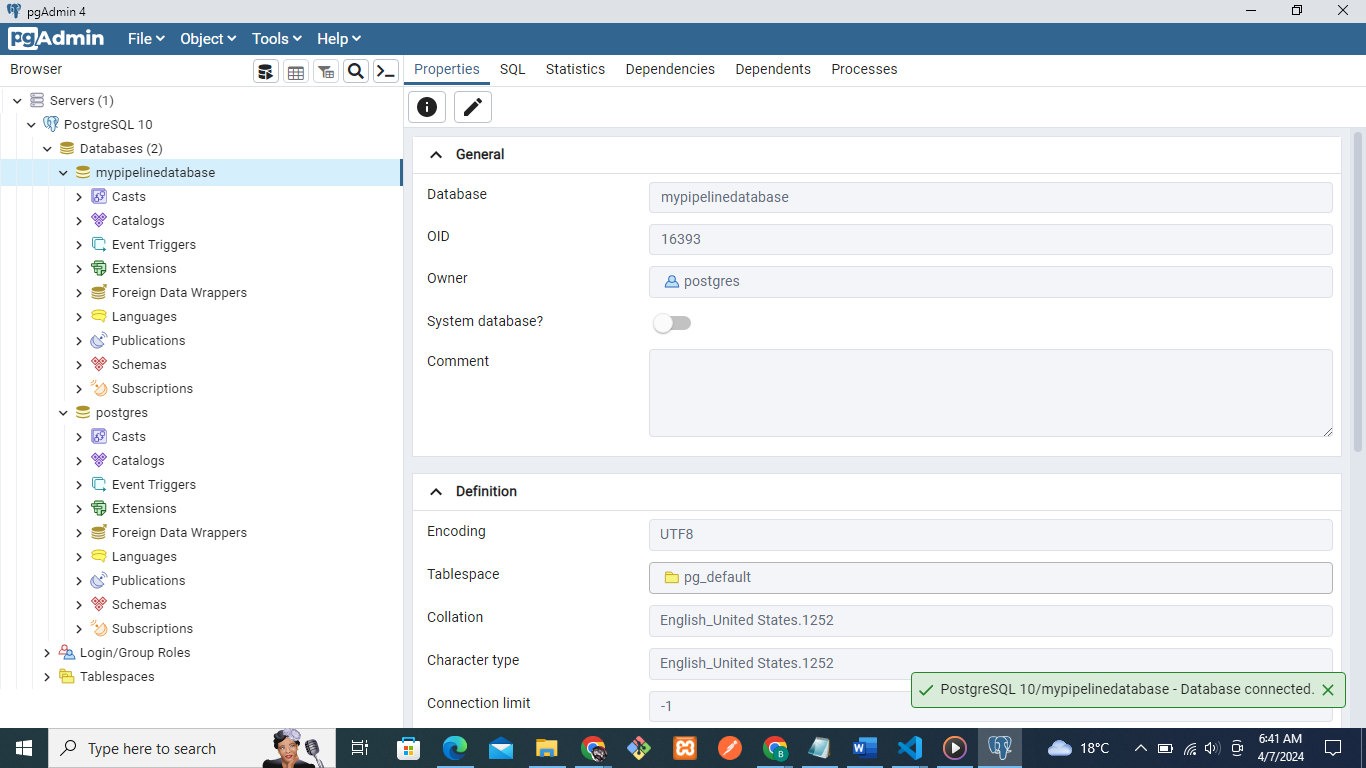


Figure 9 No tables before data ingestion.

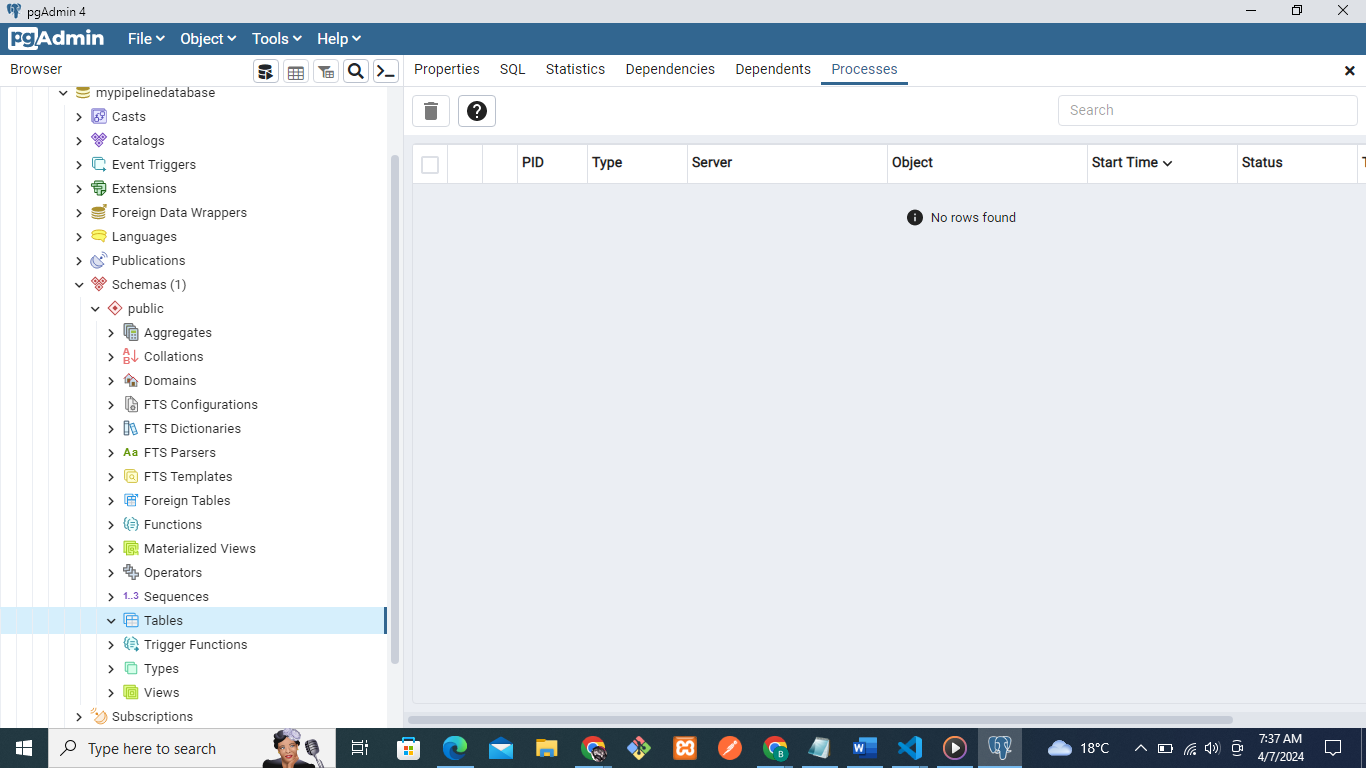


Figure 10 Data ingestion was successfully done.

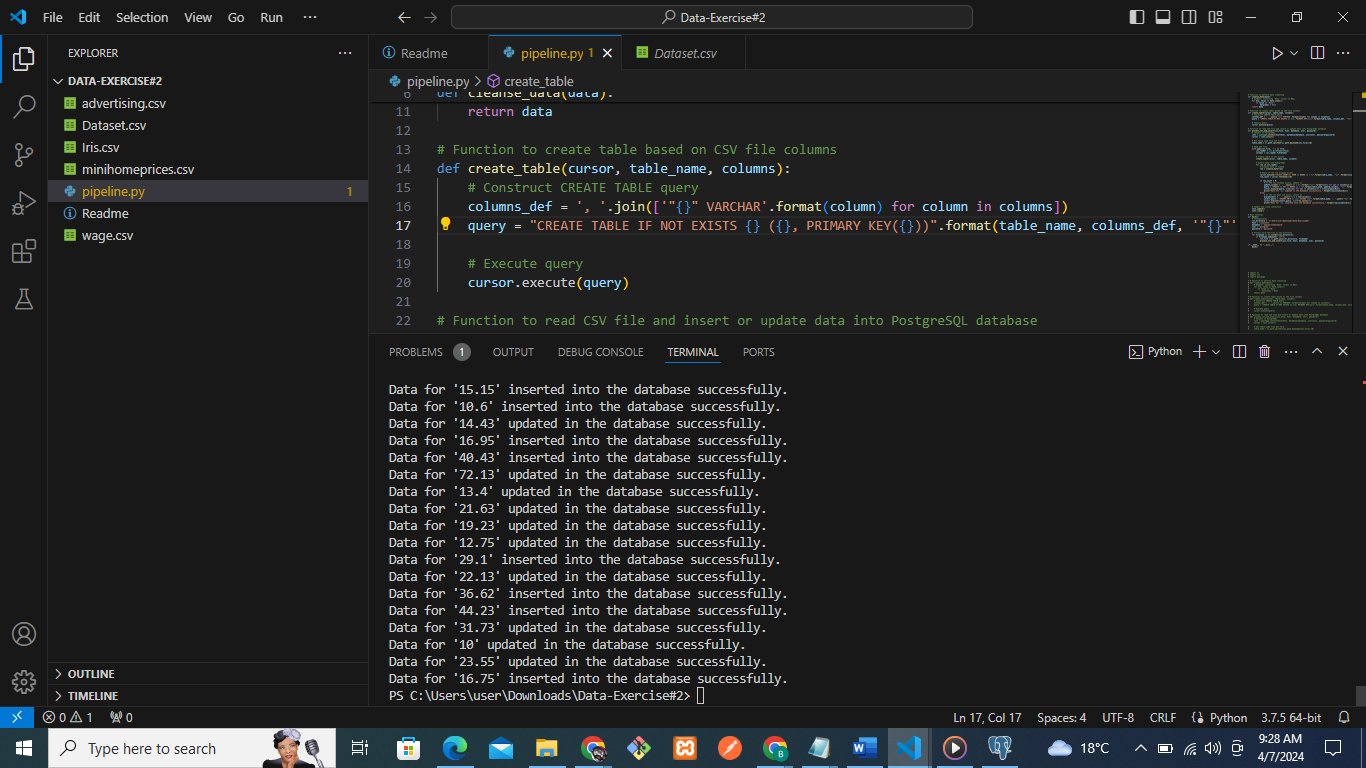


Figure 11 Wage dataset ingested in the database successfully.

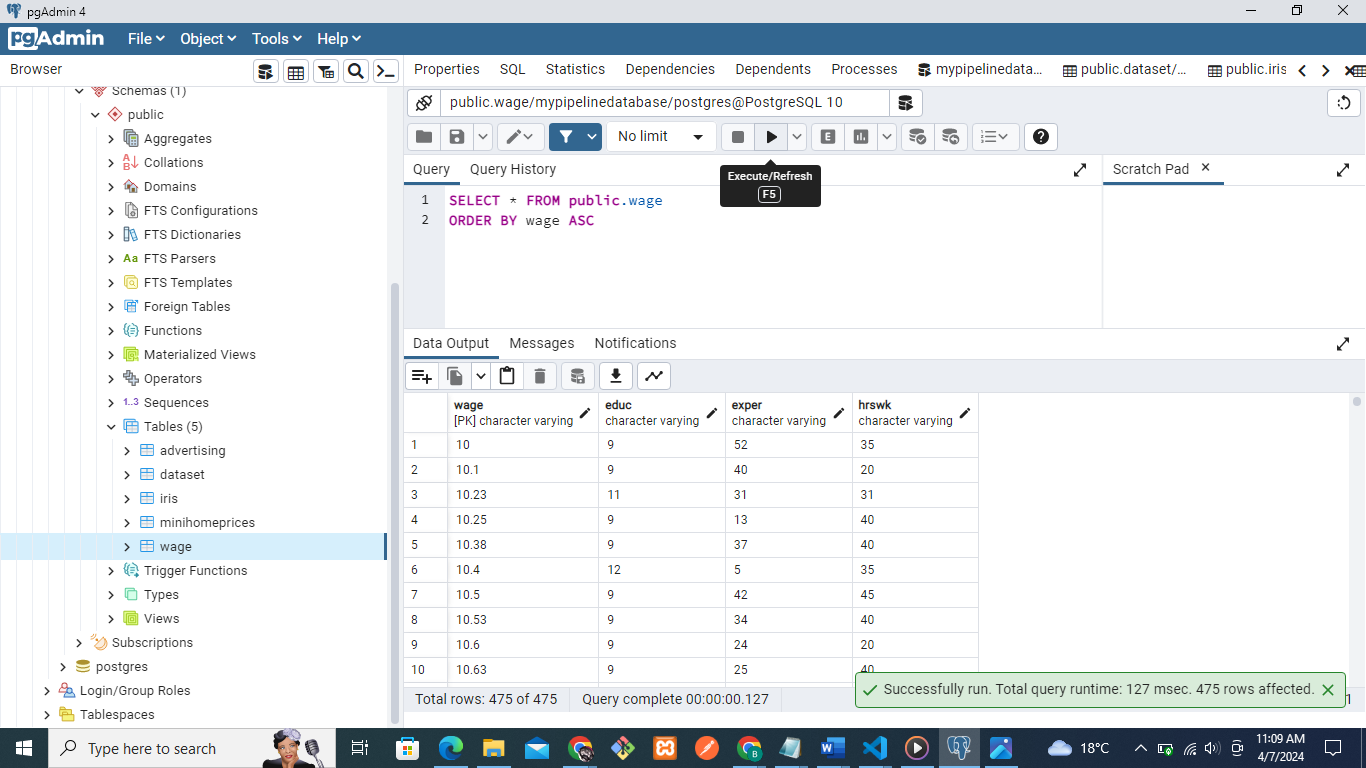


Figure 12 Minihomeprices dataset ingested in the database successfully.

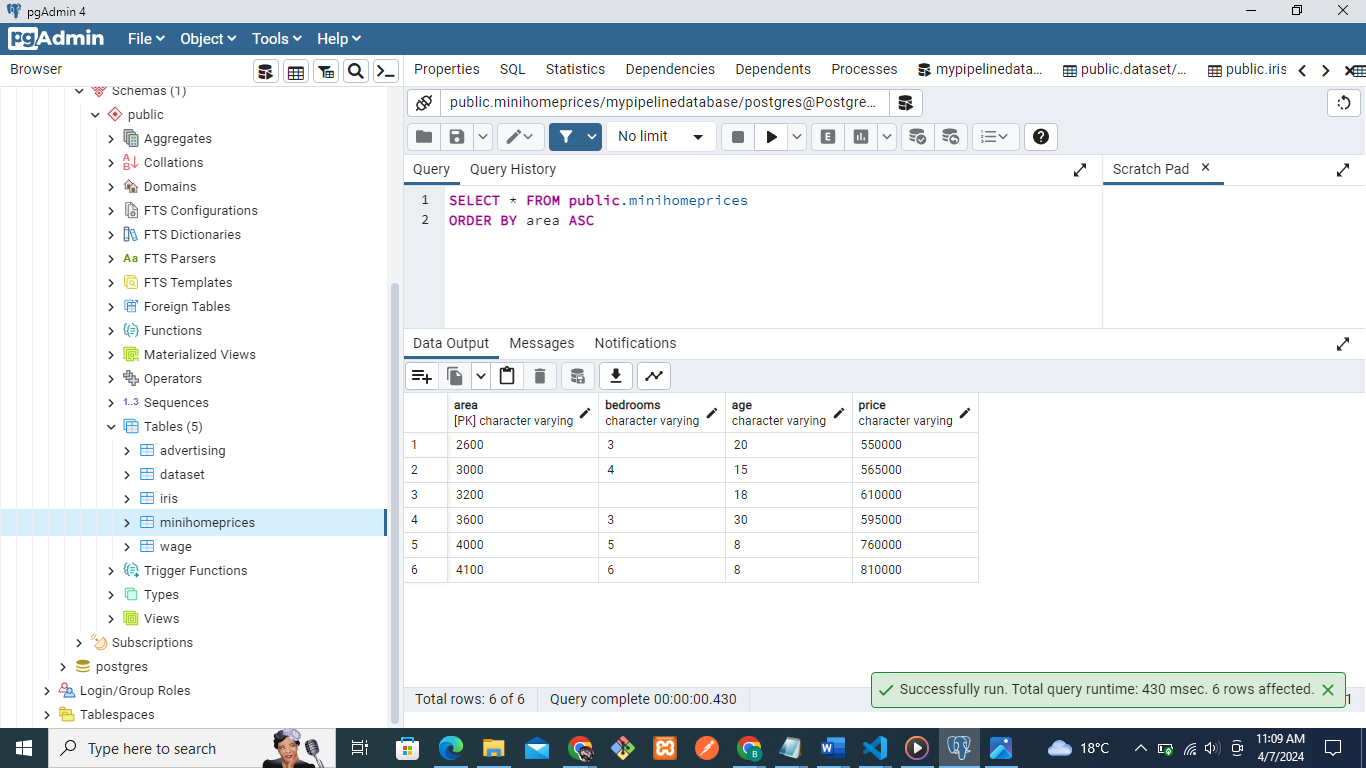


Figure 13 Iris dataset ingested in the database successfully.

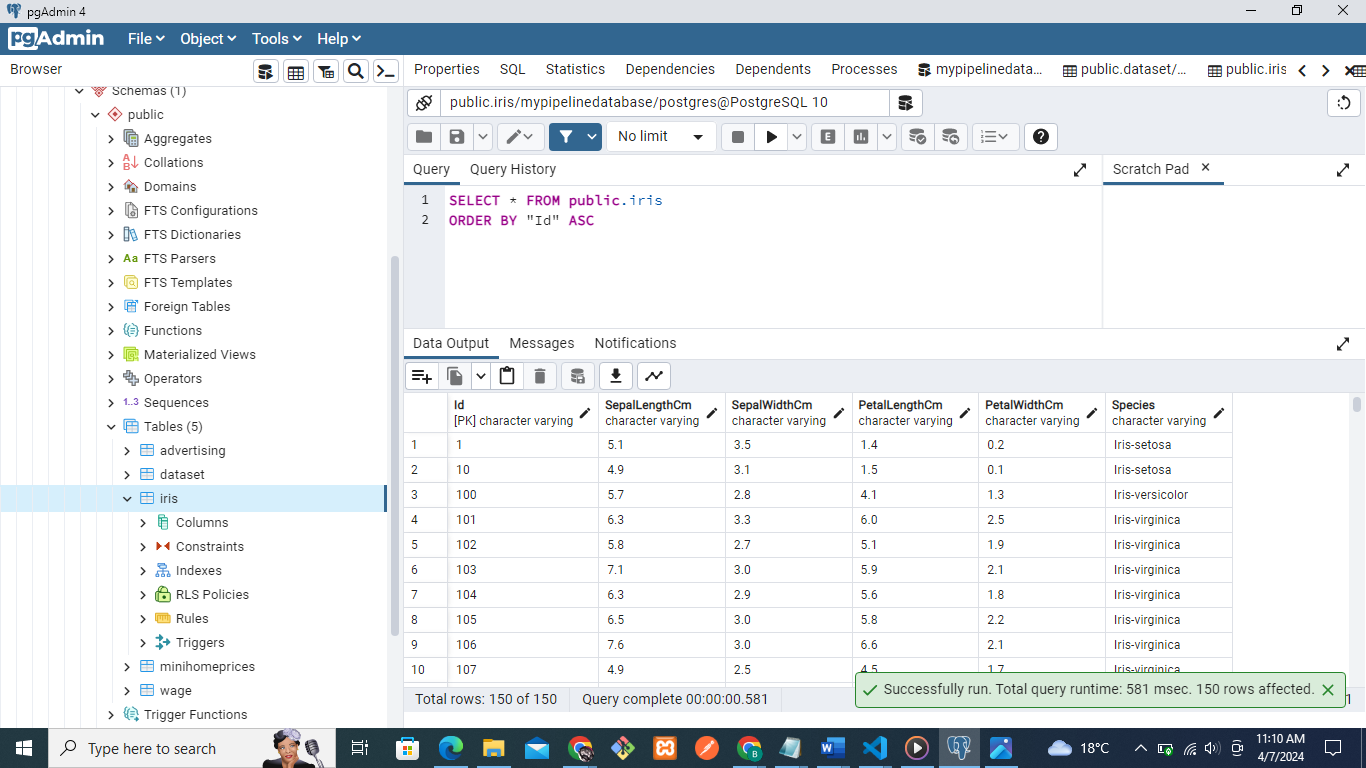


Figure 14 Waste water analysis dataset ingested in the database successfully.

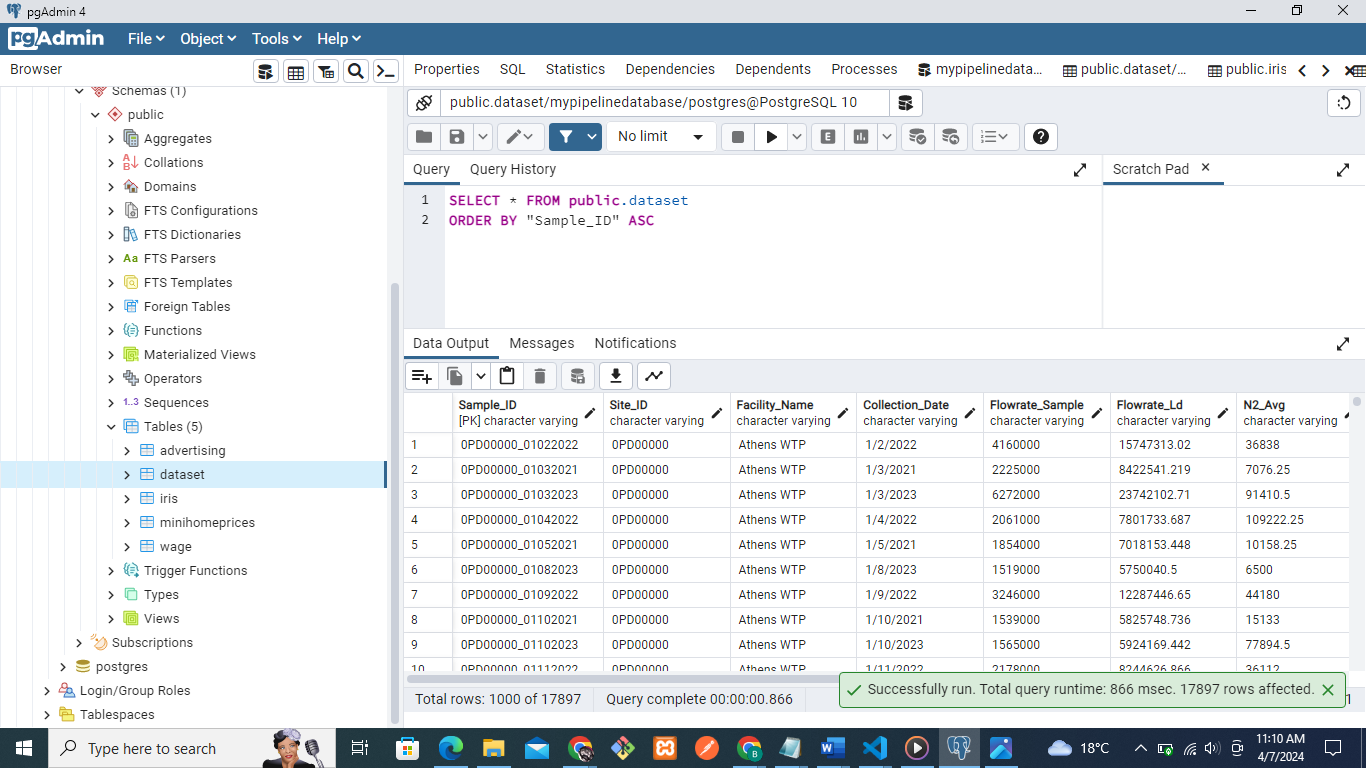
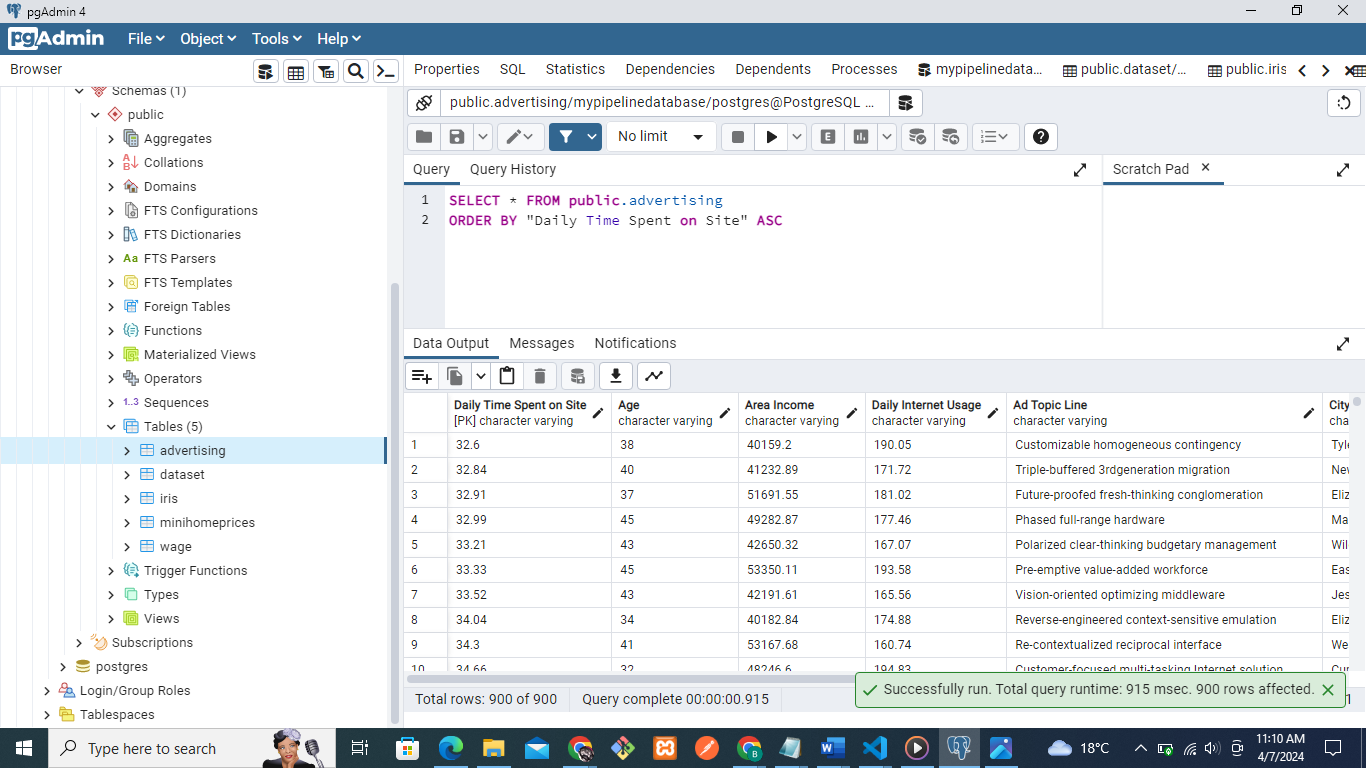


Figure 15 Advertisement dataset ingested in the database successfully.



*Automation Process*

By utilizing the Python script, the inbuilt functionality to obtain data from CSV files and to enter it into a PostgreSQL database can be executed automatically. Processing begins with the creation of appropriate functions to clean the data and create tables based on the columns listed in the comma separated value files. The `cleanse\_data() ` function could perform data clean-up operations, such as ‘None’ to NULL, which would help maintain data consistency. Also, the `create\_table()` function builds dynamically CREATE TABLE queries and guarantees the column sets of CSV and database schema are synchronous through their columns.

It is ` process\_csv\_and\_insert()` function that serves as a primary substantial processor, in charge with the reading of CSV files, creating of database table (if necessary), and inputting or updating of data into the PostgreSQL database. Inside this role, every single CSV file is treated as an isolated data set and is running through the iteration process. The script intervenes for each file and includes the table name that is based on the file name; it constructs appropriate SQL queries to either insert a new data or update existing data in the database.

Additionally, the script efficiently manages the pre-existing data within the database with great skill. In this place, it first verifies if data is in target table or not. The data is available, the script the data is representing existing rows, observing the primary key constraints, while avoiding the data duplication. On the contrary, if not data exists at all, the code will instead store it into the database.

To run the script, specify the directory with the CSV files, and the database connection properties containing the name of the host, database name, user, password, etc. On its way the script applies every CSV file hidden in the mentioned directory into the PostgreSQL DB as it is appropriate getting either new data into the DB or updating them, if necessary.

In summary, the script builds an exemplary roadmap that is both low-maintenance and efficient, during the fill of CSV data into a PostgreSQL database. One distinctive feature is that it allows the user to be flexible enough, while making sure that all inputs are consistent and ultimately can be also counted upon when dealing with large numbers of inputs.

*Conclusion*

With this task, we have successfully applied the extraction of data from the CSV files and its insertion into the database using `psycopg2` library in python. We created a script which checks for CSV files in a particular directory, reads these files, and performs the data load operation within PostgreSQL database.

Systematically logging the pipeline, we have shown a simple mean for getting data from CSV files to PostgreSQL database. The script's functioning ensures the fast processing of incoming data and thus the script becomes scalable enough for larger sets of information.

Our team is able to import four CSV datasets the project folder to the PostgreSQL database by using the automated data pipeline, demonstrating the way it is efficient. This feat reinforces the skill of the script in reducing data acquisition time, thus creating the platform for proper data analysis and decision-making.

Generally, we can see that such a project can comes to be an example of automation in data management and can prove that there can be a perfect connection between Python and PostgreSQL for data processing and analysis.