**Tableau Introduction-Assignment 2**

1. With the help of the databases of your choice, illustrate the data connection process with a SQL and a NOSQL database server in Tableau. Also, compare and contrast the pros and cons of using a SQL database server and a NOSQL database server.

To connect Tableau with a SQL database server, you'll need to follow these steps:

1. Open Tableau and select the "Connect" option.
2. Choose the SQL database server you want to connect to, such as MySQL, PostgreSQL, or Microsoft SQL Server.
3. Enter the server name, username, and password to establish the connection.
4. Select the specific database within the server that you want to connect to.
5. Tableau will then retrieve the metadata from the database and display the available tables and views.
6. You can then drag and drop the desired tables or views onto the canvas to start analyzing the data.

Now, let's move on to connecting Tableau with a NoSQL database server:

1. Similarly, open Tableau and select the "Connect" option.
2. Choose the specific NoSQL database server you want to connect to, such as MongoDB or Cassandra.
3. Provide the necessary connection details, including the server name, port, and authentication credentials.
4. Once the connection is established, Tableau will fetch the available collections or tables from the NoSQL database.
5. You can then select the desired collections/tables and start analyzing the data using Tableau's visual analytics capabilities.

Now, let's compare and contrast the pros and cons of using a SQL database server and a NoSQL database server:

SQL Database Server: Pros:

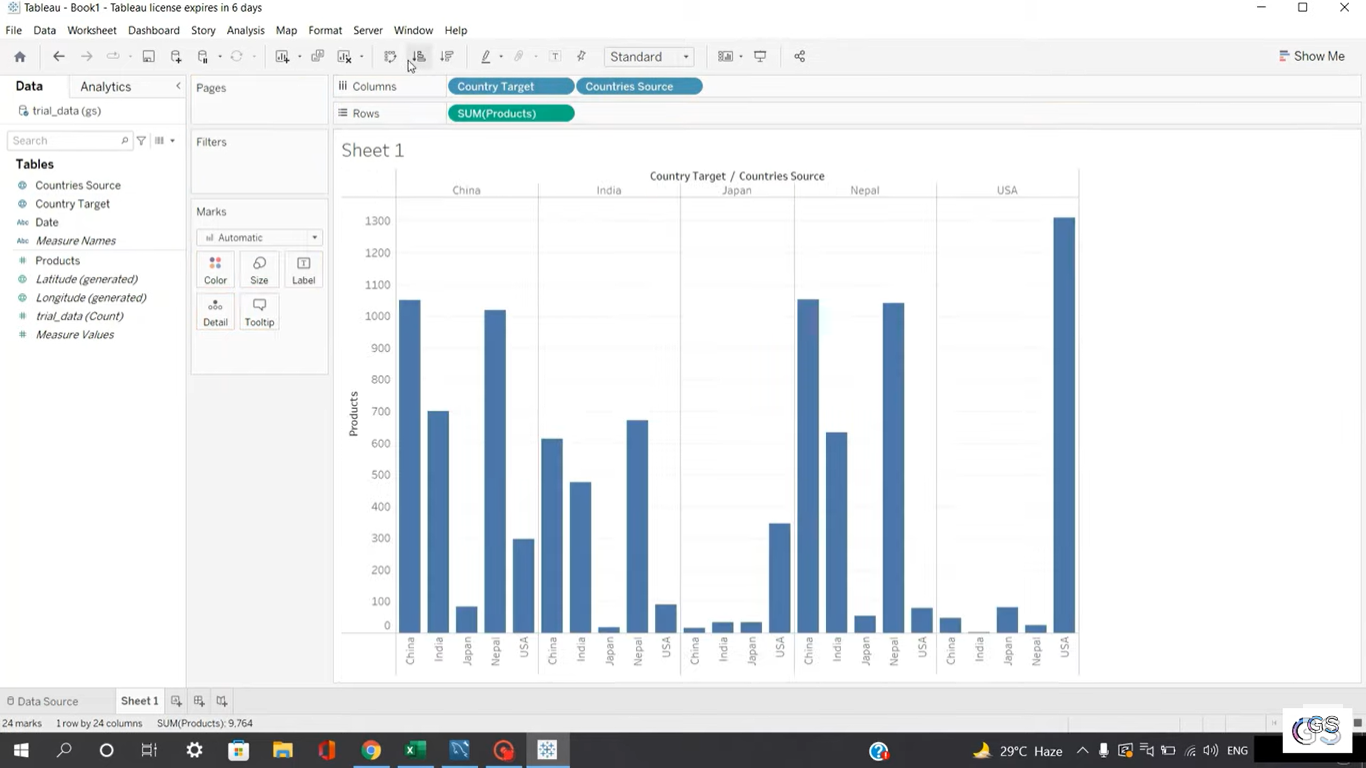
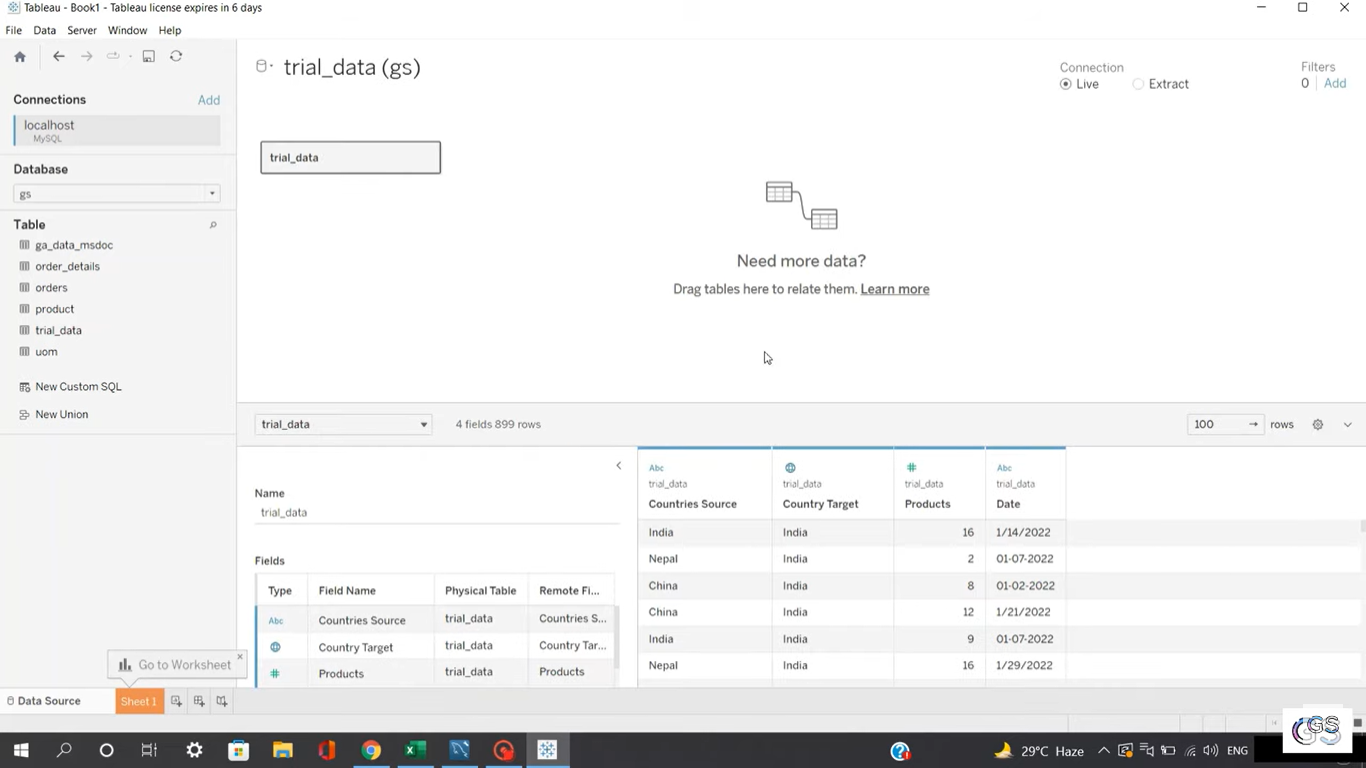
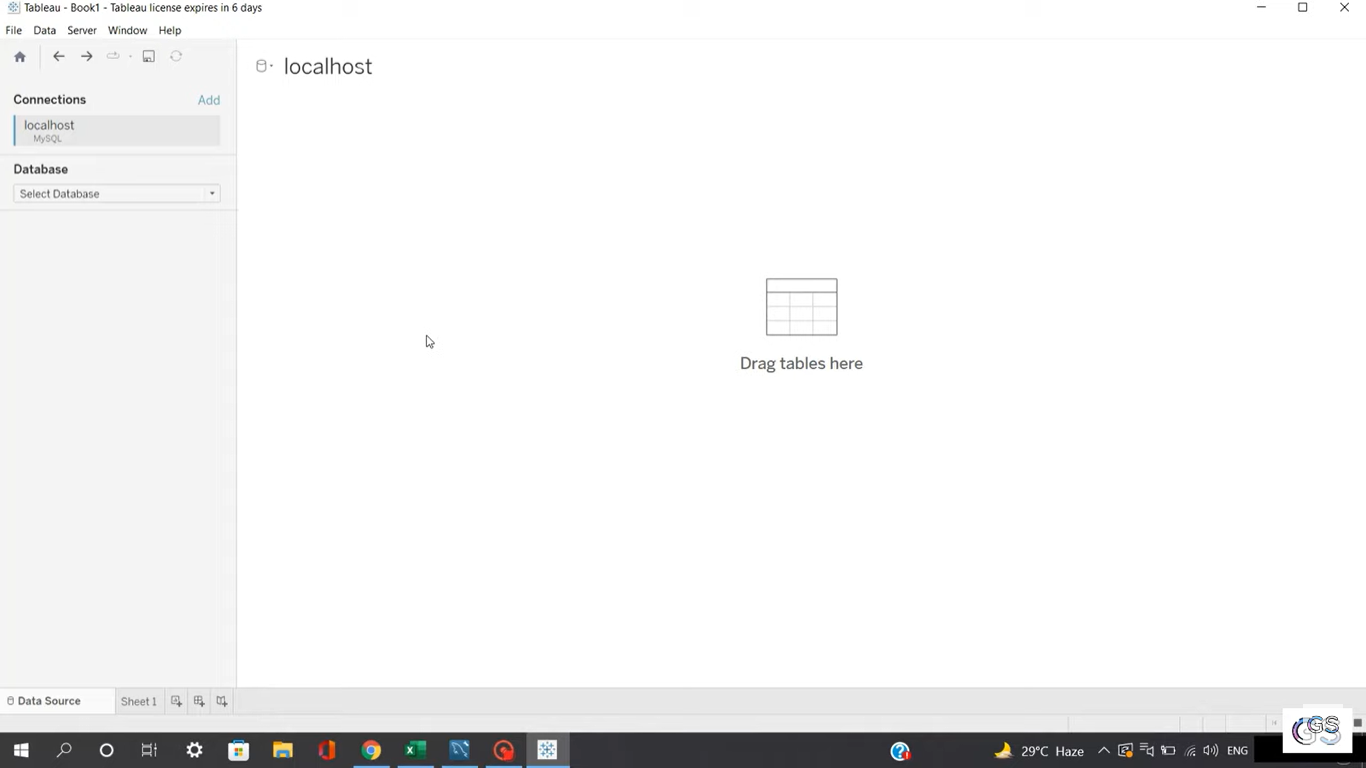
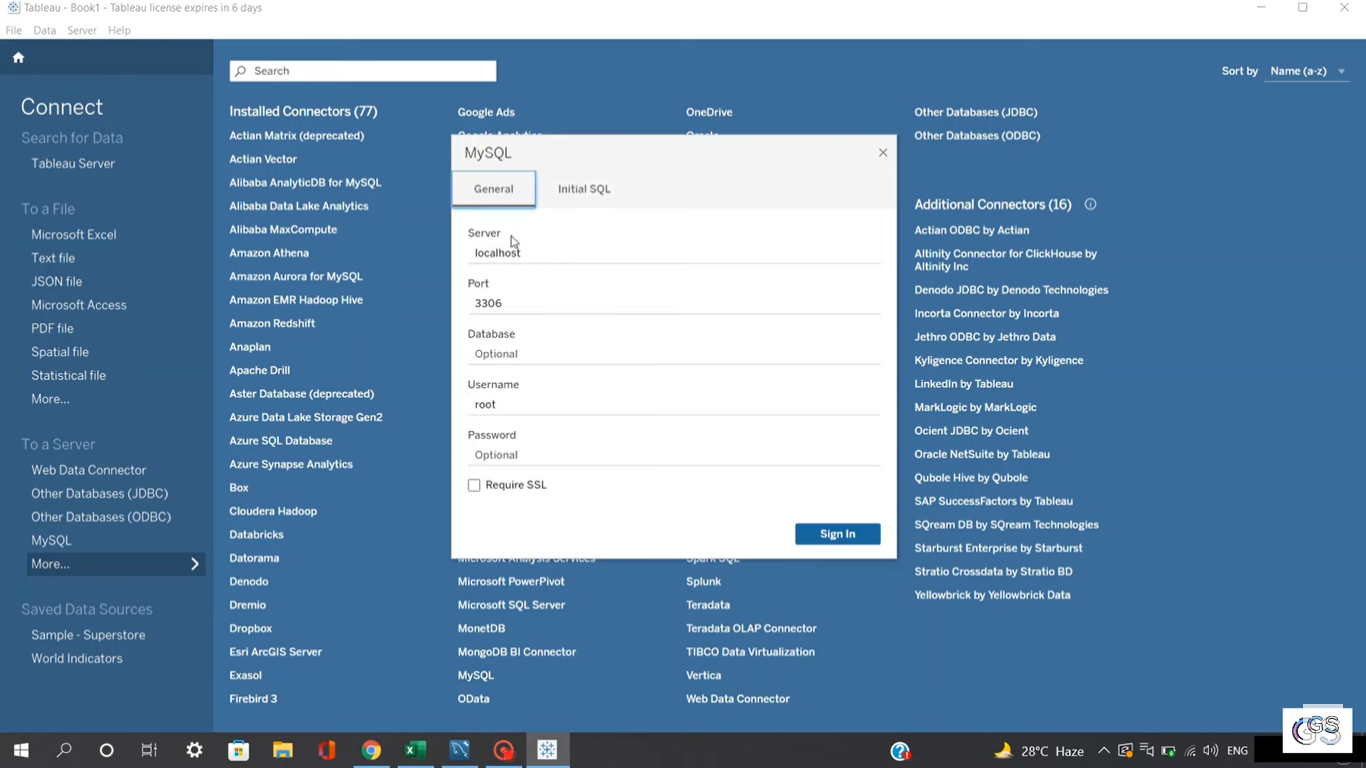
1. Structured Data: SQL databases are best suited for structured data, providing a well-defined schema.
2. ACID Compliance: SQL databases guarantee Atomicity, Consistency, Isolation, and Durability, ensuring data integrity.
3. Mature Ecosystem: SQL databases have been around for a long time, resulting in a mature ecosystem with extensive support and tools.
4. Complex Queries: SQL databases excel at handling complex relational queries using SQL syntax.

Cons:

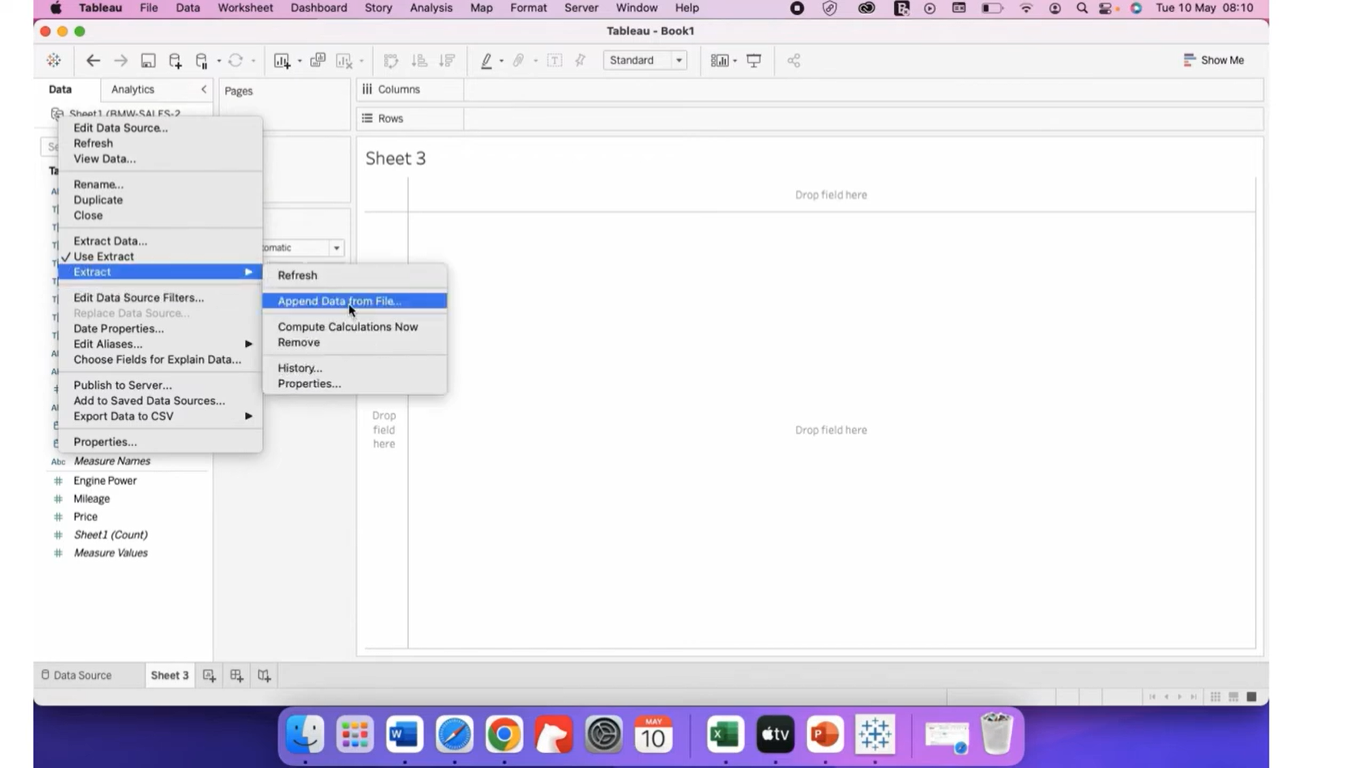
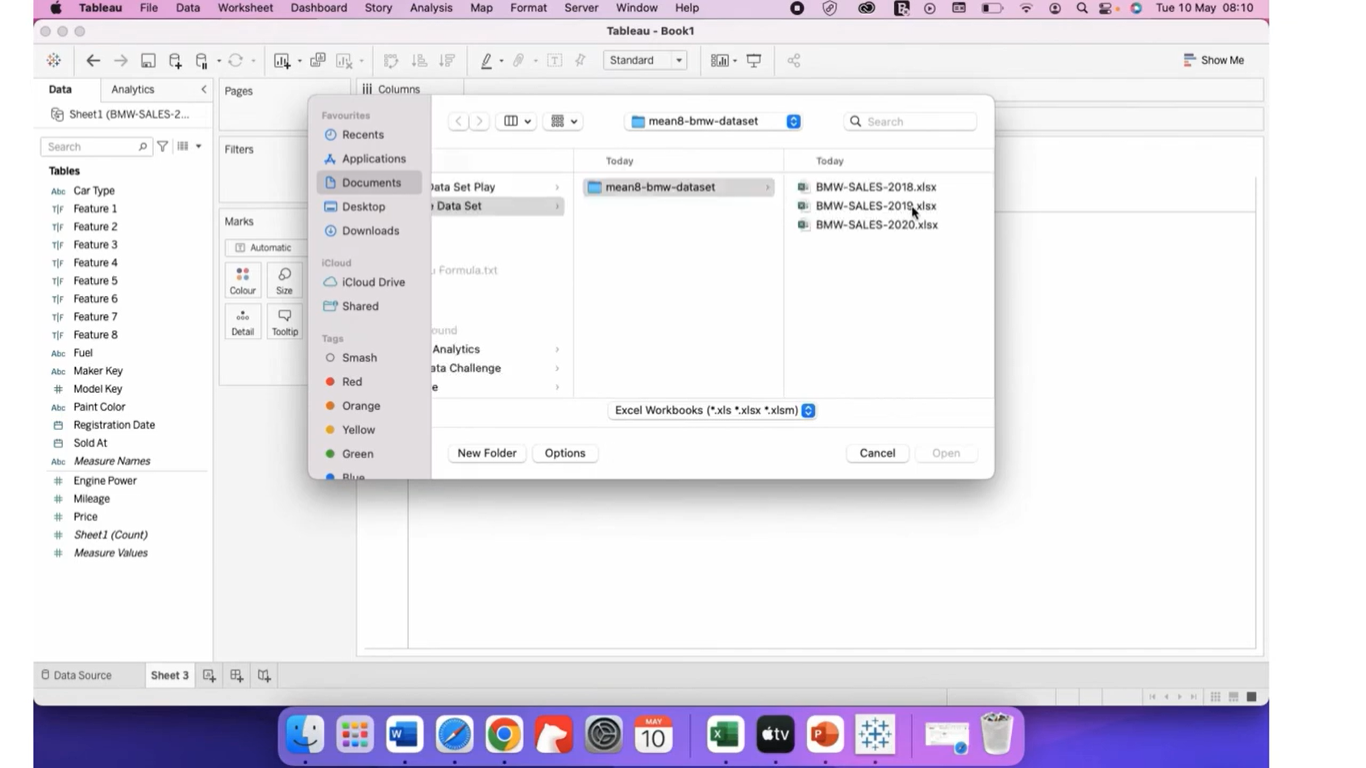
1. Limited Scalability: SQL databases may face scalability challenges when dealing with large datasets or high write loads.
2. Schema Rigidity: Changing the schema of a SQL database can be cumbersome, especially for evolving data requirements.
3. Less Flexible Data Model: SQL databases have a fixed schema, making it difficult to handle unstructured or semi-structured data.

NoSQL Database Server: Pros:

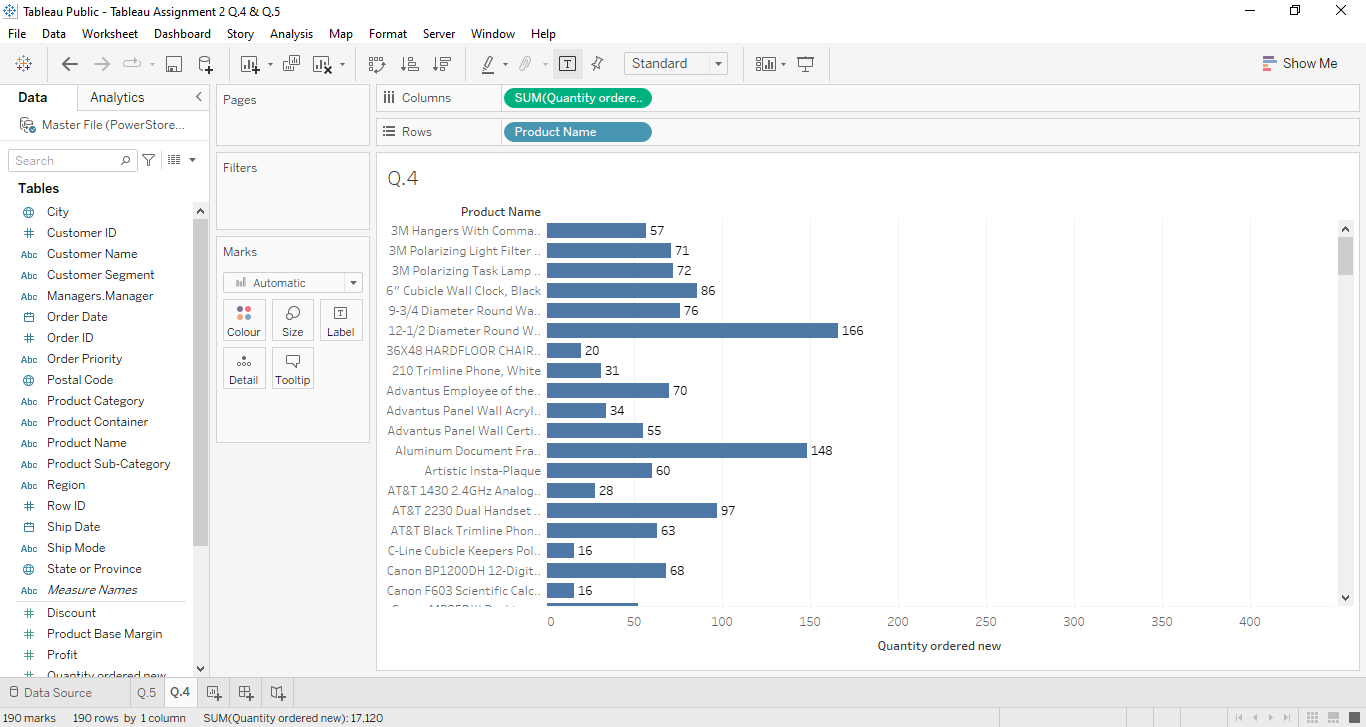
1. Flexible Data Model: NoSQL databases allow storing unstructured or semi-structured data, providing flexibility in handling diverse data types.
2. Connect with SQL database server of your choice, establish a connection with a dummy database “Employee” containing employee information (employee name, employee id, employee salary, employee department, employee years of experience) in a company. Write an SQL statement using tableau’s custom SQL feature to retrieve the employee id and employee salary in your tableau dashboard.



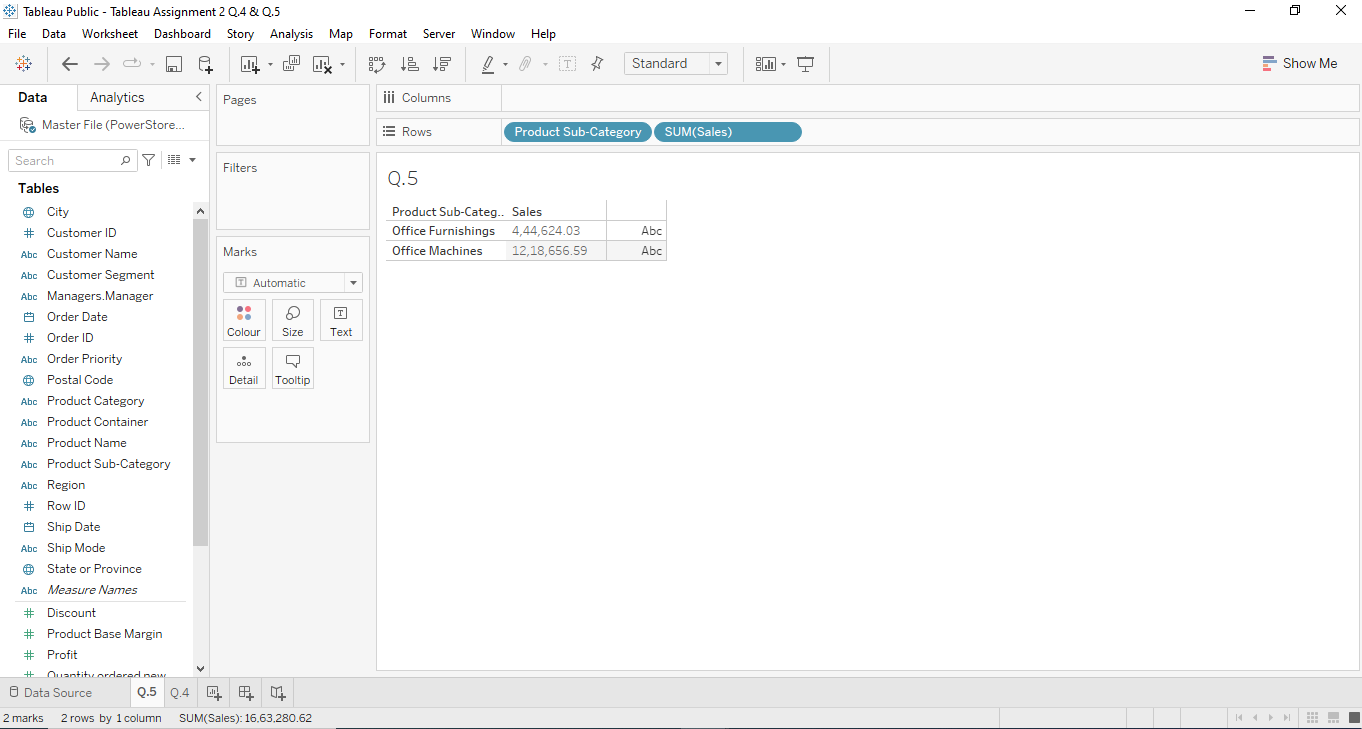
1. Imagine you are tasked with maintaining year-wise data of new students that join a certain college. Create three separate excel tables containing data of students for three consecutive years, one for each year and store it in a common database. Using the features available in the data source page of Tableau, join these tables with the same headers and create a single table.

1. Using the “PowerStore\_USA'' dataset, filter and display those products which satisfy the condition wherein the number of quantity or dered is more than the total average quantity ordered.



1. With respect to the “PowerStore\_USA'' dataset, apply a filter to your data such that only those values where the “Product-Category” starts with the letter “O” get displayed.



[*PFA links for the datasets used.*](https://drive.google.com/drive/folders/123UyMRbrReCjyn1K4g_FhsjKx6cP4zLH?usp=sharing)