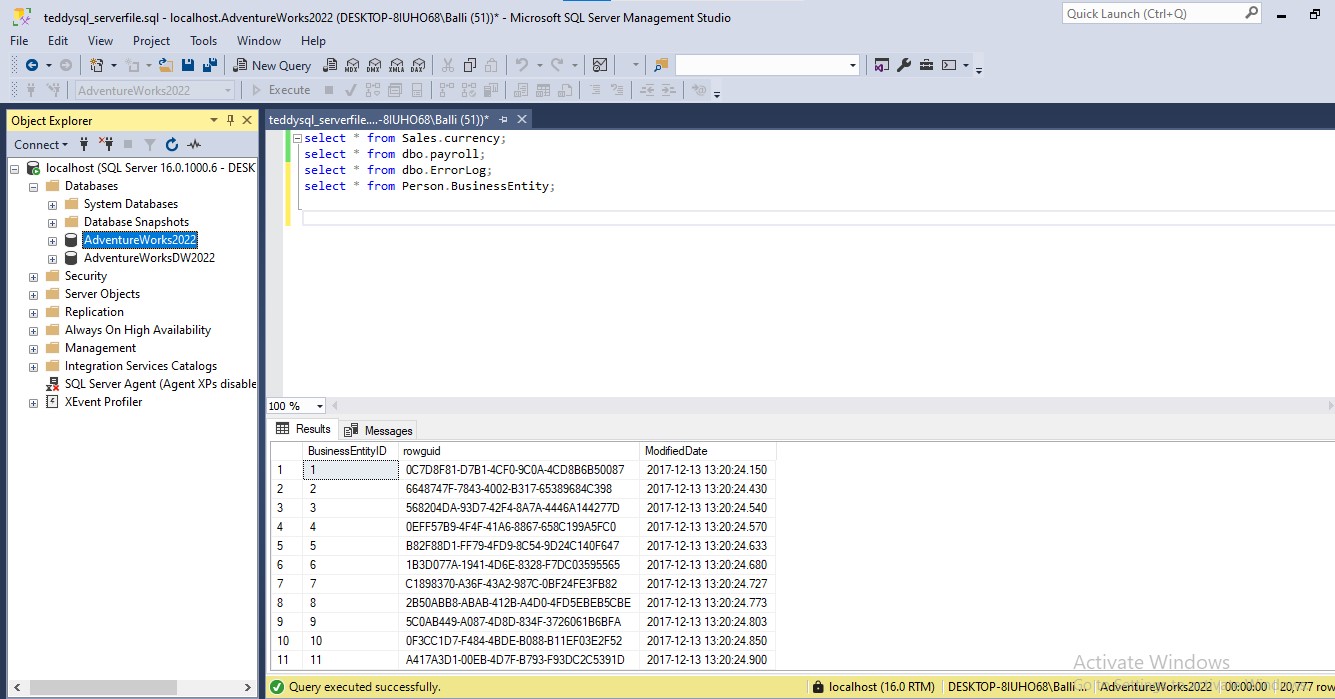
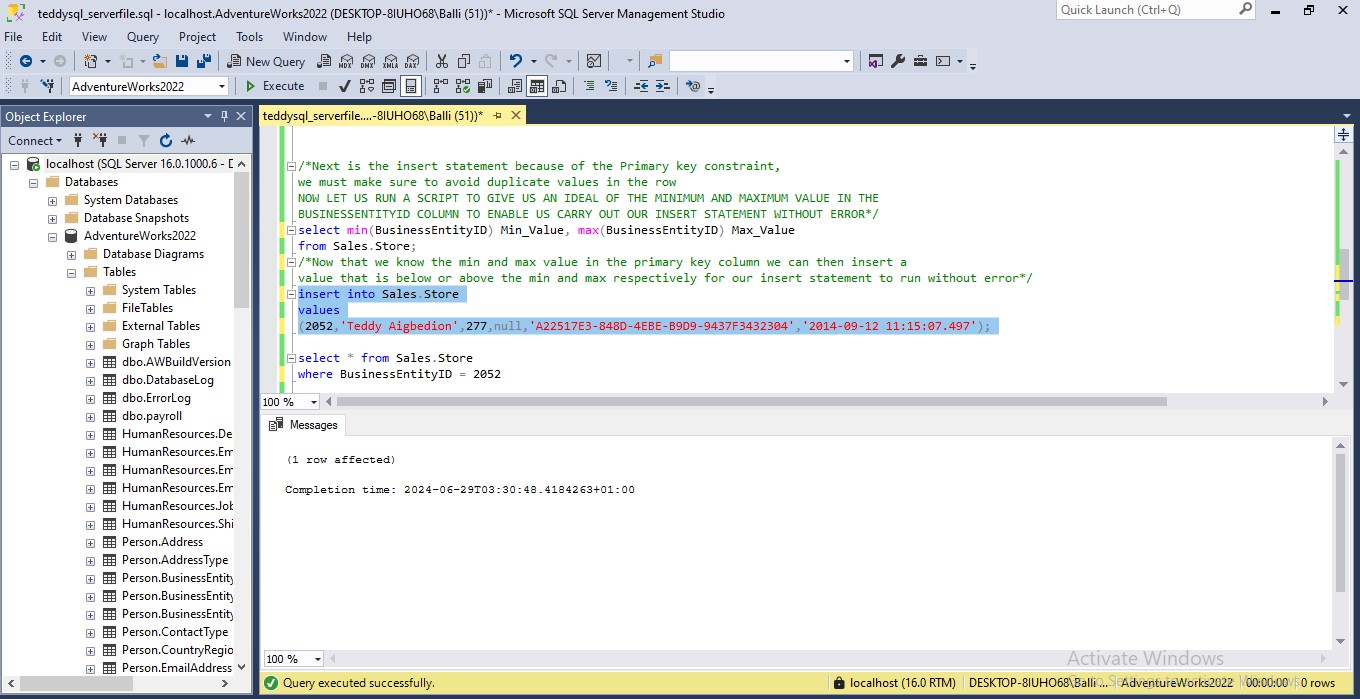
**Teddy Aigbedion**

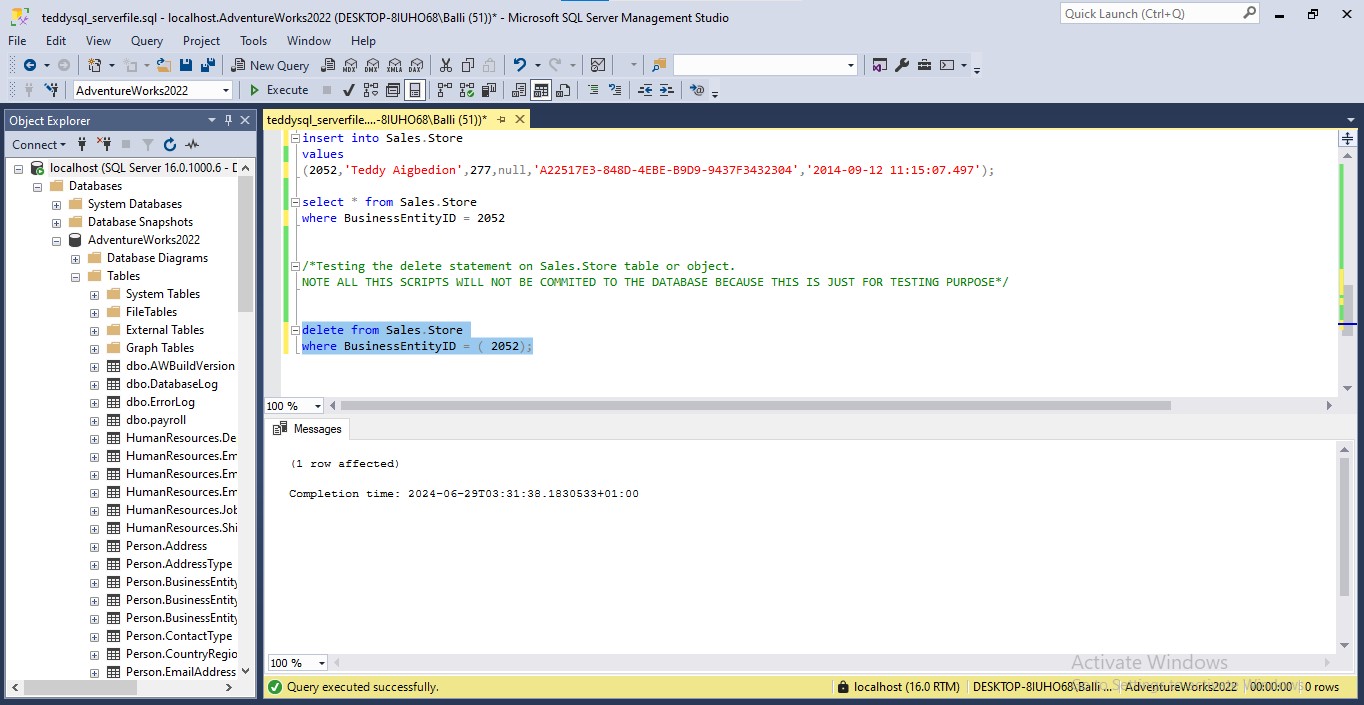
**Pivot-ED Assessment4**

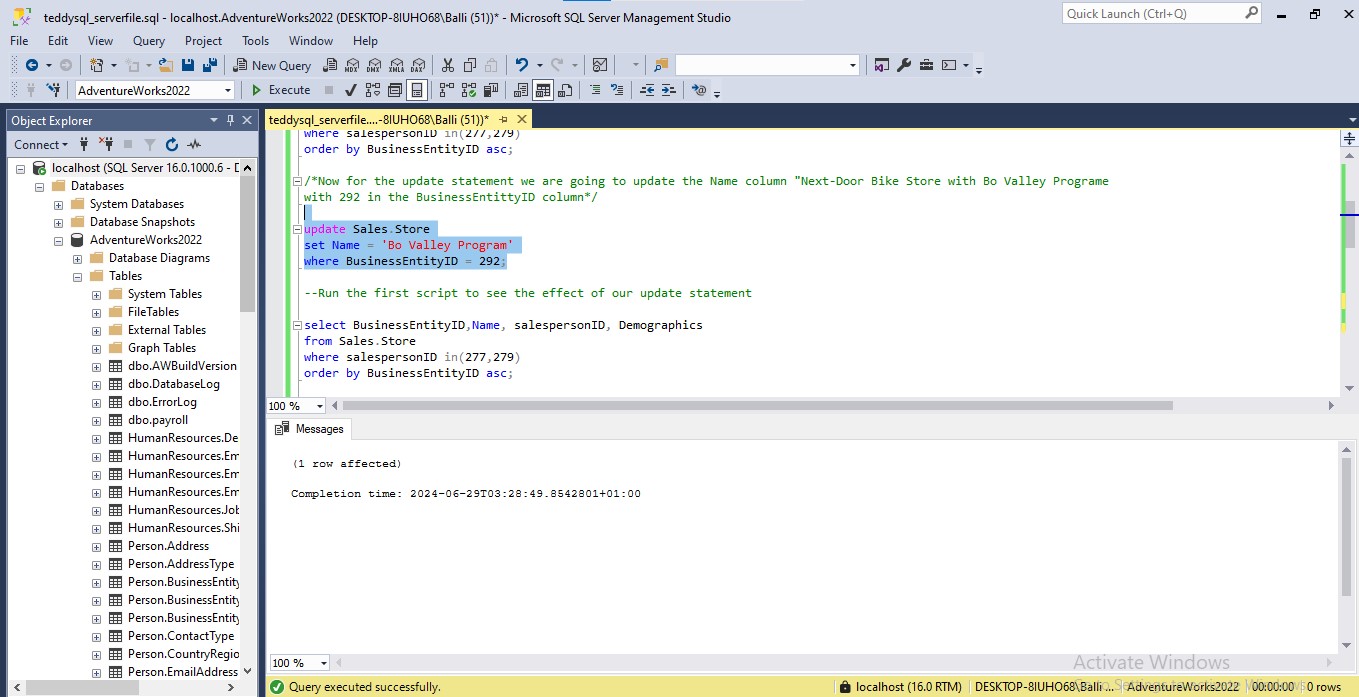
From the guide given on how to install Microsoft Sql Server in my machine, and also add two databases(AdventureWorks2022 and AdventureWorksDW2022)

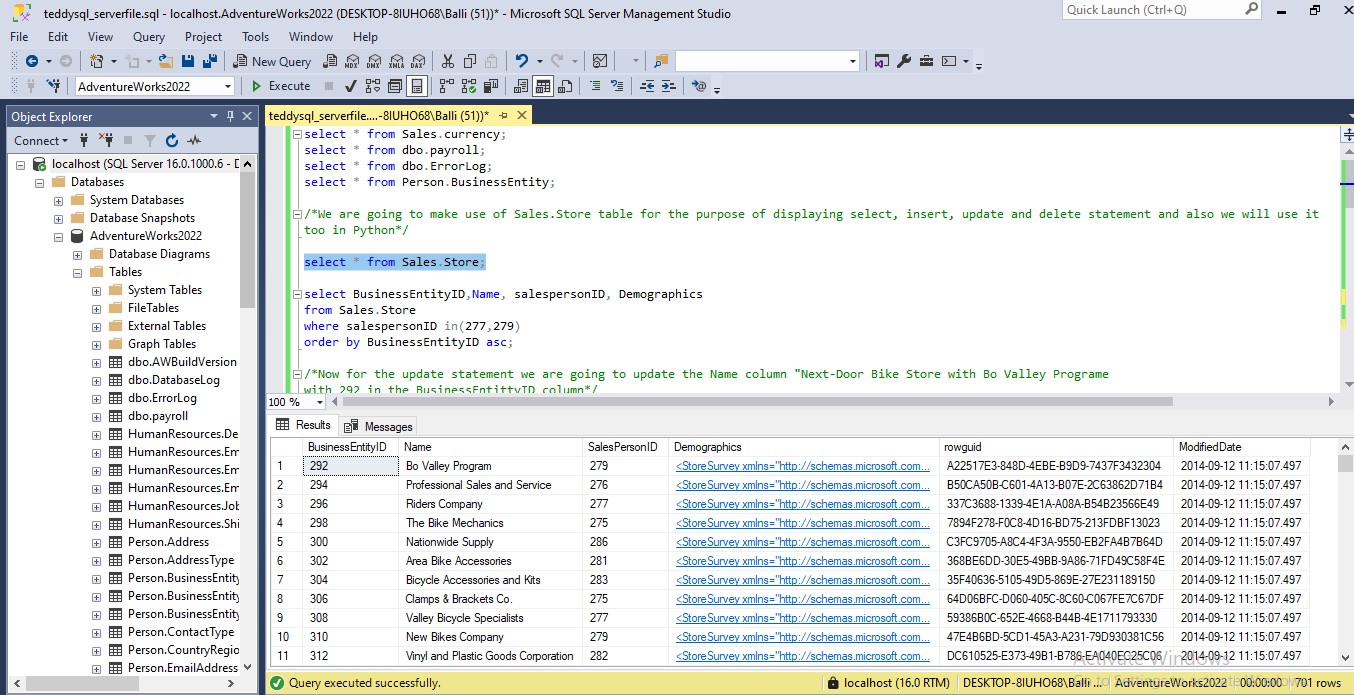
Keeping to the guide above enumerated on a step by step process, I successfully installed Microsoft Sql Server(MSS) with its accompanying tool Sql Server Management Studio(**SSMS):** SSMS is a tool used to write scripts to Microsoft Sql Server giving it instructions to effect specific programing goal or objective.

 Below is a screenshot of Microsoft Sql Sever and Sql Server Management Studio and also SQL scripts or command making data manipulation and data definition languages e.g. select, insert, update and delete statements









Having installed the Microsoft Server in my machine, we can proceed to using python, making use of the python module Sqlalchemy to connect to either AdventureWorks2022 Database or AdventureWorksDW2022 and manipulate data according to the assessment prompts. For this case I utilize the AdventureWorks2022 database so all the python scripts or code is going to administered to it

To further expand our knowledge about python Sqlalchemy and its connection to different database structure or relational database management system(RDBMS). We will be looking at the connection strings (a statement that python Sqlalchemy uses to connect to database system) of three relational database management system (Oracle, Mysql Server, Microsoft Sql Server) that python Sqlalchemy uses.

Oracle: as a norm there are important information that should be configured or made available before any connection to a database or relational database management system. Oracle is no different and this information include

**Username:** this is the user who wishes to access Oracle database

**Password:** the password used by the username or user to gain access

**Service Name:** consisting of the database name and the domain the user wishes to connect to. That is the instance the user will be able to write Sql commands to the specific database

**Port:** oracle uses a port in its connection, usually the default is 1521

**Host name:** this is usually the machine in which the server is located

It is important to talk about oracle listener and Tnsname when discussing connections because this are the two file that must be configured before any connection can be possible. And for this assessment it is not in the scope to discuss further. So I will suggest reading it up. There is another important thing to note in connections to database using python. And it is the python module that is unique to every database system, this module must be present to enable python recognize the database we are trying to connect. For Oracle this python module is Oracledb which is an upgraded version with python 3 and above different from the conventional cx\_oracle

Now that we have fully listed the parameters we need to connect to oracle database using python Sqlalchemy, we cannot proceed to creating the connection strings

Note we will not be discussing further after the connection strings is created because it is not the database we are going to use for this assessment

Python code

***Oracle:connectionstrings=f”oracle+oracledb://{username}:{password}@{host name}:{port}/?service\_name={service name}”***

***Connection becomes***

***Engine =create\_engine(connection strings)***

MySQL: Mysql is another relational database management system that manages databases for its client. To get to MYSQL using python Sqlalchemy we should define the necessary connection parameter as we did with Oracle

As we discussed in above the unique python connection module for MYSQL SERVER is mysqlclient. This is what we will be making use of in defining our connection strings. First thing first let us define our other parameters

**Username:** this is the user who wishes to access Oracle database

**Password:** the password used by the username or user to gain access

**Database name:** this is the database we are connecting to inside the Mysql server

**Port:** Mysql server port for connection, usually the default is 3306

**Host name:** this is usually the machine in which the server is located

Python code

***Mysqlserver:connectionstrings=f”mysql+mysqlclient://{username}:{password}@{host name}:{port}/{database name}”***

***Connection becomes***

***Engine =create\_engine(connection strings)***

Microsoft SQL SERVER

It is no longer news that every server of a relational database management system needs a connection information or parameter to be configured before initiating any connection command using python Sqlalchemy. Pyodbc is the the unique python module for connection to Microsoft sql server just like others above.

**Username:** this is the user who wishes to access Oracle database

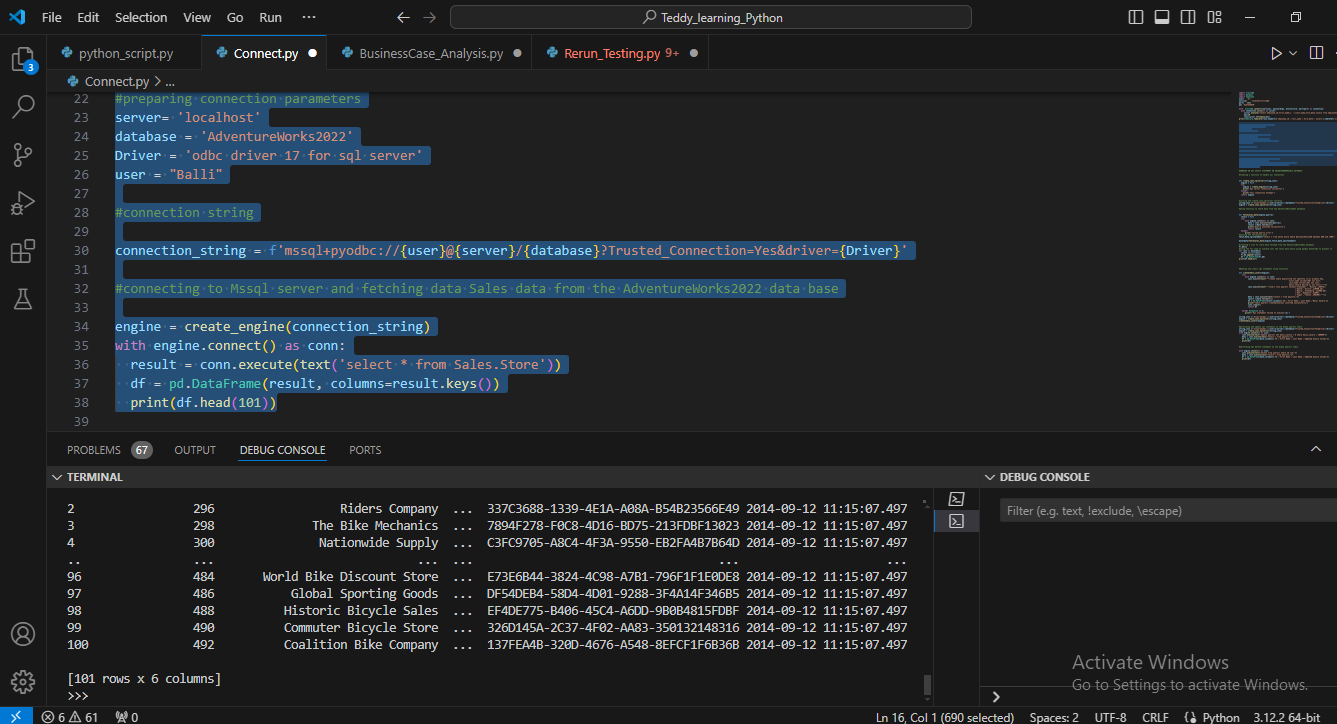
**Password:** the password used by the username or user to gain access

**Database name:** this is the database we are connecting to inside the Mysql server

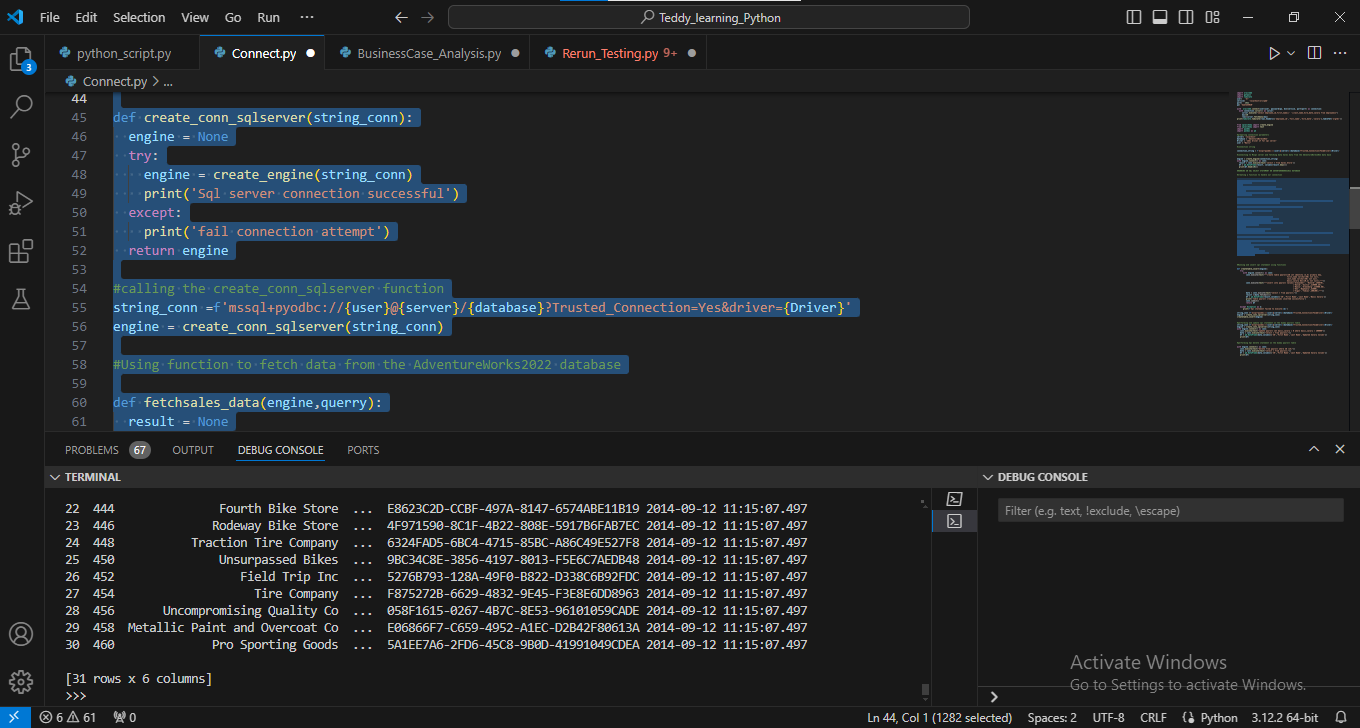
**Driver:** this is a file that aids communication between python and Microsoft sql server

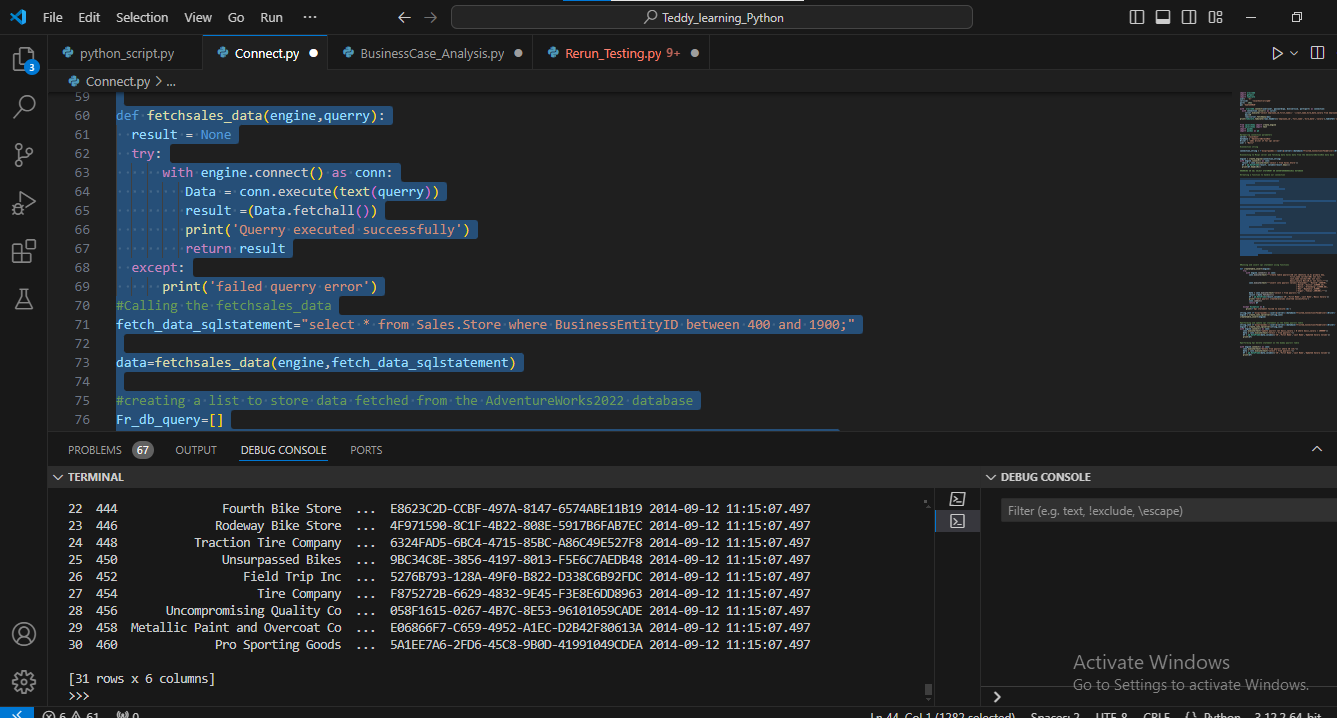
**Host name:** this is usually the machine in which the server is located

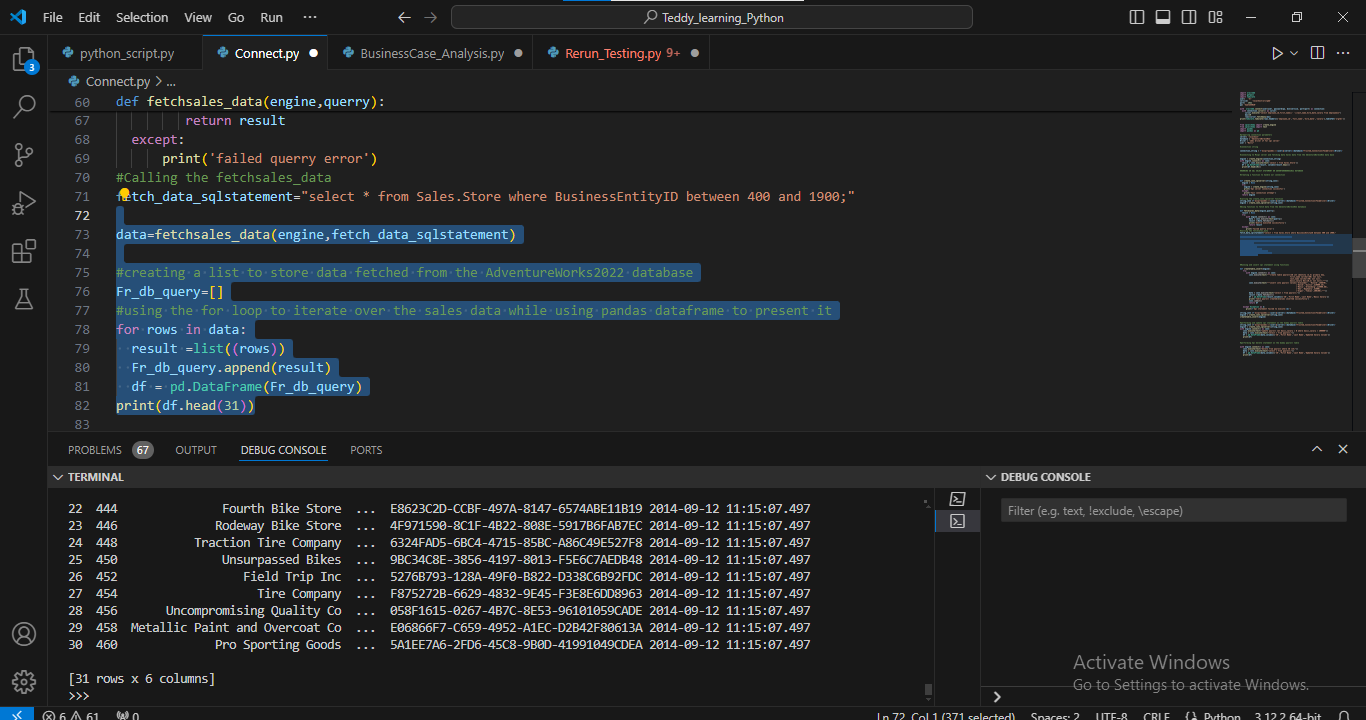
Because our assessment focus on using Microsoft sql server, thus below is a screenshot of python codes defining our connection parameters listed above as well as the connection string used with an “sql select \* from Sales.Store” to test our connection



To further test the connection, I decided to define a function for connection so that when fetching data from the database we can easily call the connection function already defined to secure our connection and afterwards issue our sql statement. Below is a python code screenshots of this connection function with another function defined, making use of the connection function to fetch data from Sales.Store table in the AdventureWorks2022 database whose BusinessEntityID is between 400 and 1900

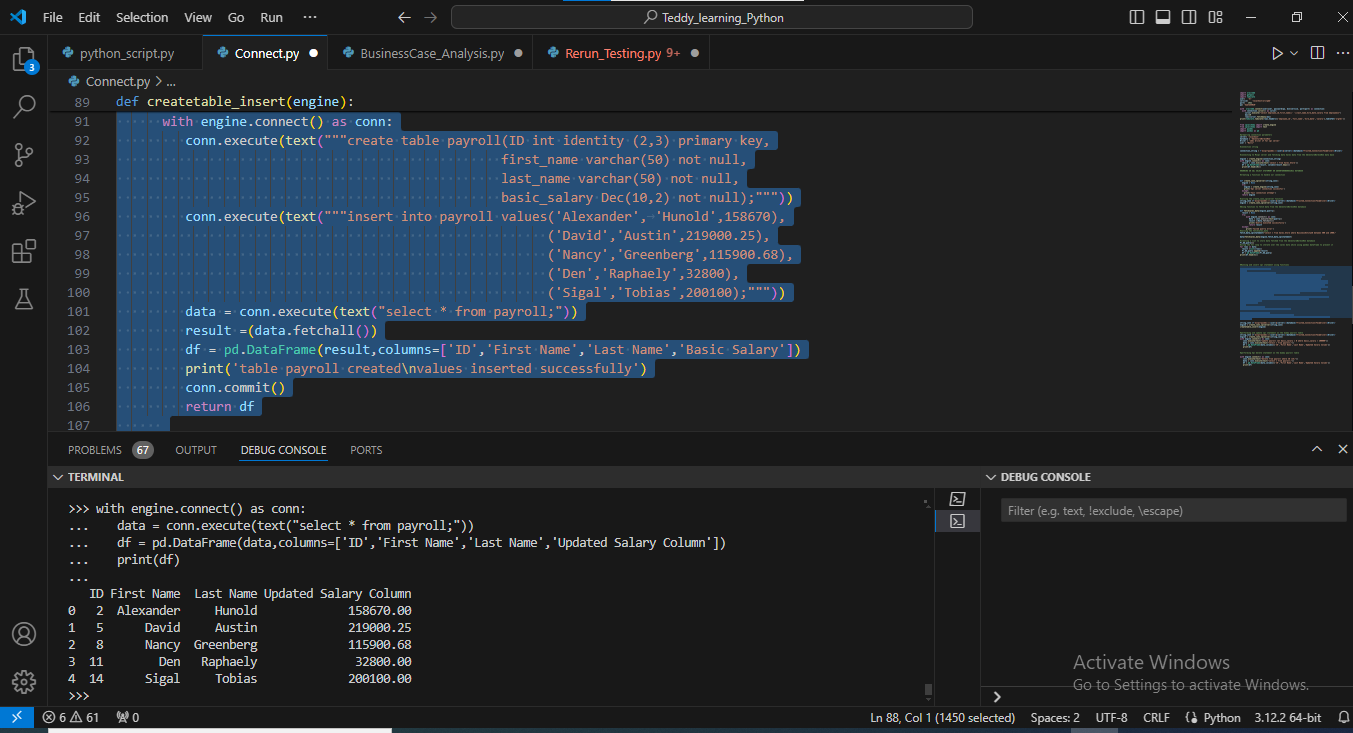




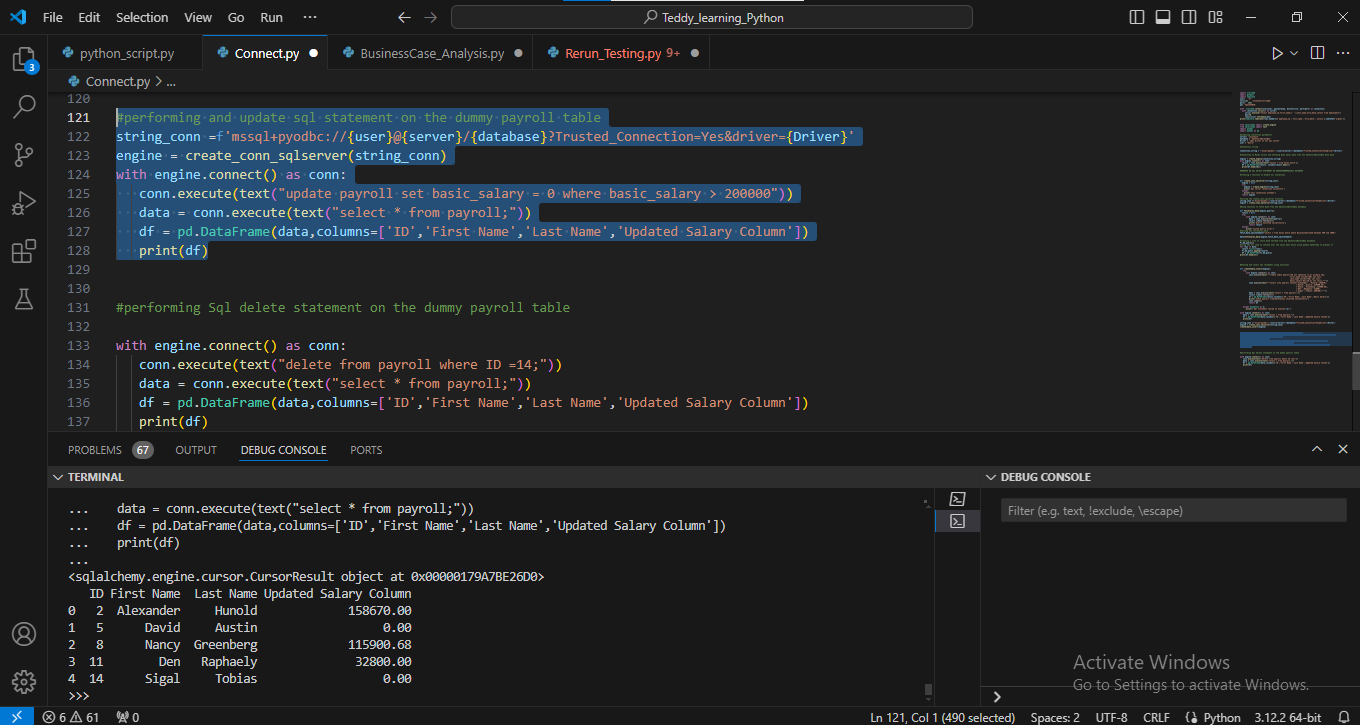


Having tested our connection with and also tried Sql select statement. Going forward I decided to create a new dummy table object called payroll in the AdventureWorks2022 database to answer the select, update, insert and delete sql statement. It was more preferable to use a new table object compared to the default tables in the AdventureWorks2022 database because my initial examples with SSMS already catered for that. So to get more innovative I decide to also issue a create statement to create a dummy table in the database and performed select, insert, update and delete on the table.

**Below is a screenshot of the python code for it**

**Python code that created the payroll table, inserted values and displayed it**

**Python code that displayed an update statement on the dummy payroll table created:**



**Python code that performed an sql delete statement on the created payroll dummy table:**

