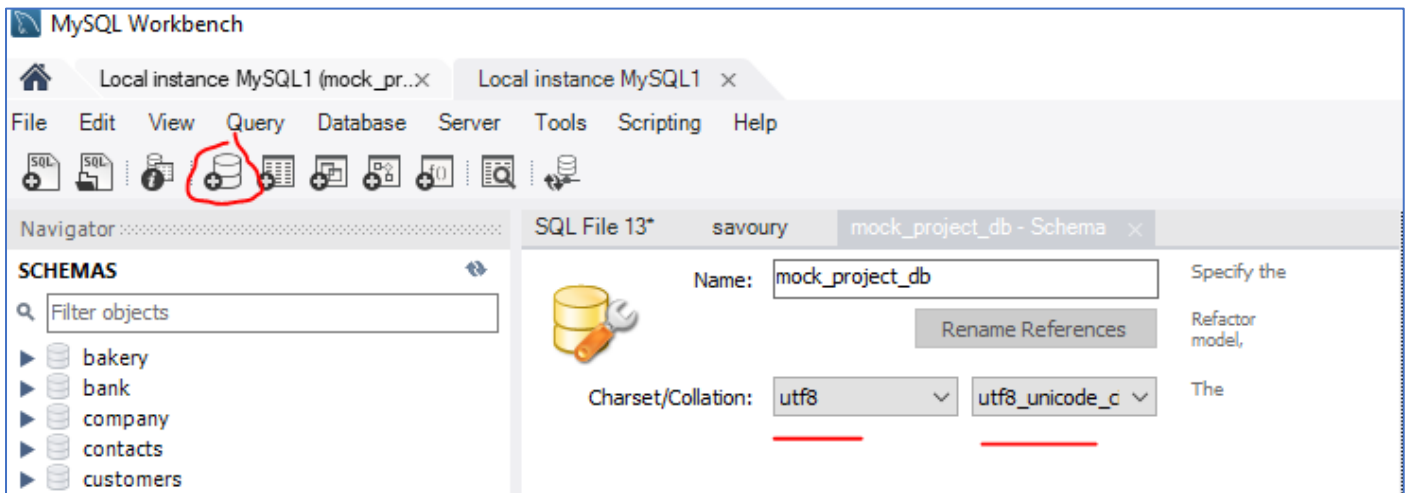


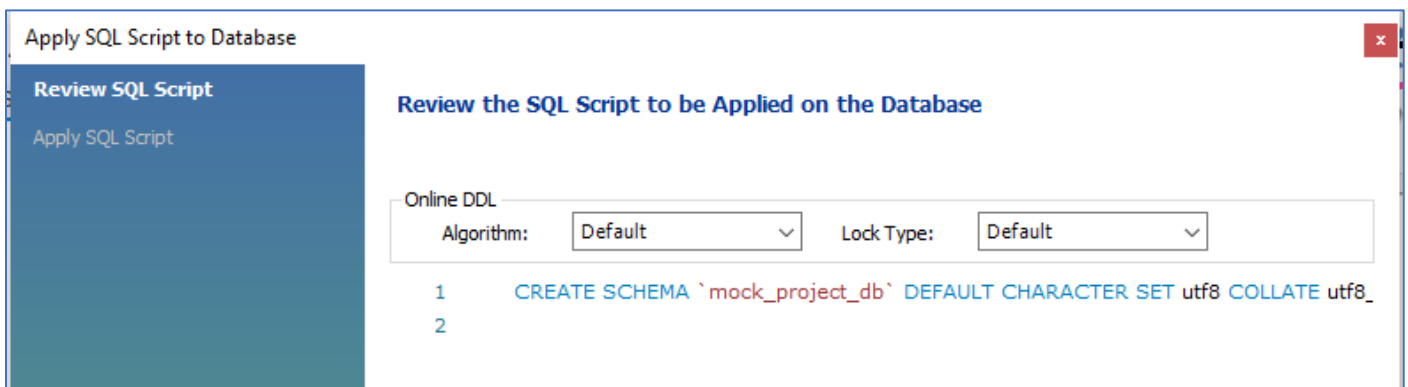
MOCK PROJECT GUIDELINES

1. Design and Create a new DB

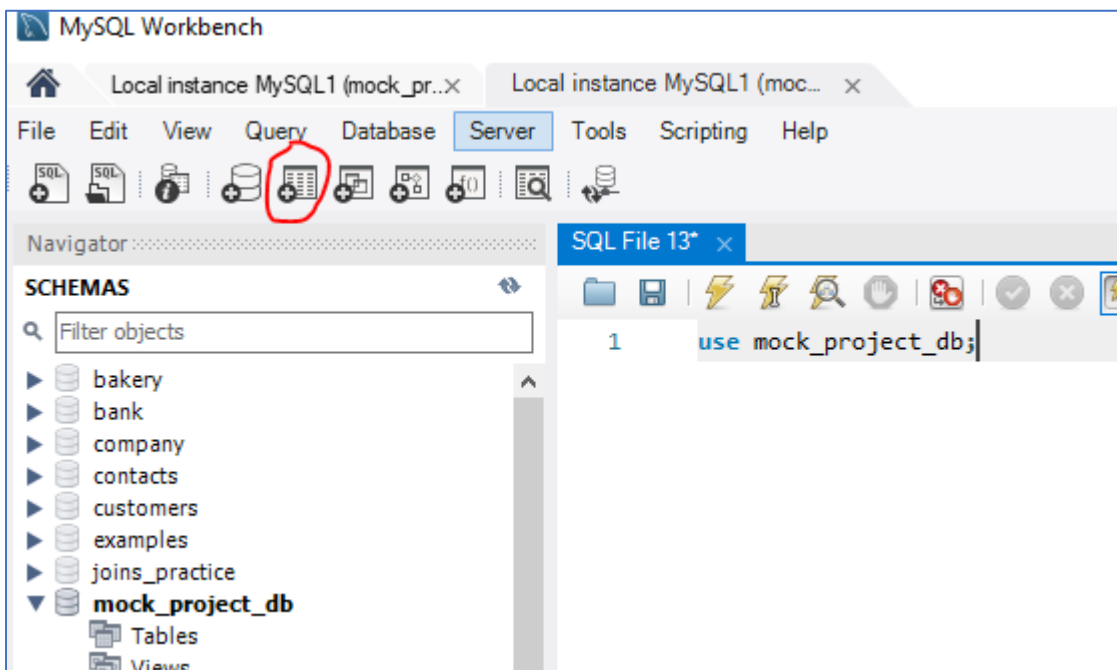
- Create an empty shell for a DB using Workbench DB design tools. (not SLQ code statements, as we want to demonstrate another way of creating a DB)



- Click Apply button
- And again



- When the table is created, programmatically use the relevant DB and then click add a new table icon.



- We are aiming to create a table called orders, just like here (you have a csv file in your folder that contains all data)

	A	B	C	D
1	order_id	total_price	cust_id	
2	9001	23.2	1000	
3	9002	123.2	1001	
4	9003	7.9	1002	
5	9004	12.3	1003	

- IMPORTANT: use workbench tools to add column names, types, put constraints (PK, FK etc)

The screenshot shows the MySQL Workbench Table Designer interface for a table named 'orders' in the 'mock_project_db' schema. The table is configured with the following settings:

- Table Name:** orders
- Schema:** mock_project_db
- Charset/Collation:** utf8 / Default Collation
- Engine:** InnoDB
- Comments:** (Empty text area)

The column list shows one column:

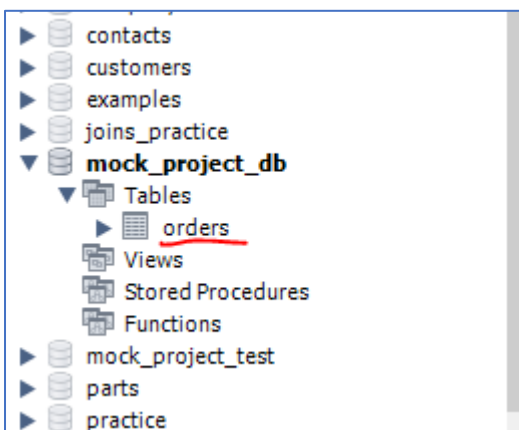
Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
order_id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

The column configuration panel for 'order_id' shows:

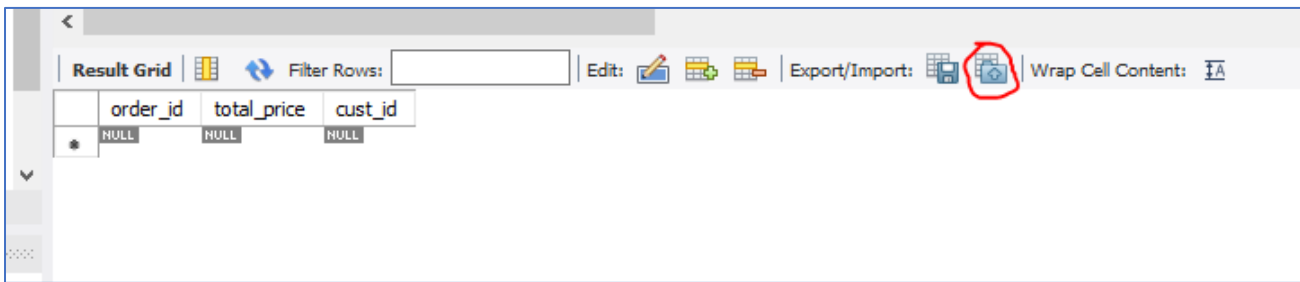
- Column Name:** order_id
- Data Type:** INT
- Charset/Collation:** Default Charset / Default Collation
- Comments:** (Empty text area)
- Storage:**
 - ☐ Virtual ☐ Stored
 - ☒ Primary Key ☒ Not Null ☐ Unique
 - ☐ Binary ☐ Unsigned ☐ Zero Fill
 - ☐ Auto Increment ☐ Generated

At the bottom, there are tabs for Columns, Indexes, Foreign Keys, Triggers, Partitioning, and Options. 'Apply' and 'Revert' buttons are at the bottom right.

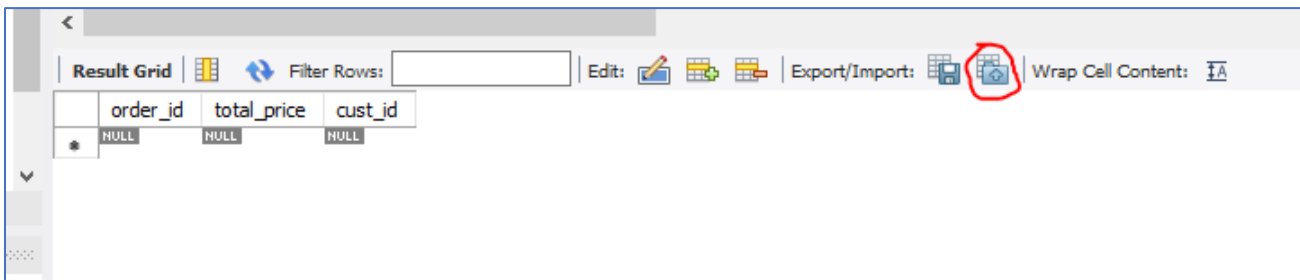
- Now that all columns are added, we have a table



- Run a query to return everything from a newly created table – it would return an empty table.
- Now we want to load CSV data into our table.
- Now click this button to upload data from an external file. Follow all steps



- Once that done we are going to create another table called customers (you have csv file). Do not create a new table shell, we can just load a new table on the fly, so click the same button again:



- This time choose the option 'create new table' and proceed.

Select destination table and additional options.

☐ Use existing table: mock_project_db.orders

☒ Create new table: mock_project_db customers

☒ Drop table if exists

SQL File 13* orders customers orders - Table x

Table Name: orders Schema: mock_project_db

Charset/Collation: utf8 utf8_bin Engine: InnoDB

Comments:

Foreign Key Name	Referenced Table	Column	Referenced Column
fk123	mock_project_db.`customers`	<input checked="" type="checkbox"/> cust_id	cust_id

Foreign Key Options

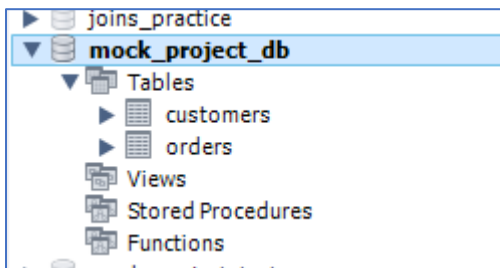
On Update: NO ACTION

On Delete: NO ACTION

☐ Skip in SQL generation

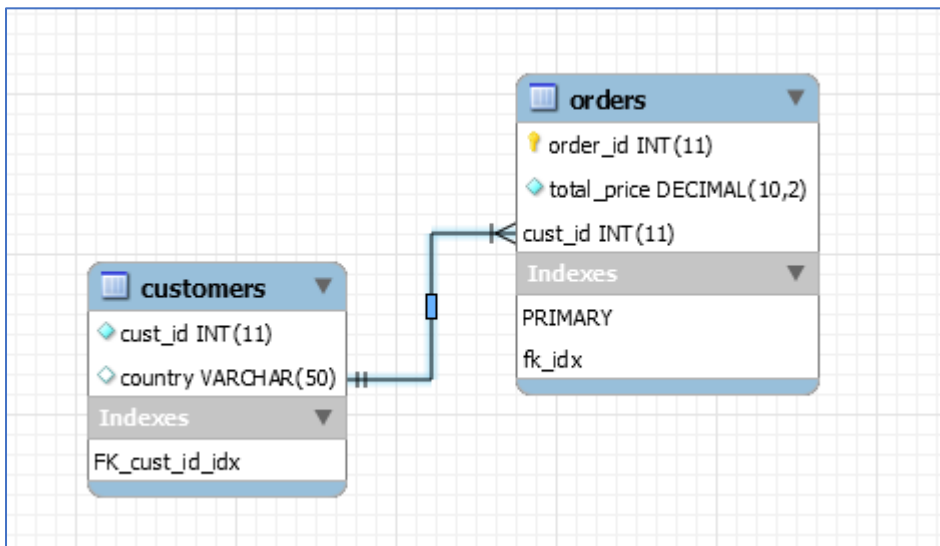
- Foreign Key Comment

- Ensure that everything is created and data added to the tables:



2. Create ER diagram for the new DB

- Database → Reverse Engineer → follow the steps.



3. Write queries to select required sample data

- Select the sum of money spent per customer

SQL File 13* orders customers customers orders

Limit to 500 rows

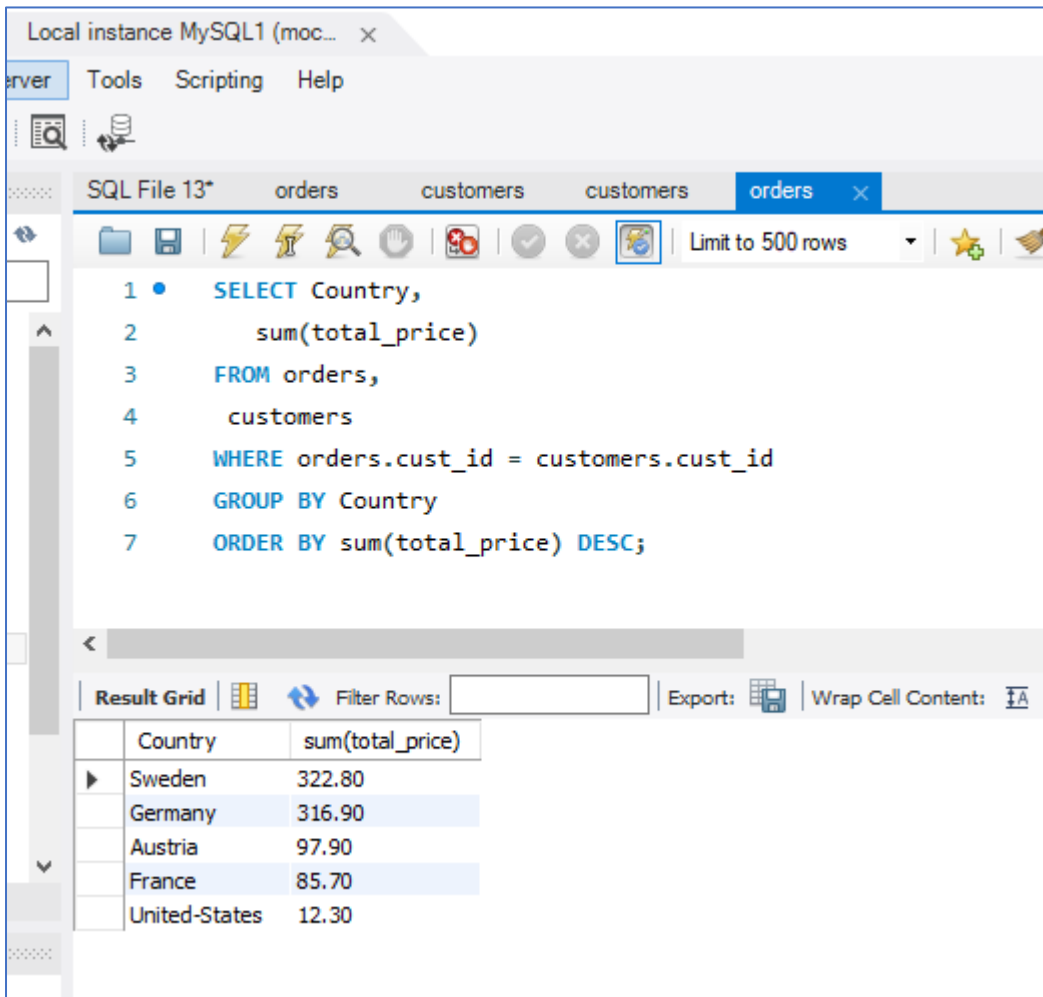
```

1  SELECT cust_id,
2         sum(total_price)
3  FROM orders
4  GROUP BY cust_id
5  ORDER BY sum(total_price) DESC;
  
```

Result Grid | Filter Rows: | Export: | Wrap Cell Center

	cust_id	sum(total_price)
▶	1001	232.20
	1005	146.70
	1000	134.40
	1002	97.90
	1006	90.60
	1007	54.40
	1004	35.80

- Revenue customer-country



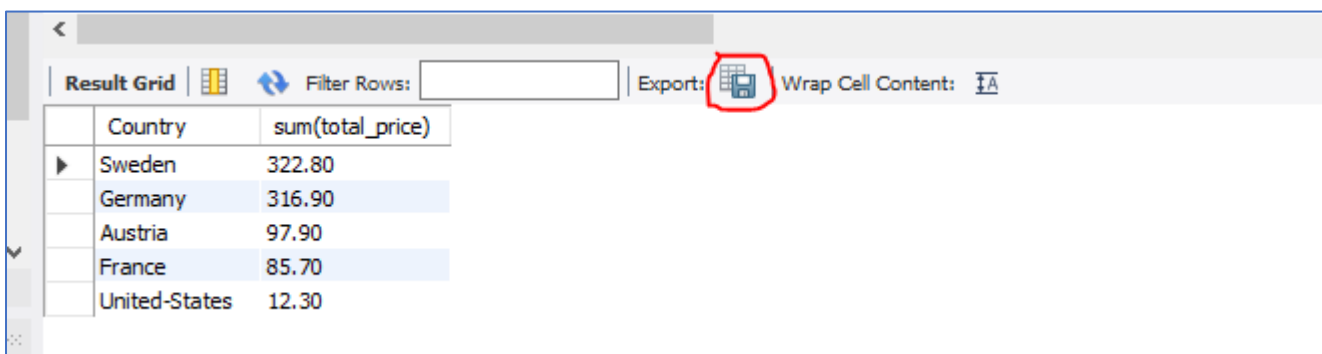
The screenshot shows the MySQL Workbench interface. The top menu bar includes 'Server', 'Tools', 'Scripting', and 'Help'. Below the menu is a toolbar with various icons. The main window displays a SQL query in a text editor:

```
1 • SELECT Country,  
2     sum(total_price)  
3 FROM orders,  
4     customers  
5 WHERE orders.cust_id = customers.cust_id  
6 GROUP BY Country  
7 ORDER BY sum(total_price) DESC;
```

Below the query editor is a 'Result Grid' tab. The grid shows the following data:

Country	sum(total_price)
Sweden	322.80
Germany	316.90
Austria	97.90
France	85.70
United-States	12.30

4. Export data into a new csv file

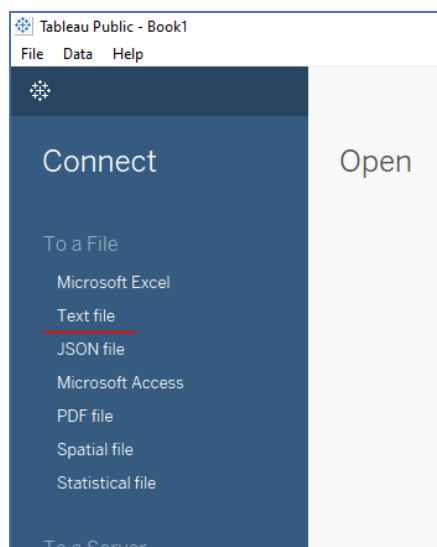


This screenshot is a close-up of the 'Result Grid' tab from the previous image. The 'Export' button, which is a small icon of a document with a grid, is circled in red. The 'Filter Rows' field is empty, and the 'Wrap Cell Content' button is visible to the right of the 'Export' button.

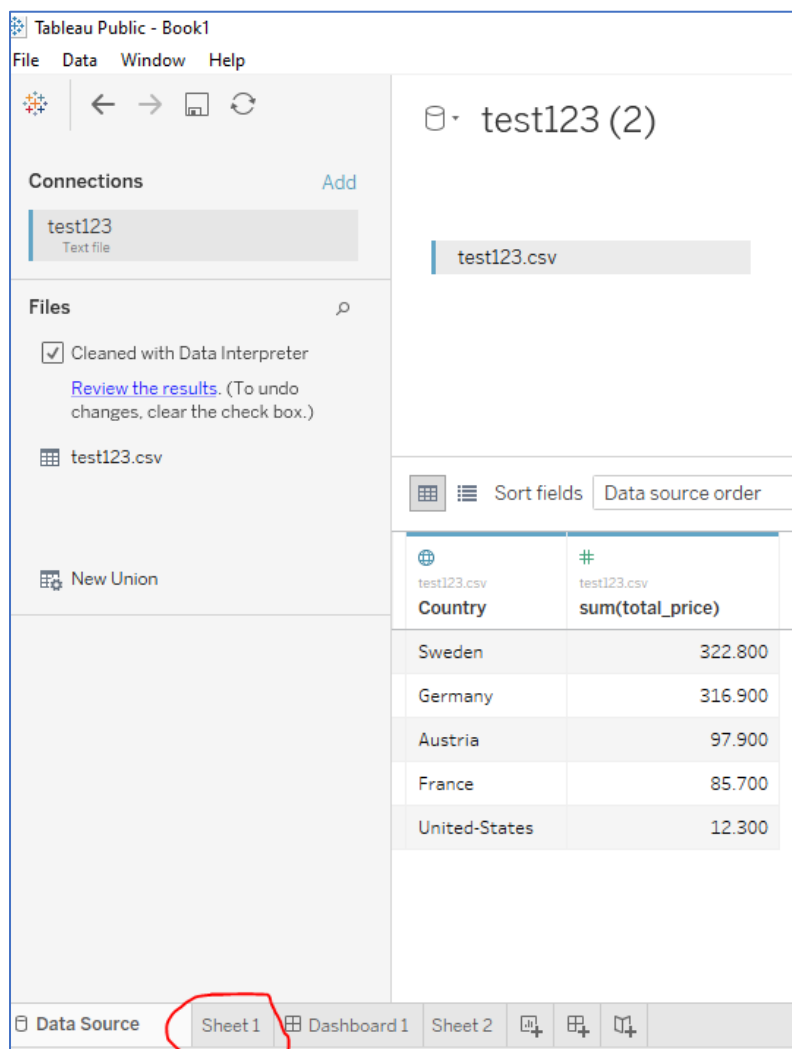
- Click the export button and export data into a new csv file.

5. Visualise resulting data set with Tableau

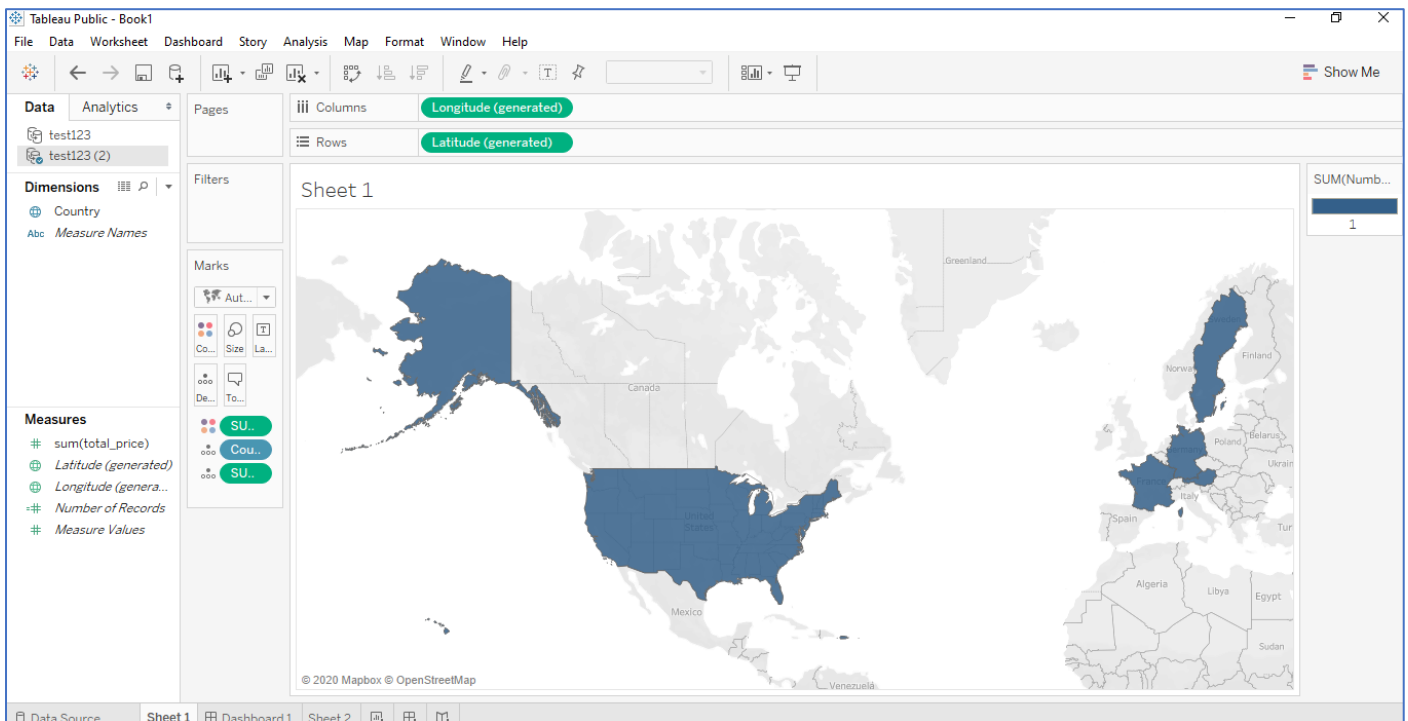
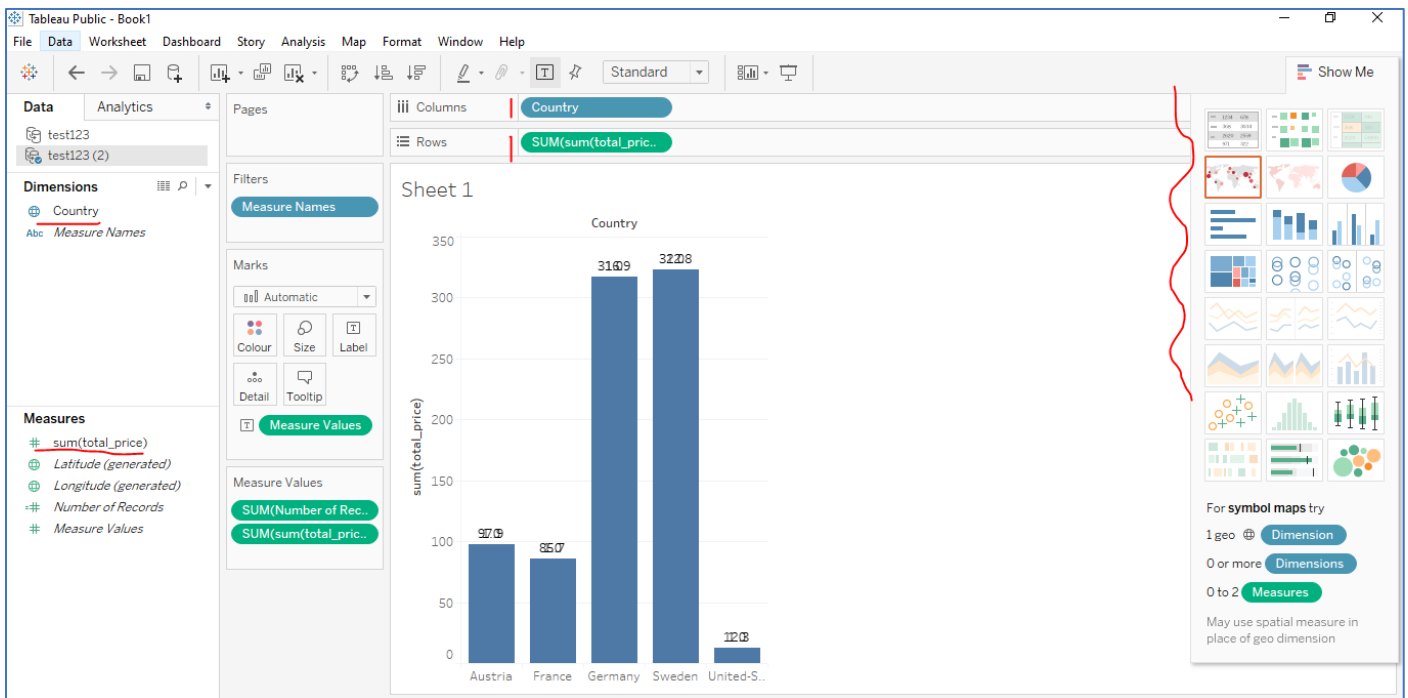
- Start up Tableau. In order to use csv file, we need to connect to a source that would allow us to load a text file, so in the start up panel choose the **Text file** option.



- Load the csv file with data. So you get to this screen.
- And then click the Sheet tab at the bottom



- Drag Country / Sum data sets into column / rows fields to see different types of graphs that can be produced.



- Save your work (you need to create a free account with Tableau Public and then you can download your sheet with data).

