

TOLANI COLLEGE OF COMMERCE
(Affiliated to University of Mumbai)
Sher E Punjab Colony, Andheri East
MUMBAI-MAHARASHTRA-400093
DEPARTMENT OF BSc (INFORMATION TECHNOLOGY)



CERTIFICATE

This is to certify that the Journal entitled, "**Business Intelligence**", is bonafied work of **Yuvraj Vijay Achrekar** bearing roll no: **03** submitted in partial fulfilment of the requirements for the award of degree of **BACHELOR OF SCIENCE** in **INFORMATION TECHNOLOGY** from University of Mumbai, during the academic year 2022 - 2023.

Internal Examiner

Coordinator

External Examiner

Date:

College Seal

INDEX

<u>Sr. No.</u>	<u>Date</u>	<u>Practical</u>	<u>Sign</u>
0		Installation and Setup of Softwares A) Power BI B) R Studio	
1		Import the Legacy Data from Different Sources A) Importing Excel File B) Importing O-Data File	
2		ETL Process A) Removing columns B) Changing Data Types C) Expanding the Order Details table D) Calculating Line Total E) Renaming and Reordering Columns F) Combine the Products and Total Sales	
3		Implementation of Classification Algorithm in R Programming	
4		K- means Clustering using R Programming	
5		Prediction using Linear Regression in R Programming	
6		Perform the logistic regression on the given data warehouse data	
7		Apply What-if Analysis for Data Visualization.	
8		Data Analyzing using Time-series Analysis.	
9		Implementation of Decision Tree using R Tool	

PRACTICAL 0 – Installation and Software Setup

- A.) Power Bi Software**
- B.) R Studio Software**
- C.) SQL Server 2012 Software Setup and Installation:**

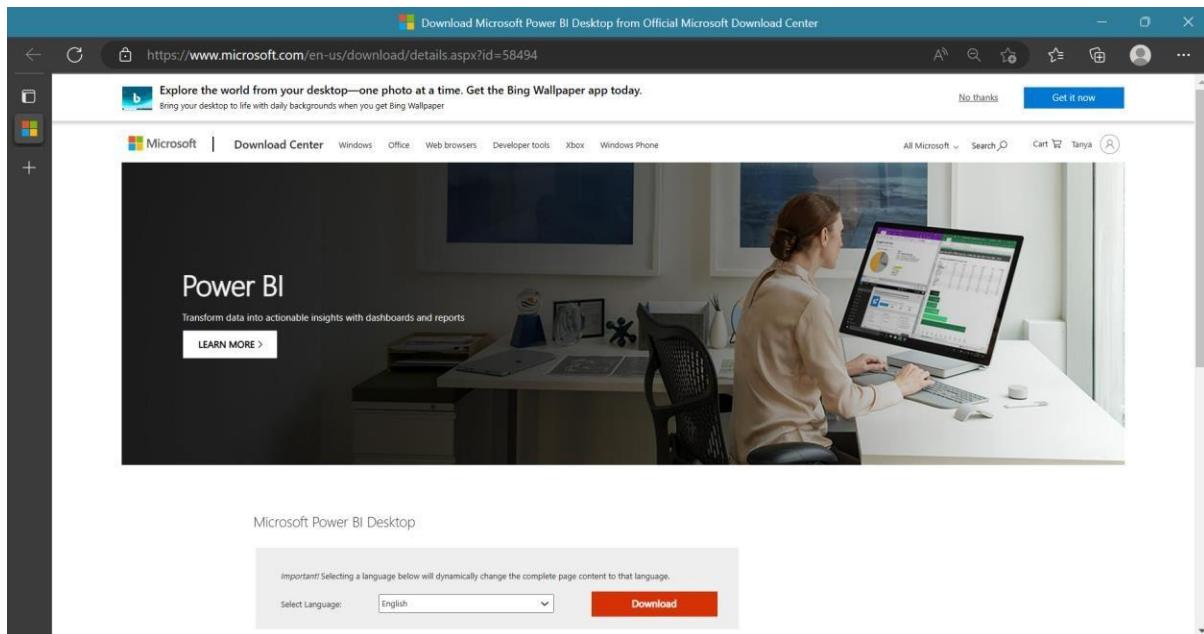
A) Power Bi Software Setup and Installation:

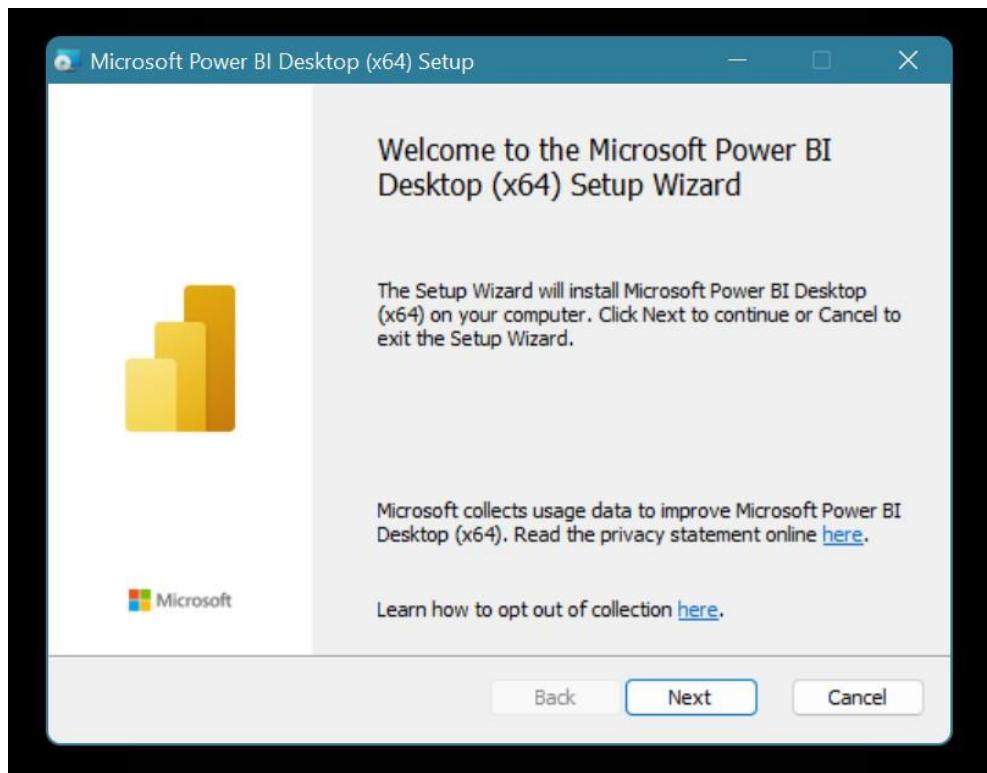
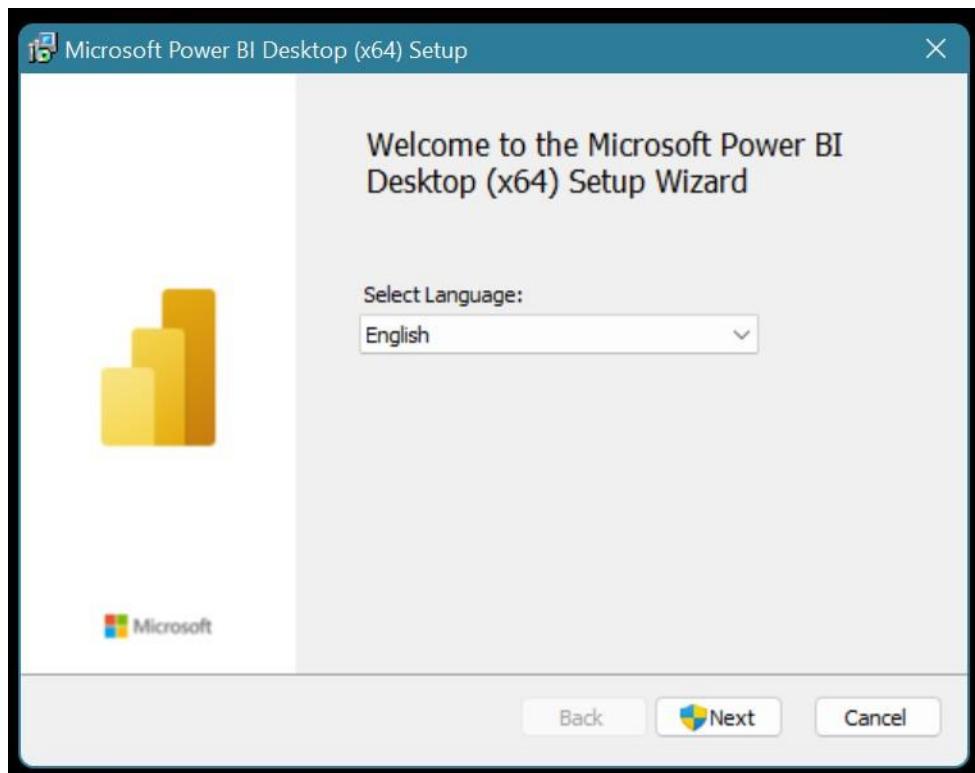
Power BI is a collection of software services, apps, and connectors that work together to turn your unrelated sources of data into coherent, visually immersive and interactive insights. Your data may be an Excel spreadsheet, or a collection of cloud-based and on-premises hybrid data warehouses, Power BI lets you easily connect to your data sources, visualize, and discover what's important, and share that with anyone or everyone you want.

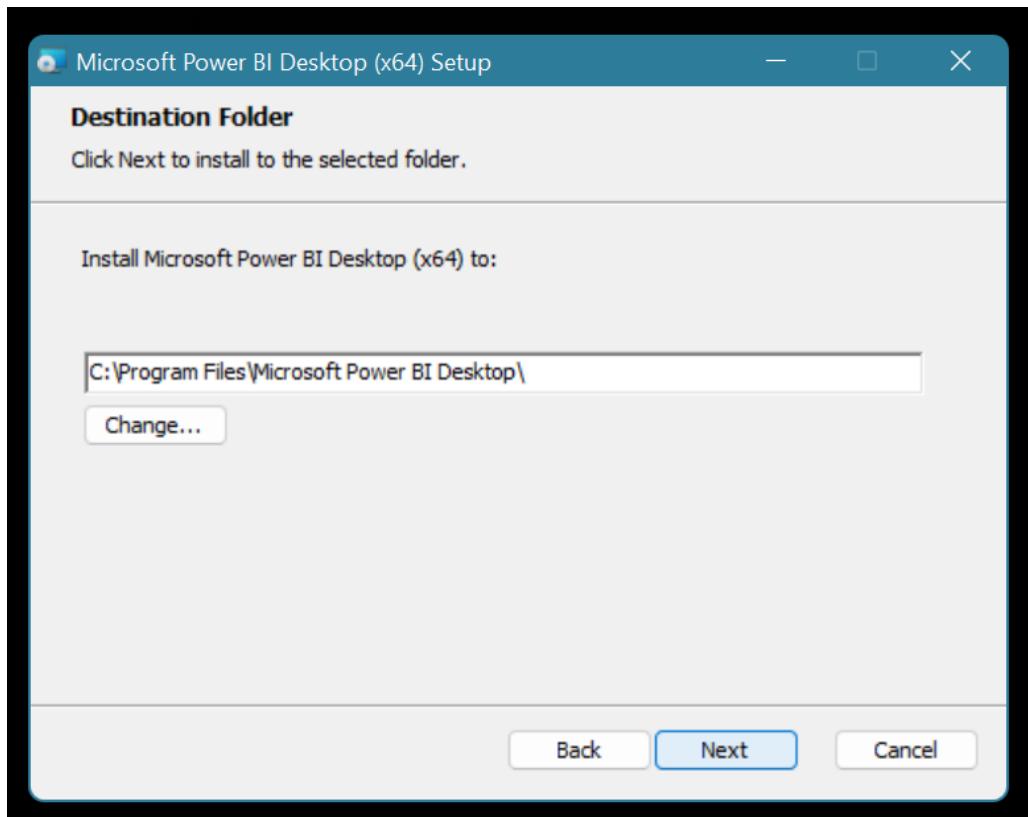
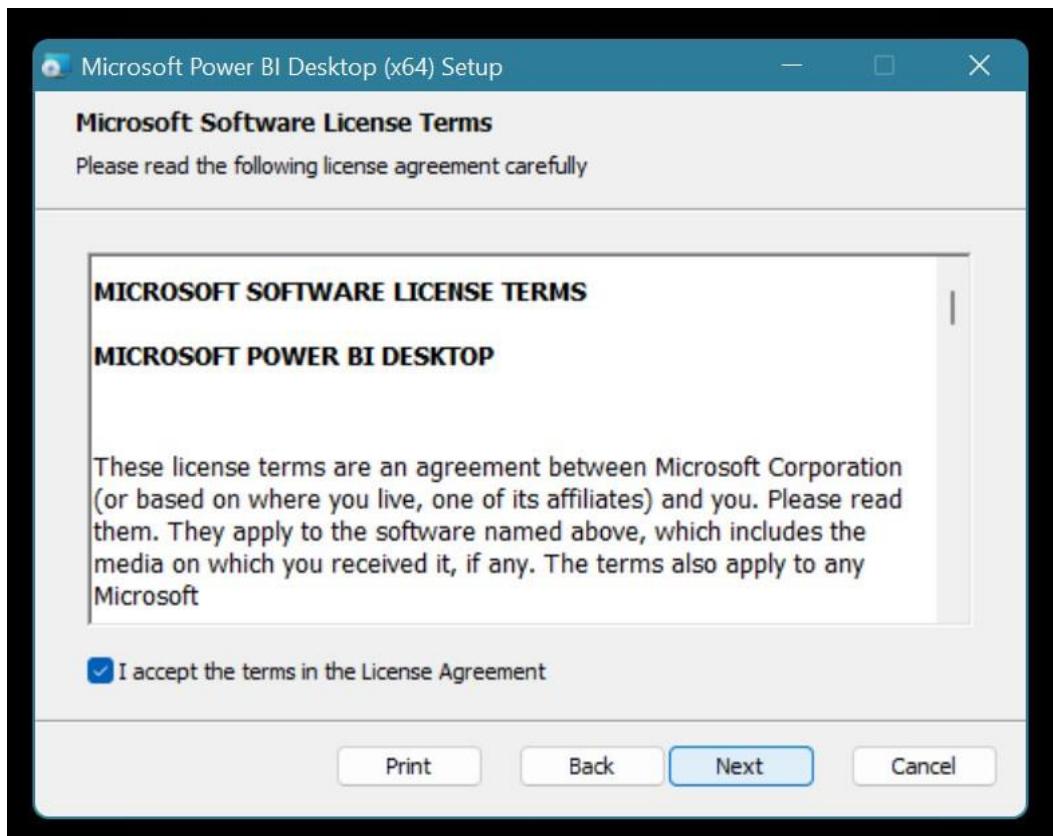
Download Power Bi software by clicking on the link given below and follow the following screenshots for further setup steps:-

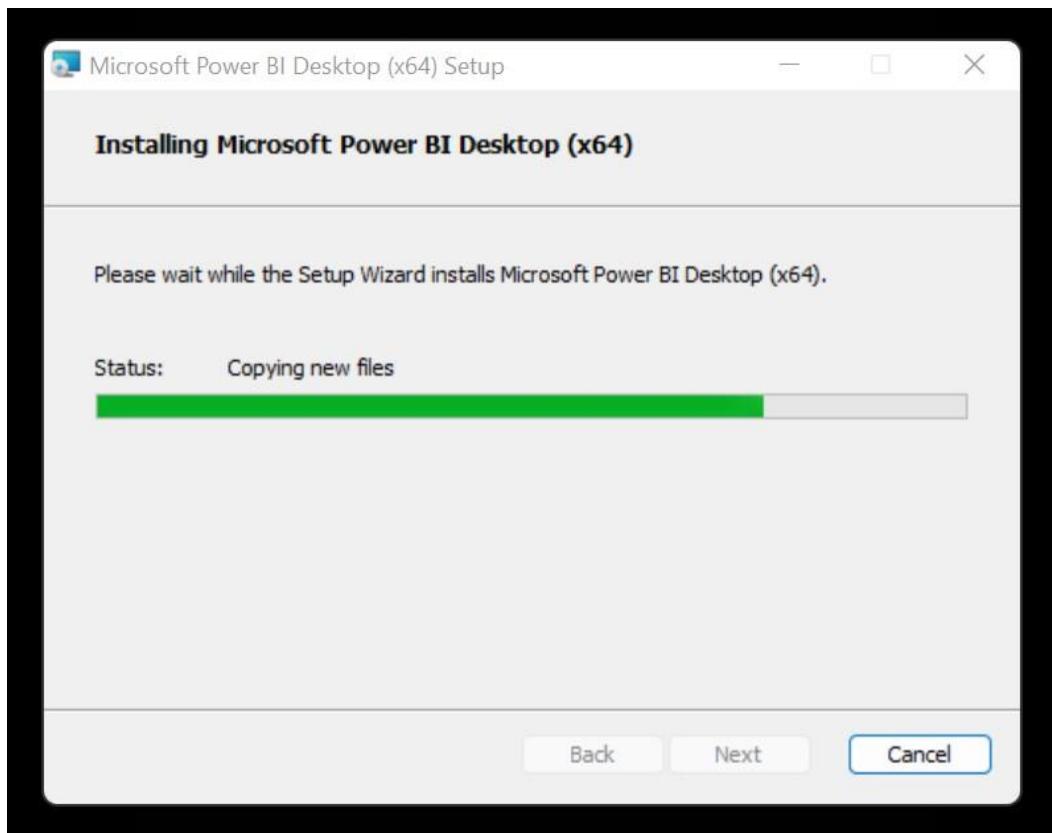
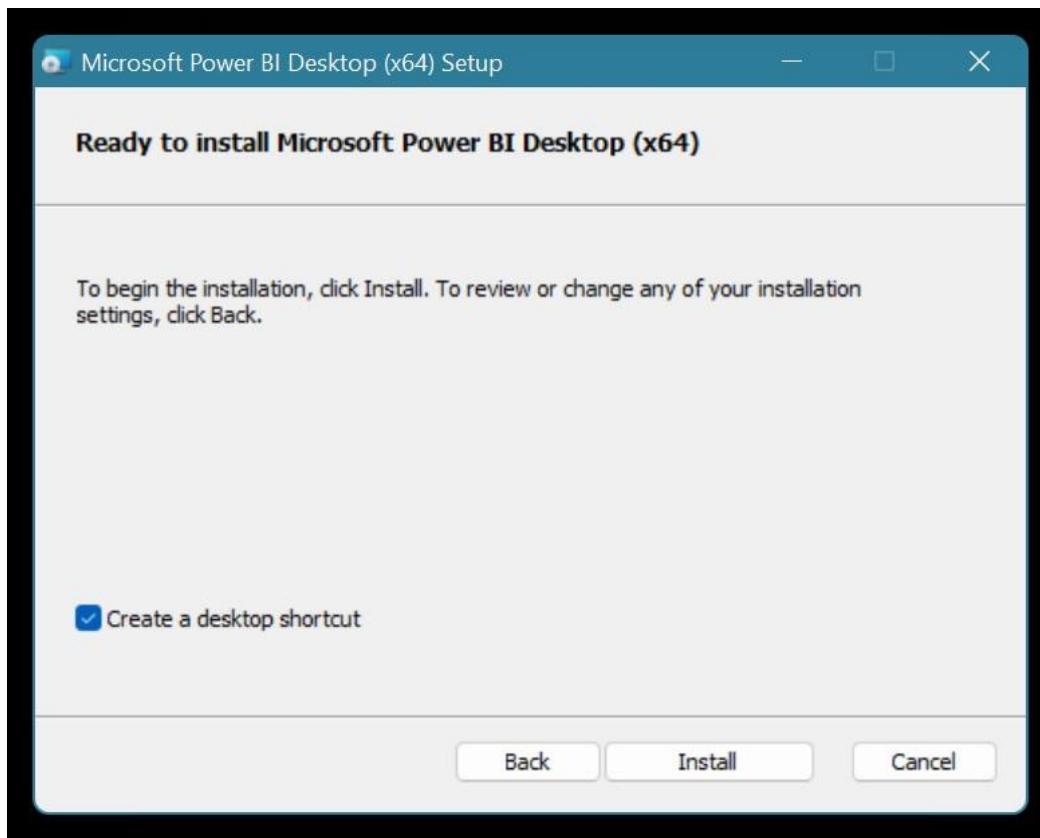
[Download Microsoft Power BI Desktop from Official Microsoft Download Center](https://www.microsoft.com/en-us/download/details.aspx?id=58494)

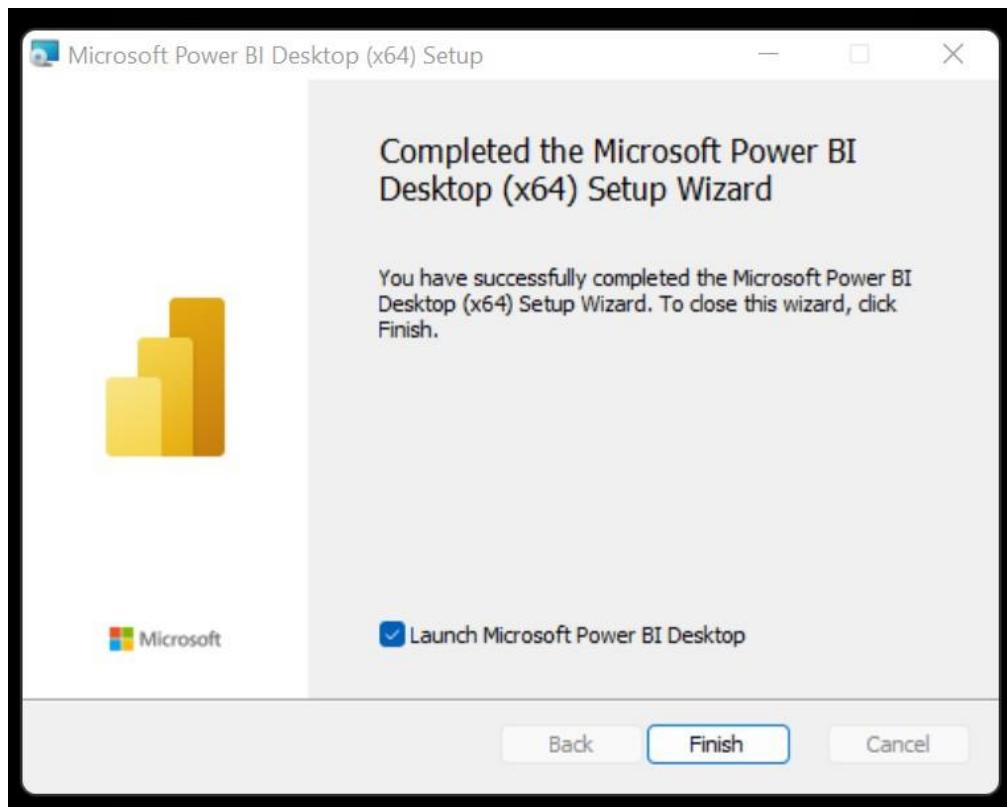
<https://www.microsoft.com/en-us/download/details.aspx?id=58494>







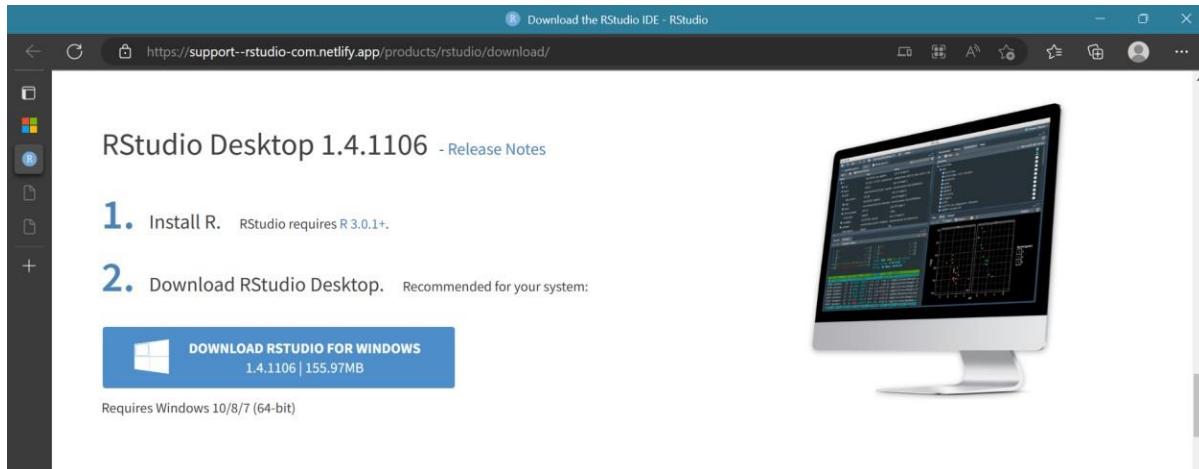




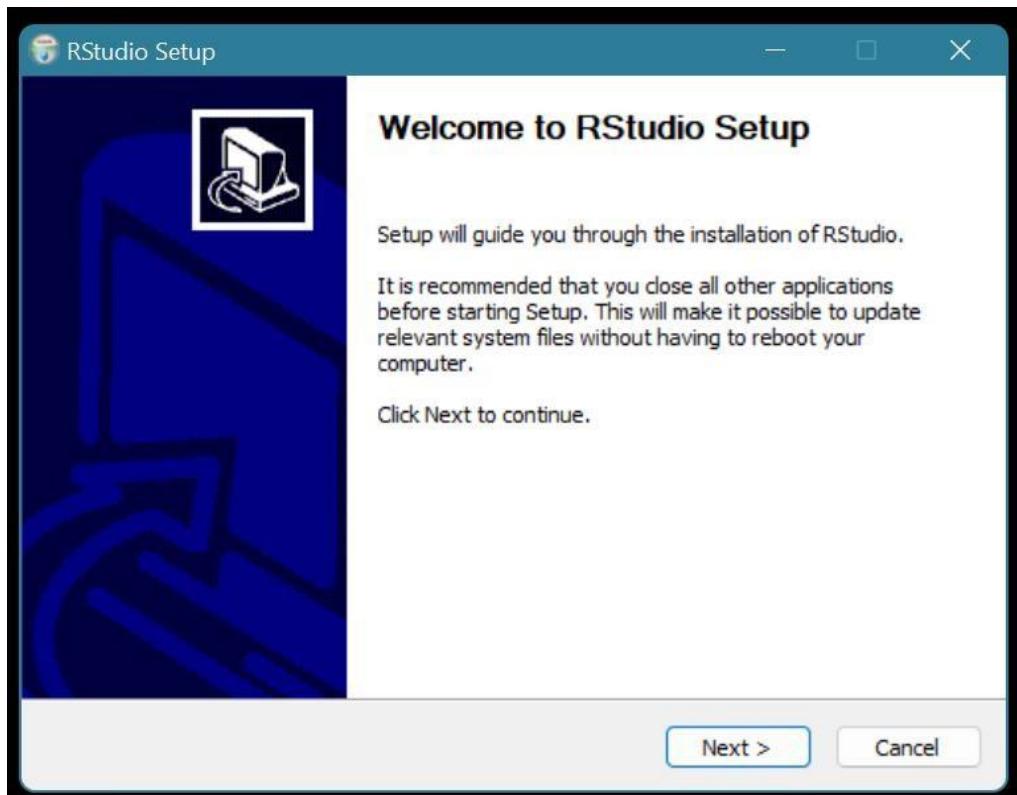
B) R Studio Software Setup and Installation:

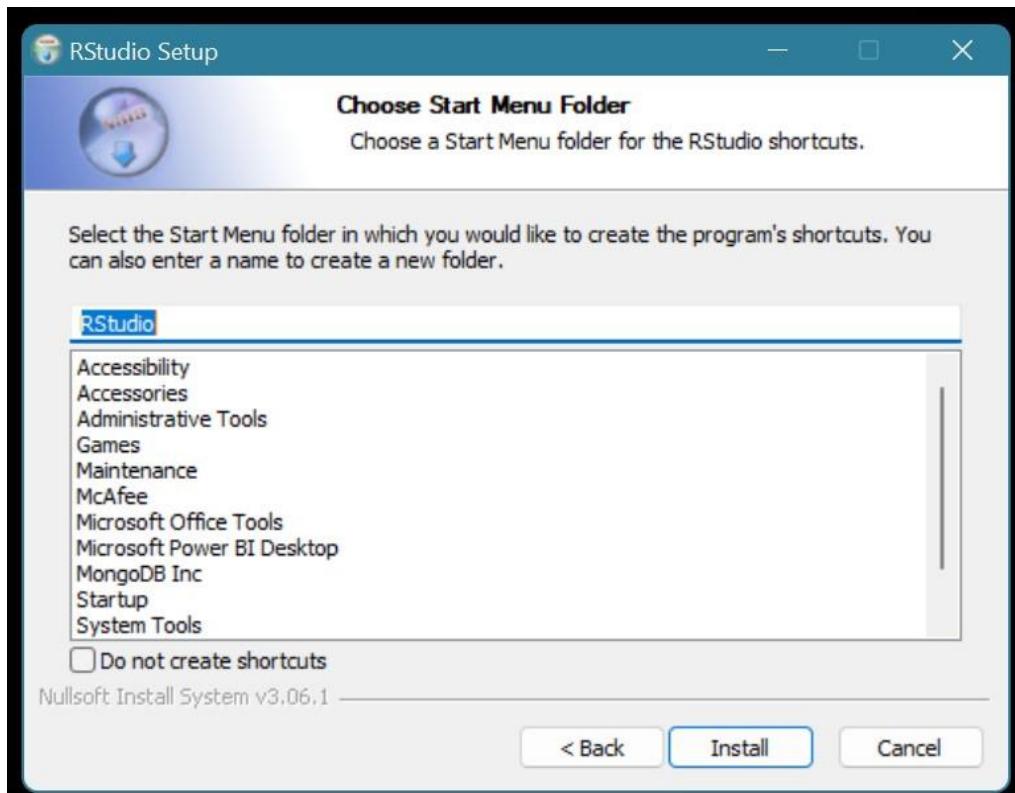
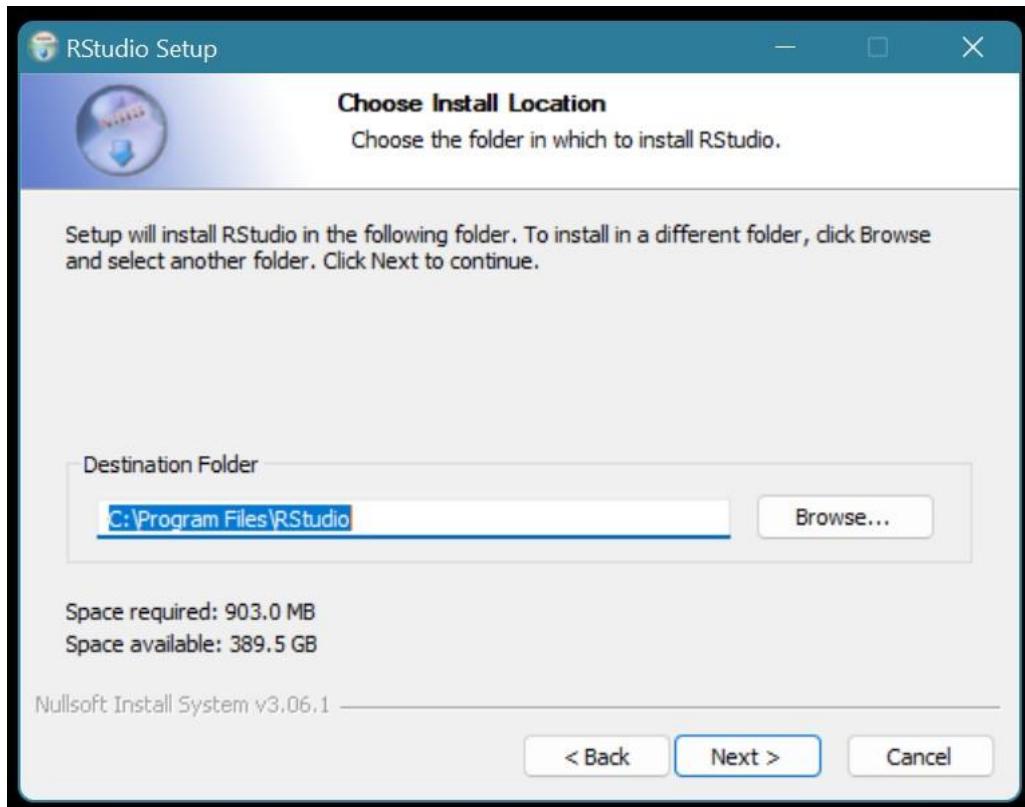
Download R Studio software by clicking on the link given below and follow the following screenshots for further setup steps:-

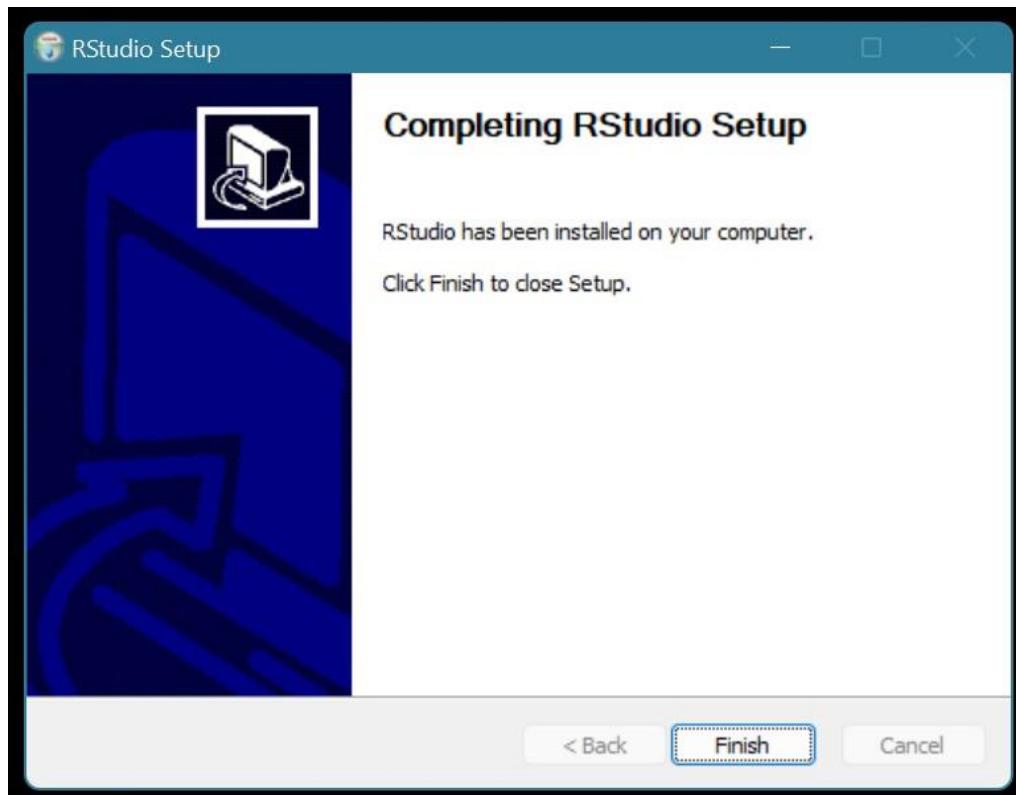
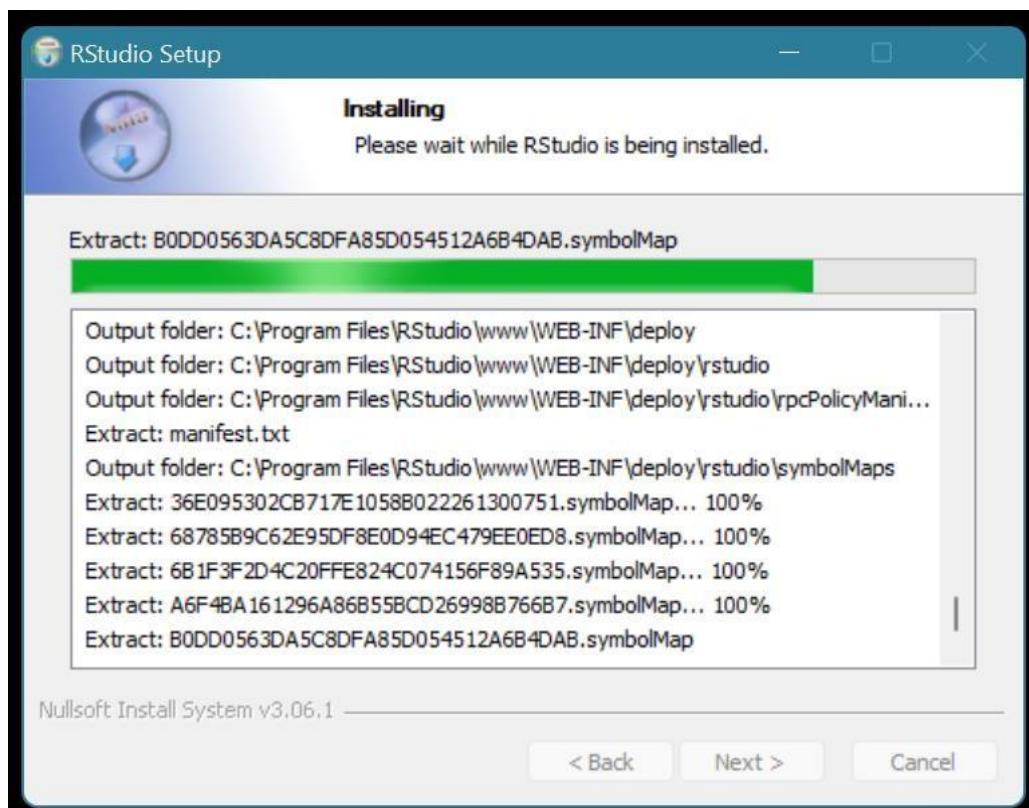
[Download the RStudio IDE - RStudio \(support--rstudio-com.netlify.app\)](https://support--rstudio-com.netlify.app/products/rstudio/download/)



Download







Practical 1

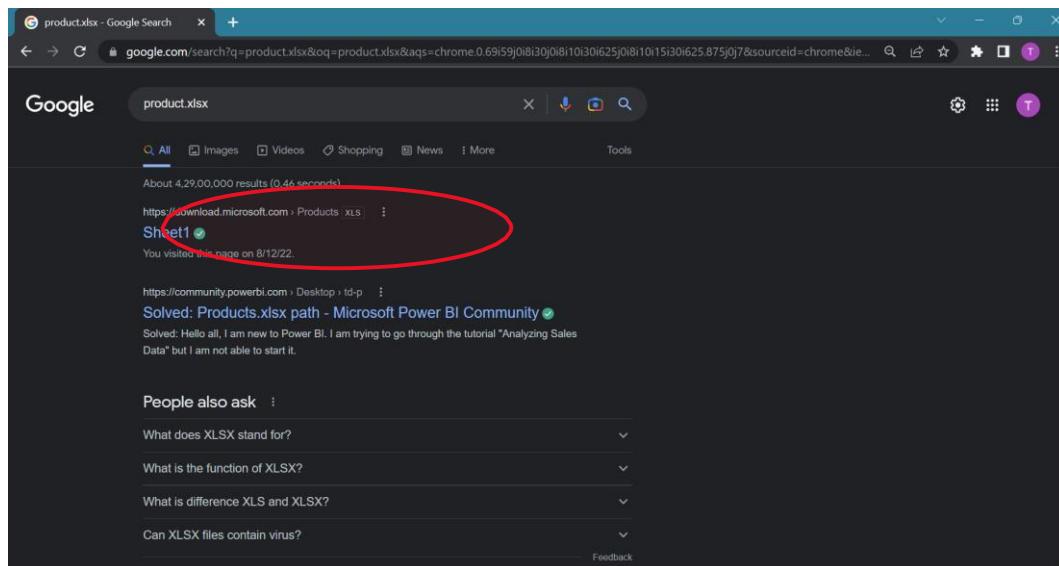
Import the legacy data from different sources such as (Excel, SQL Server, Oracle etc.) and load in the target system.

A) Importing Excel File :-

Step 1: Downloading excel file

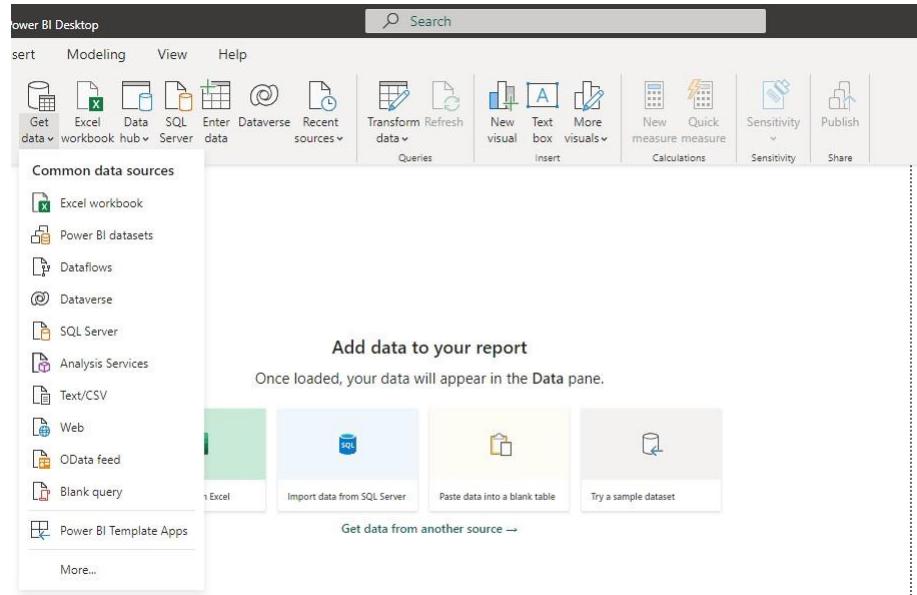
- To download excel file click on the link -
<https://www.google.com/search?q=product.xlsx&oq=product.xlsx&aqs=chrome..69i57j0i8i30j0i8i10i30i625j0i8i10i15i30i625.767j0j7&sourceid=chrome&ie=UTF-8>

- Click on Sheet 1 to download it.

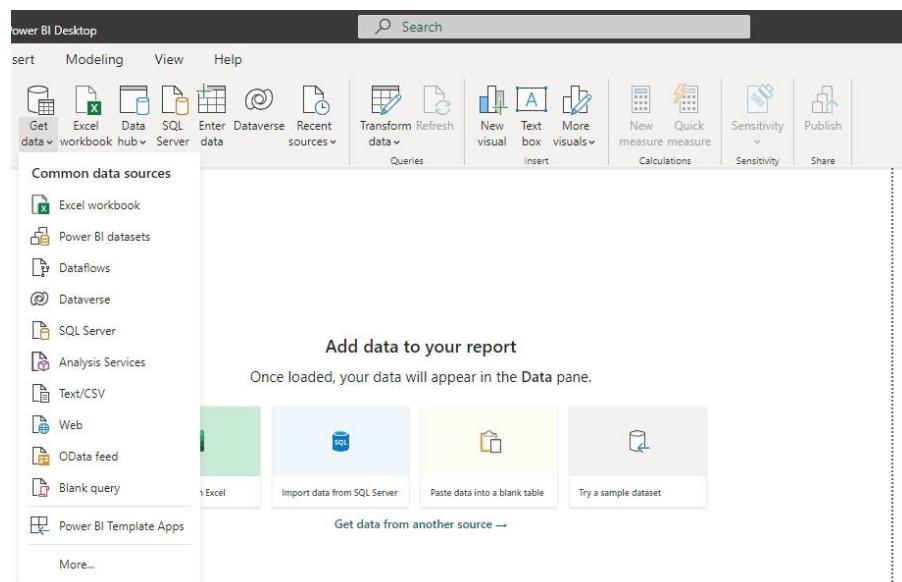


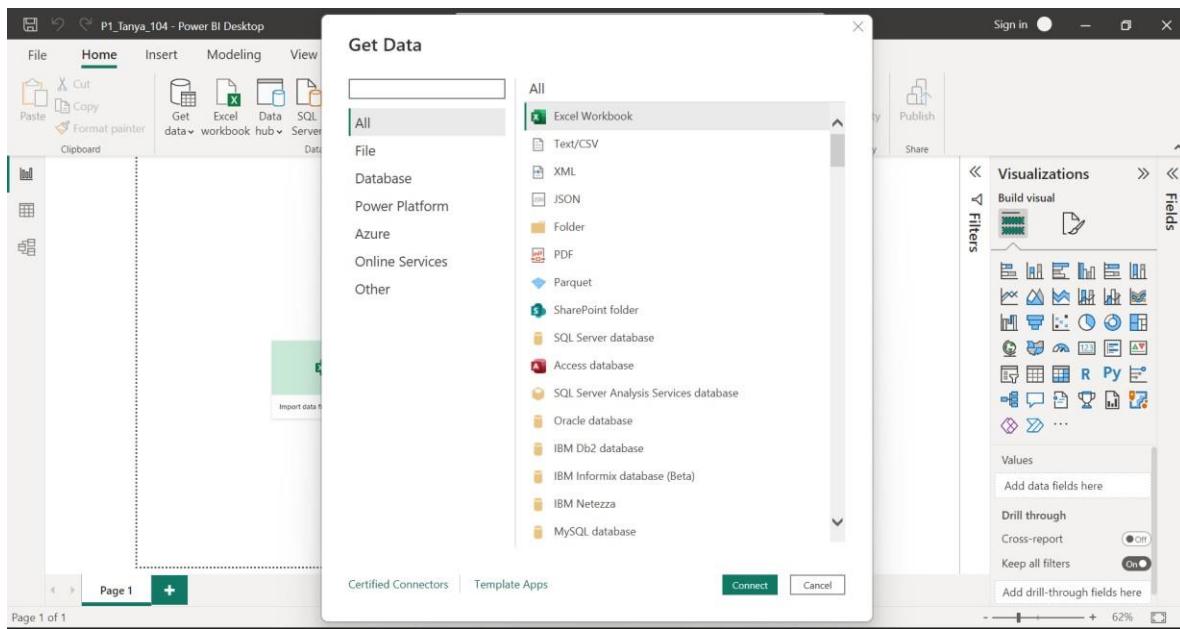
Step 2: Importing Excel file

- To import file, we can directly click on the “import data from excel” option.

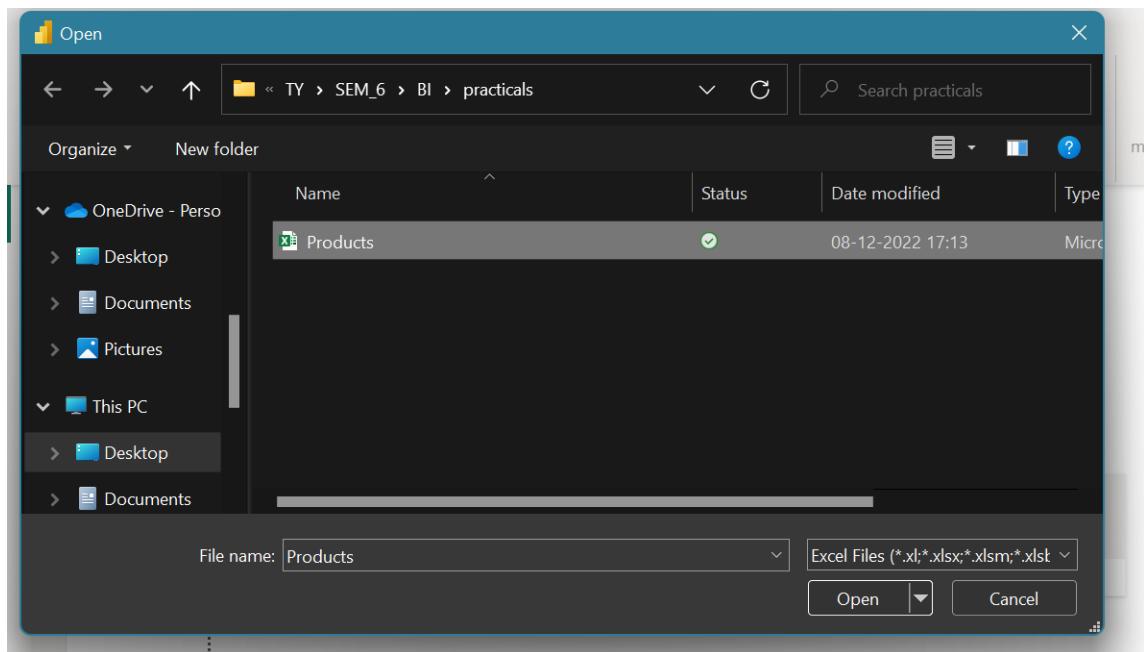


- Also, data can be imported by selecting – “Get data” from “HOME” ribbon : Go to “get data” → select “Excel Workbook” and click on “connect”.





- Select the downloaded file and click on “open”



- Select “products” and click on “load”

The screenshot shows the Power BI Navigator window. On the left, there's a file tree with 'Products (1).xlsx [2]' expanded, showing 'Products' and 'Sheet1'. The 'Products' item is selected and has a checkmark next to it. On the right, a preview of the 'Products' table is displayed with 22 rows of data. At the bottom, there are three buttons: 'Load' (highlighted in green), 'Transform Data', and 'Cancel'.

ProductID	ProductName	SupplierID	CategoryID	Quar
1	Chai	1	1	1
2	Chang	1	1	2
3	Aniseed Syrup	1	2	1
4	Chef Anton's Cajun Seasoning	2	2	4
5	Chef Anton's Gumbo Mix	2	2	3
6	Grandma's Boysenberry Spread	3	2	1
7	Uncle Bob's Organic Dried Pears	3	7	1
8	Northwoods Cranberry Sauce	3	2	1
9	Mishi Kobe Niku	4	6	1
10	Ikura	4	8	1
11	Queso Cabrales	5	4	1
12	Queso Manchego La Pastora	5	4	1
13	Konbu	6	8	2
14	Tofu	6	7	4
15	Genen Shouyu	6	2	2
16	Pavlova	7	3	3
17	Alice Mutton	7	6	2
18	Carnarvon Tigers	7	8	1
19	Teatime Chocolate Biscuits	8	3	1
20	Sir Rodney's Marmalade	8	3	3
21	Sir Rodney's Scones	8	3	2
22	Gustaf's Knäckebrot	9	5	2

- From left side, select “data” to display data in tabular format.

The screenshot shows the Power BI Data view. At the top, there's a ribbon with 'File', 'Home', 'Help', and 'Table tools'. Under 'Table tools', there are buttons for 'Name' (set to 'Products'), 'Mark as date table', 'Manage relationships', 'New measure', 'Quick measure', 'New column', 'New table', and 'Calculations'. Below the ribbon is a search bar and a dropdown menu set to 'Products'. The main area is a large grid table with columns: ProductID, ProductName, SupplierID, CategoryID, QuantityPerUnit, UnitPrice, UnitsInStock, UnitsOnOrder, ReorderLevel, and Discontinued. The table contains 77 rows of product data. A 'Data' tab is visible at the top right of the grid.

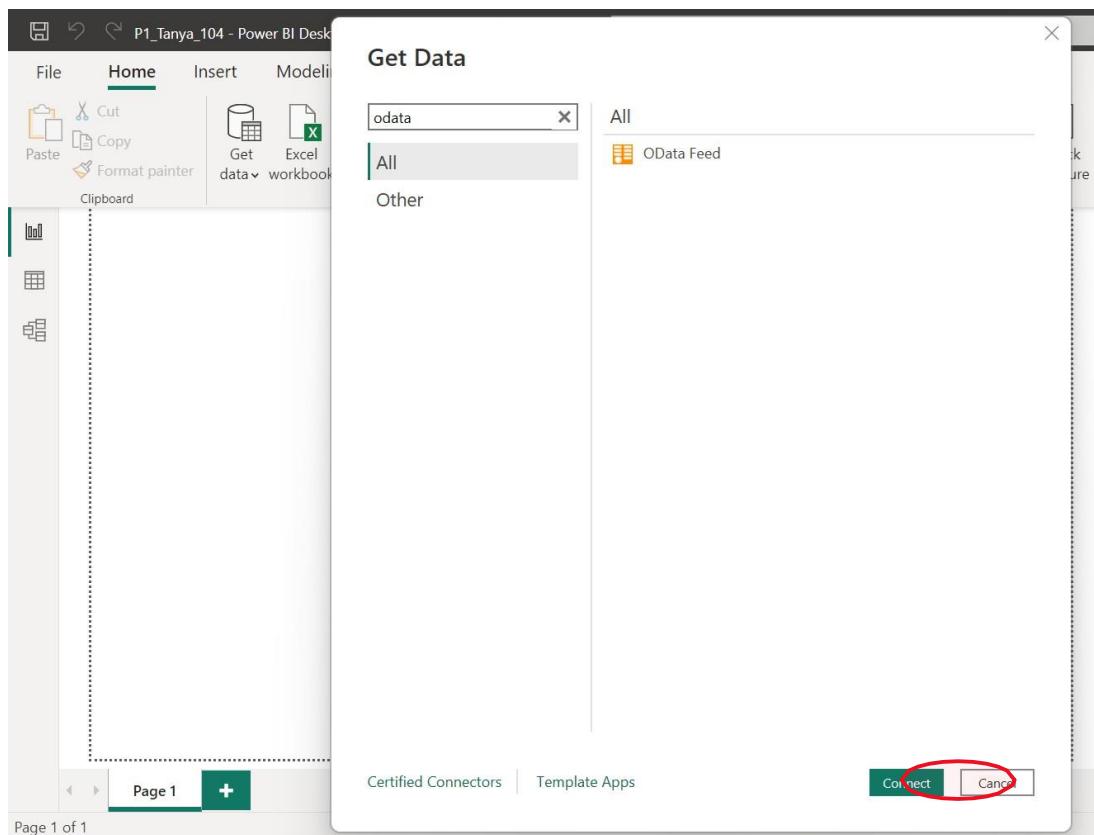
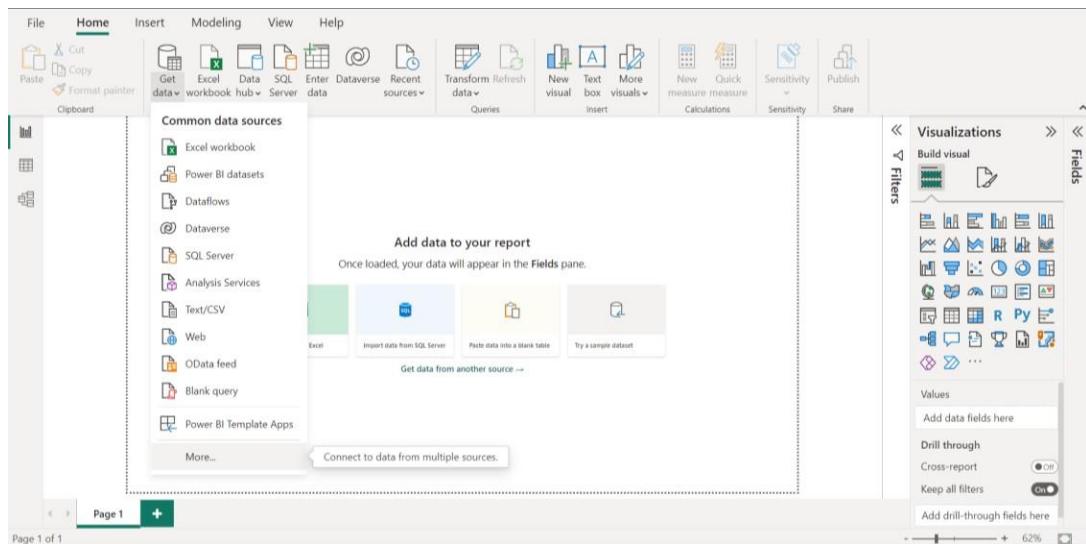
- Similarly, select “Model” to display in a mathematical format.

The screenshot shows the Microsoft Power BI desktop interface. The ribbon at the top is set to 'Home'. On the left, there's a navigation pane with 'Clipboard', 'Model' (which is selected), 'Products', 'CategoryID', 'Discontinued', 'ProductID', 'ProductName', 'QuantityPerUnit', 'ReorderLevel', 'SupplierID', 'UnitPrice', and 'UnitsInStock'. Below this is a 'Collapse' button. On the right, the 'Properties' pane is open, showing settings for 'Cards': 'Show the database in the header when applicable' (No), 'Show related fields when card is collapsed' (Yes), and 'Pin related fields to top of card' (No). The 'Fields' pane shows a search bar and a list containing 'Products'. At the bottom, there are buttons for 'All tables' and '+'. The status bar at the bottom right shows zoom levels from 100% to 150%.

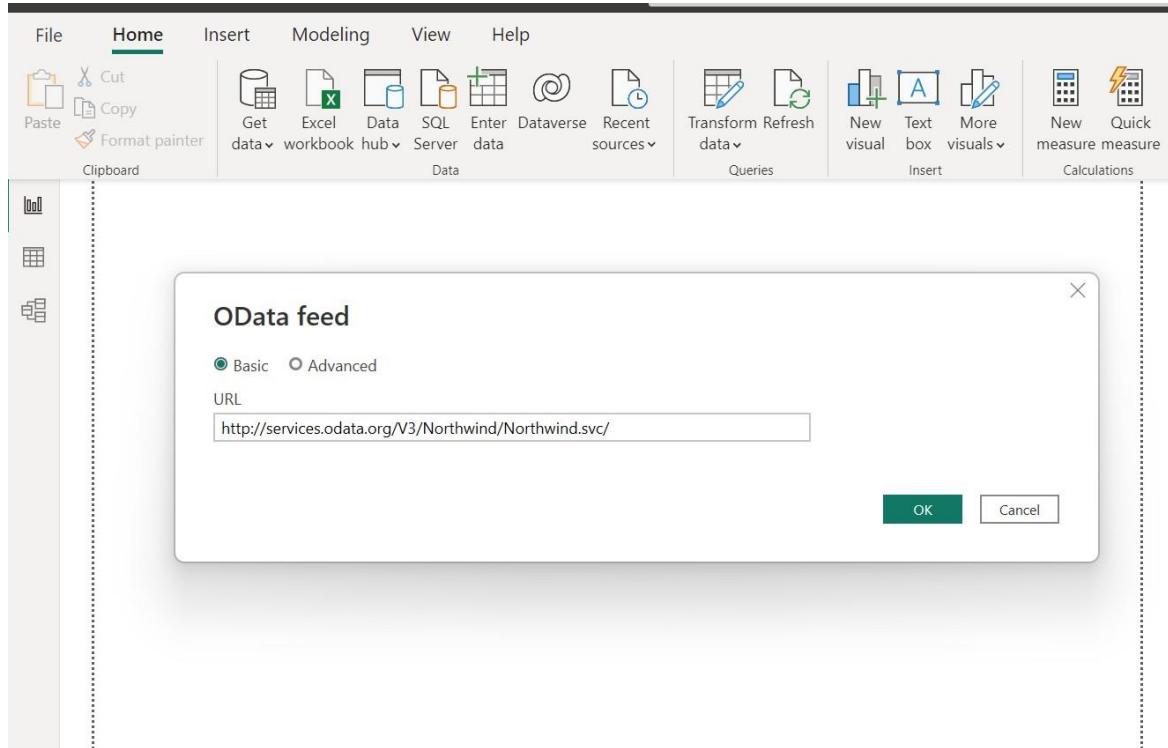
B) Importing O-data File :-

Step 1: Importing O-data file: O-data file is the file that has large amount of data. It is a type of big data.

- Data can be imported by selecting – get data from “HOME” ribbon :
Go to “get data” → select “OData feed” and click on “connect”.



- Now, enter the following URL "<http://services.odata.org/V3/Northwind/Northwind.svc/>" and click on "OK"



- Select "Orders" and click on "Load"

OrderID	CustomerID	EmployeeID	OrderDate	RequiredDate
10248	VINET		5	04-07-1996 00:00:00
10249	TOMSP		6	05-07-1996 00:00:00
10250	HANAR		4	08-07-1996 00:00:00
10251	VICTE		3	08-07-1996 00:00:00
10252	SUPRD		4	09-07-1996 00:00:00
10253	HANAR		3	10-07-1996 00:00:00
10254	CHOPS		5	11-07-1996 00:00:00
10255	RICSU		9	12-07-1996 00:00:00
10256	WELLI		3	15-07-1996 00:00:00
10257	HILAA		4	16-07-1996 00:00:00
10258	ERNSH		1	17-07-1996 00:00:00
10259	CENTC		4	18-07-1996 00:00:00
10260	OTTIK		4	19-07-1996 00:00:00
10261	QUEDE		4	19-07-1996 00:00:00
10262	RATTC		8	22-07-1996 00:00:00
10263	ERNSH		9	23-07-1996 00:00:00
10264	FOLKO		6	24-07-1996 00:00:00
10265	BLOTP		2	25-07-1996 00:00:00
10266	WARTH		3	26-07-1996 00:00:00
10267	FRANK		4	29-07-1996 00:00:00
10268	GROSR		8	30-07-1996 00:00:00

- From left side, select “data” to display data in tabular format.

The screenshot shows the Microsoft Power BI interface in 'Data' mode. The ribbon at the top has 'Table tools' selected. The main area displays a table titled 'Orders' with 830 rows. The columns include CustomerID, EmployeeID, OrderDate, RequiredDate, ShippedDate, ShipVia, Freight, ShipName, ShipAddress, ShipCity, and ShipCountry. The data shows various orders placed by different customers like TOMSP, OTTR, FRANK, etc., with details such as shipping via truck (ShipVia 1) or air (ShipVia 2), and freight amounts ranging from 11.61 to 249.06.

- Similarly, select “Model” to display in a mathematical format.

The screenshot shows the Microsoft Power BI interface in 'Model' mode. The ribbon at the top has 'Home' selected. The main area displays the 'Orders' table structure. On the left, a tree view shows fields like CustomerID, EmployeeID, Freight, OrderDate, OrderID, RequiredDate, ShipAddress, ShipCity, and ShipCountry. On the right, the 'Properties' pane is open, showing settings for 'Cards', 'Fields', and 'Sensitivity'. The 'Cards' section includes options for showing the database header and pinning related fields. The 'Fields' section includes options for showing related fields when collapsed and pinning them to the top of the card.

- Since, we have loaded the data under same files, we can view it simultaneously.

The screenshot shows the Microsoft Power BI Data View interface. At the top, there's a ribbon with File, Home, and Help tabs, followed by various icons for clipboard operations, data sources (Get data from Excel, Data, SQL, Enter data, Dataverse, Server), queries (Transform, Refresh, Manage relationships, New measure column, New table, Security), Q&A, Language setup, Sensitivity, and Publish. Below the ribbon, there are two tables displayed: 'Products' and 'Orders'. The 'Products' table contains fields like CategoryID, Discontinued, ProductID, ProductName, QuantityPerUnit, ReorderLevel, SupplierID, UnitPrice, UnitsInStock, and UnitsOnOrder. The 'Orders' table contains fields like CustomerID, EmployeeID, Freight, OrderDate, OrderID, RequiredDate, ShipAddress, ShipCity, ShipCountry, ShipName, ShippedDate, ShipPostalCode, ShipRegion, and ShipVia. On the right side of the screen, a 'Properties' pane is open, showing settings for the current card. It includes options for 'Cards' (Show the database in the header when applicable, with 'No' selected), 'Related fields' (Show related fields when card is collapsed, with 'Yes' selected), and 'Pin related fields' (Pin related fields to top of card, with 'No' selected). There's also a 'Fields' section with a search bar and links to 'Orders' and 'Products'. At the bottom left, there are buttons for 'All tables' and a '+' sign. The bottom right corner shows a zoom level of 100%.

Practical 2

ETL Process

- A) Removing Columns**
- B) Changing Data types**
- C) Expanding the Order_Details table**
- D) Calculating line total**
- E) Renaming and reordering columns**
- F) Combine the Products and Total Sales**

A) Removing other columns to only display columns of interest:-

In this step, we will remove all columns except ProductID, ProductName, UnitsInStock and QuantityPerUnit.

Power BI Desktop includes Query Editor, which is where you shape and transform your data connections. Query Editor opens automatically when you select Edit from Navigator. You can also open the Query Editor by selecting Edit Queries from the Home ribbon in Power BI Desktop.

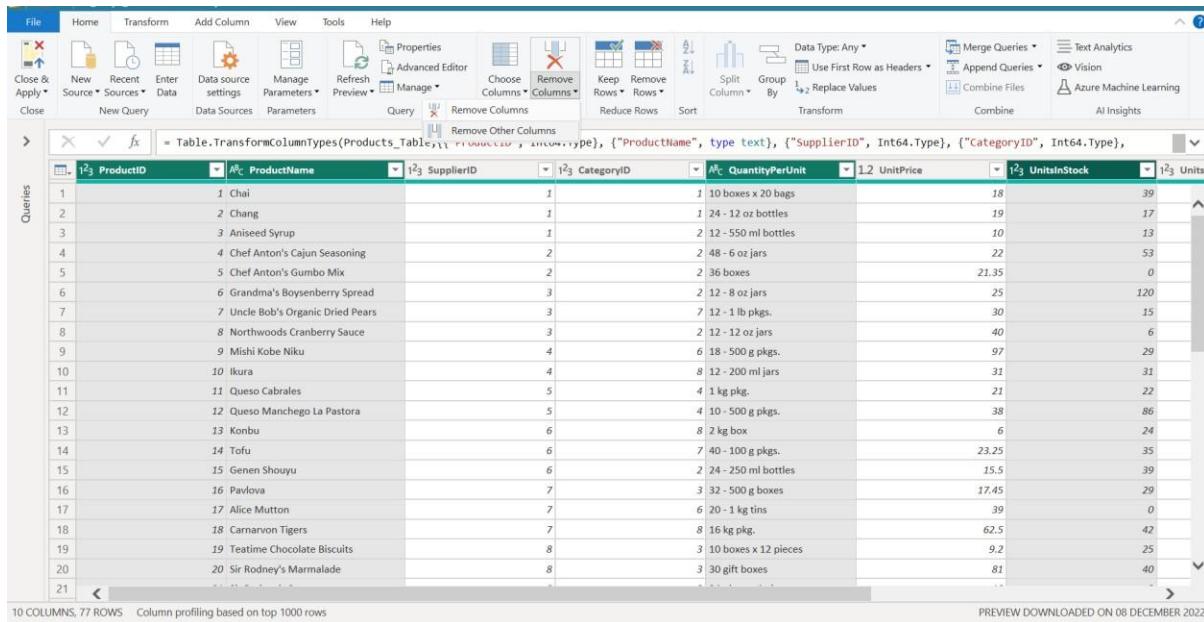
Step 1:

- Select “Transform data” from “Table tools” ribbon.

ProductID	ProductName	SupplierID	CategoryID	QuantityPerUnit	UnitPrice	UnitsInStock	UnitsOnOrder	ReorderLevel	Discontinued
1	Chai	1	1	10 boxes x 20 bags	18	39	0	10	False
2	Chang	1	1	24 - 12 oz bottles	19	17	40	25	False
3	Aniseed Syrup	1	2	12 - 550 ml bottles	10	13	70	25	False
4	Chef Anton's Cajun Seasoning	2	2	48 - 6 oz jars	22	53	0	0	False
5	Chef Anton's Gumbo Mix	2	2	36 boxes	21.35	0	0	0	True
6	Grandma's Boysenberry Spread	3	2	12 - 8 oz jars	25	120	0	25	False
7	Uncle Bob's Organic Dried Pears	3	7	12 - 1 lb pkgs.	30	15	0	10	False
8	Northwoods Cranberry Sauce	3	2	12 - 12 oz jars	40	6	0	0	False
9	Mishi Kobe Niku	4	6	18 - 500 g pkgs.	97	29	0	0	True
10	Ikura	4	8	12 - 200 ml jars	31	31	0	0	False
11	Queso Cabrales	5	4	1 kg pkg.	21	22	30	30	False
12	Queso Manchego La Pastora	5	4	10 - 500 g pkgs.	38	86	0	0	False
13	Konbu	6	8	2 kg box	6	24	0	5	False
14	Tofu	6	7	40 - 100 g pkgs.	23.25	35	0	0	False
15	Genen Shouyu	6	2	24 - 250 ml bottles	15.5	39	0	5	False
16	Pavlova	7	3	32 - 500 g boxes	17.45	29	0	10	False
17	Alice Mutton	7	6	20 - 1 kg tins	39	0	0	0	True
18	Carnarvon Tigers	7	8	16 kg pkg.	62.5	42	0	0	False
19	Teatime Chocolate Biscuits	8	3	10 boxes x 12 pieces	9.2	25	0	5	False
20	Sir Rodney's Marmalade	8	3	30 gift boxes	81	40	0	0	False
21	Moreton Bay Rum	9	9	12 - 1.5 liter cans	12	12	0	10	False
22	Alfredo's Spaghetti	9	10	16 oz jars	31	31	0	15	False
23	Watterson Natural Sea Salt	10	10	1 kg tubs	40	40	0	20	False
24	Quince Paste	11	11	12 - 100 g jars	30	30	0	15	False
25	Green Chile Pepper Paste	11	11	12 - 100 g jars	30	30	0	15	False
26	Smoked Sardines in Oil	12	12	12 - 100 g cans	30	30	0	15	False
27	Preserved Lemons	12	12	12 - 100 g cans	30	30	0	15	False
28	Ginger Paste	13	13	12 - 100 g jars	30	30	0	15	False
29	Badia Ground Cumin	13	13	12 - 100 g jars	30	30	0	15	False
30	Badia Ground Coriander	13	13	12 - 100 g jars	30	30	0	15	False
31	Badia Ground Turmeric	13	13	12 - 100 g jars	30	30	0	15	False
32	Badia Ground Cinnamon	13	13	12 - 100 g jars	30	30	0	15	False
33	Badia Ground Cloves	13	13	12 - 100 g jars	30	30	0	15	False
34	Badia Ground Nutmeg	13	13	12 - 100 g jars	30	30	0	15	False
35	Badia Ground Allspice	13	13	12 - 100 g jars	30	30	0	15	False
36	Badia Ground Black Pepper	13	13	12 - 100 g jars	30	30	0	15	False
37	Badia Ground White Pepper	13	13	12 - 100 g jars	30	30	0	15	False
38	Badia Ground Star Anise	13	13	12 - 100 g jars	30	30	0	15	False
39	Badia Ground Cardamom	13	13	12 - 100 g jars	30	30	0	15	False
40	Badia Ground Cinnamon Leaf	13	13	12 - 100 g jars	30	30	0	15	False
41	Badia Ground Cinnamon Bark	13	13	12 - 100 g jars	30	30	0	15	False
42	Badia Ground Cinnamon Sticks	13	13	12 - 100 g jars	30	30	0	15	False
43	Badia Ground Cinnamon Powder	13	13	12 - 100 g jars	30	30	0	15	False
44	Badia Ground Cinnamon Sticks	13	13	12 - 100 g jars	30	30	0	15	False
45	Badia Ground Cinnamon Powder	13	13	12 - 100 g jars	30	30	0	15	False
46	Badia Ground Cinnamon Sticks	13	13	12 - 100 g jars	30	30	0	15	False
47	Badia Ground Cinnamon Powder	13	13	12 - 100 g jars	30	30	0	15	False
48	Badia Ground Cinnamon Sticks	13	13	12 - 100 g jars	30	30	0	15	False
49	Badia Ground Cinnamon Powder	13	13	12 - 100 g jars	30	30	0	15	False
50	Badia Ground Cinnamon Sticks	13	13	12 - 100 g jars	30	30	0	15	False
51	Badia Ground Cinnamon Powder	13	13	12 - 100 g jars	30	30	0	15	False
52	Badia Ground Cinnamon Sticks	13	13	12 - 100 g jars	30	30	0	15	False
53	Badia Ground Cinnamon Powder	13	13	12 - 100 g jars	30	30	0	15	False
54	Badia Ground Cinnamon Sticks	13	13	12 - 100 g jars	30	30	0	15	False
55	Badia Ground Cinnamon Powder	13	13	12 - 100 g jars	30	30	0	15	False
56	Badia Ground Cinnamon Sticks	13	13	12 - 100 g jars	30	30	0	15	False
57	Badia Ground Cinnamon Powder	13	13	12 - 100 g jars	30	30	0	15	False
58	Badia Ground Cinnamon Sticks	13	13	12 - 100 g jars	30	30	0	15	False
59	Badia Ground Cinnamon Powder	13	13	12 - 100 g jars	30	30	0	15	False
60	Badia Ground Cinnamon Sticks	13	13	12 - 100 g jars	30	30	0	15	False
61	Badia Ground Cinnamon Powder	13	13	12 - 100 g jars	30	30	0	15	False
62	Badia Ground Cinnamon Sticks	13	13	12 - 100 g jars	30	30	0	15	False
63	Badia Ground Cinnamon Powder	13	13	12 - 100 g jars	30	30	0	15	False
64	Badia Ground Cinnamon Sticks	13	13	12 - 100 g jars	30	30	0	15	False
65	Badia Ground Cinnamon Powder	13	13	12 - 100 g jars	30	30	0	15	False
66	Badia Ground Cinnamon Sticks	13	13	12 - 100 g jars	30	30	0	15	False
67	Badia Ground Cinnamon Powder	13	13	12 - 100 g jars	30	30	0	15	False
68	Badia Ground Cinnamon Sticks	13	13	12 - 100 g jars	30	30	0	15	False
69	Badia Ground Cinnamon Powder	13	13	12 - 100 g jars	30	30	0	15	False
70	Badia Ground Cinnamon Sticks	13	13	12 - 100 g jars	30	30	0	15	False
71	Badia Ground Cinnamon Powder	13	13	12 - 100 g jars	30	30	0	15	False
72	Badia Ground Cinnamon Sticks	13	13	12 - 100 g jars	30	30	0	15	False
73	Badia Ground Cinnamon Powder	13	13	12 - 100 g jars	30	30	0	15	False
74	Badia Ground Cinnamon Sticks	13	13	12 - 100 g jars	30	30	0	15	False
75	Badia Ground Cinnamon Powder	13	13	12 - 100 g jars	30	30	0	15	False
76	Badia Ground Cinnamon Sticks	13	13	12 - 100 g jars	30	30	0	15	False
77	Badia Ground Cinnamon Powder	13	13	12 - 100 g jars	30	30	0	15	False

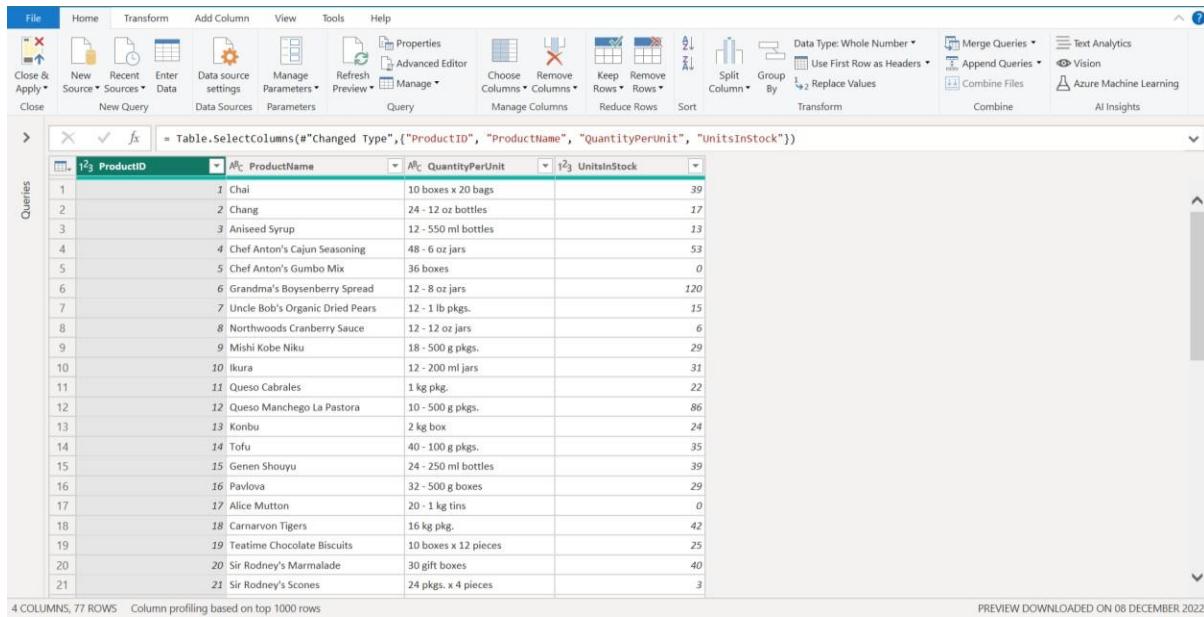
Step 2:

- Press “CTRL” key and select the columns of interest (ProductID, ProductName, QuantityPerUnit, UnitInStock)
- From the “Home” ribbon → Select “Remove Columns” → Click on “Remove Other Columns”



The screenshot shows the Microsoft Power BI desktop interface. The ribbon is visible at the top with the "Home" tab selected. A table preview is displayed below the ribbon, showing the first 21 rows of the "Products" table. The columns are labeled: ProductID, ProductName, SupplierID, CategoryID, QuantityPerUnit, UnitPrice, and UnitsInStock. The "QuantityPerUnit" column contains various unit descriptions like "10 boxes x 20 bags" and "12 - 550 ml bottles". The "UnitPrice" column shows values like 18, 19, 10, 22, etc. The "UnitsInStock" column shows values like 39, 17, 13, 53, etc. The status bar at the bottom indicates "10 COLUMNS, 77 ROWS" and "Column profiling based on top 1000 rows".

- This is the final display



The screenshot shows the Microsoft Power BI desktop interface with the same table preview as the previous image. However, the table now only displays four columns: ProductID, ProductName, QuantityPerUnit, and UnitsInStock. The other columns have been removed. The data remains the same, showing the reduced set of columns for each product row. The status bar at the bottom indicates "4 COLUMNS, 77 ROWS" and "Column profiling based on top 1000 rows".

B) Change the data type of the UnitInStock column:

In this step, we will confirm the UnitInStock column's datatype is Whole Number.

When Query Editor connects to data, it reviews each field and to determine the best data type. For the Excel workbook, products in stock will always be a whole number.

Step 1:

- Select “UnitInStock” column
- Select “Whole Number” from the “Data Type” panel which is in “Transform” ribbon

The screenshot shows the Microsoft Power BI Query Editor interface. The 'Transform' ribbon is open, and the 'Data Type' dropdown is set to 'Whole Number'. The table preview shows the first few rows of data, including ProductID, ProductName, QuantityPerUnit, and UnitInStock. The 'UnitInStock' column is currently typed as 'Int64'. The table has 21 rows in total.

ProductID	ProductName	QuantityPerUnit	UnitInStock
1	Chai	10 boxes x 20 bags	39
2	Chang	24 - 12 oz bottles	17
3	Aniseed Syrup	12 - 550 ml bottles	13
4	Chef Anton's Cajun Seasoning	48 - 6 oz jars	53
5	Chef Anton's Gumbo Mix	36 boxes	0
6	Grandma's Boysenberry Spread	12 - 8 oz jars	220
7	Uncle Bob's Organic Dried Pears	12 - 1 lb pkgs.	15
8	Northwoods Cranberry Sauce	12 - 12 oz jars	6
9	Mishi Kobe Niku	18 - 500 g pkgs.	29
10	Ikura	12 - 200 ml jars	31
11	Queso Cabrales	1 kg pkgs.	22
12	Queso Manchego La Pastora	10 - 500 g pkgs.	86
13	Konbu	2 kg box	24
14	Tofu	40 - 100 g pkgs.	35
15	Genen Shouyu	24 - 250 ml bottles	39
16	Pavlova	32 - 500 g boxes	29
17	Alice Mutton	20 - 1 kg tins	0
18	Carnarvon Tigers	16 kg pkgs.	42
19	Teatime Chocolate Biscuits	10 boxes x 12 pieces	25
20	Sir Rodney's Marmalade	30 gift boxes	40
21	Sir Rodney's Scones	24 pkgs. x 4 pieces	3

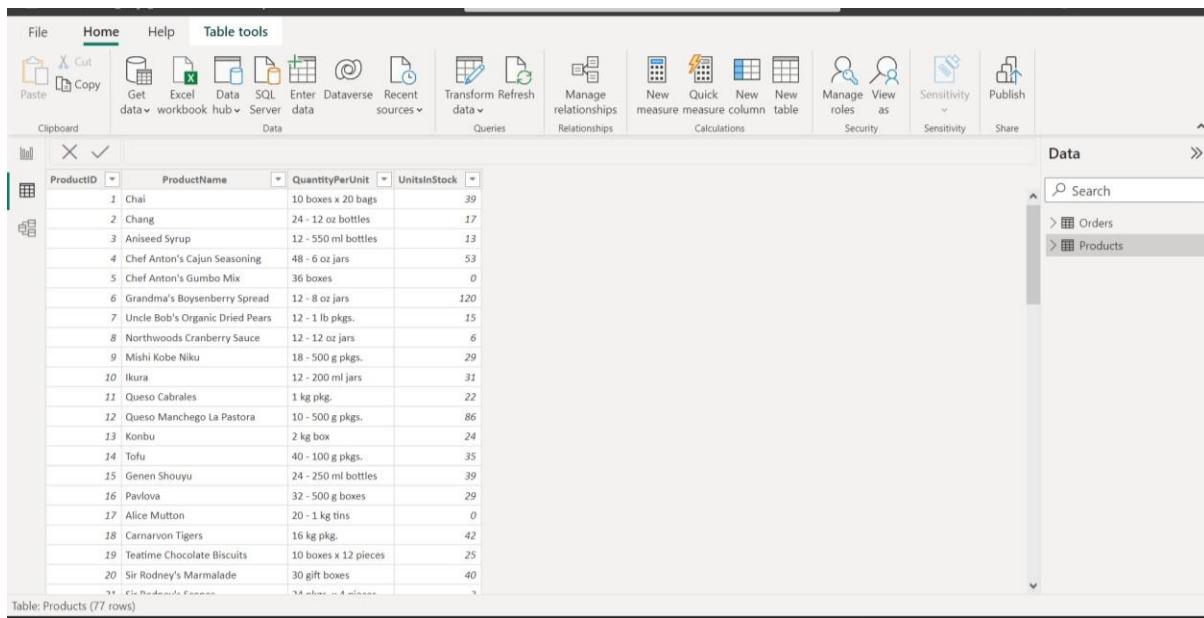
Step 2:

- Applying the necessity changes by selecting “Close & Apply” option from “Home” ribbon.

The screenshot shows the Microsoft Power BI Query Editor interface with the 'Home' ribbon selected. The 'Close & Apply' button is highlighted. The table preview shows the same data as before, but the 'UnitInStock' column is now correctly typed as 'Whole Number'. The table has 21 rows in total.

ProductID	ProductName	QuantityPerUnit	UnitInStock
1	Chai	10 boxes x 20 bags	39
2	Chang	24 - 12 oz bottles	17
3	Aniseed Syrup	12 - 550 ml bottles	13
4	Chef Anton's Cajun Seasoning	48 - 6 oz jars	53
5	Chef Anton's Gumbo Mix	36 boxes	0
6	Grandma's Boysenberry Spread	12 - 8 oz jars	220
7	Uncle Bob's Organic Dried Pears	12 - 1 lb pkgs.	15
8	Northwoods Cranberry Sauce	12 - 12 oz jars	6
9	Mishi Kobe Niku	18 - 500 g pkgs.	29
10	Ikura	12 - 200 ml jars	31
11	Queso Cabrales	1 kg pkgs.	22
12	Queso Manchego La Pastora	10 - 500 g pkgs.	86
13	Konbu	2 kg box	24
14	Tofu	40 - 100 g pkgs.	35
15	Genen Shouyu	24 - 250 ml bottles	39
16	Pavlova	32 - 500 g boxes	29
17	Alice Mutton	20 - 1 kg tins	0
18	Carnarvon Tigers	16 kg pkgs.	42
19	Teatime Chocolate Biscuits	10 boxes x 12 pieces	25
20	Sir Rodney's Marmalade	30 gift boxes	40
21	Sir Rodney's Scones	24 pkgs. x 4 pieces	3

This is the updated list.



The screenshot shows the Microsoft Power BI Data Editor interface. The top navigation bar includes File, Home, Help, and Table tools tabs. Under the Home tab, there are sections for Clipboard (Paste, Cut, Copy), Data (Get data from workbook hub, Data, SQL Server, Enter data, Dataverse, Recent sources), Queries (Transform data, Refresh data), Relationships (Manage relationships), Calculations (New measure, Quick measure, New column, New table), Security (Manage roles, View as), Sensitivity (Sensitivity), and Share (Publish). A search bar labeled "Search" is located in the top right. On the left, there's a sidebar with a tree view showing "Orders" and "Products". The main area displays a table titled "Table: Products (77 rows)". The columns are ProductID, ProductName, QuantityPerUnit, and UnitsInStock. The data includes various products like Chai, Chang, Aniseed Syrup, Chef Anton's Cajun Seasoning, etc., with their respective details.

ProductID	ProductName	QuantityPerUnit	UnitsInStock
1	Chai	10 boxes x 20 bags	39
2	Chang	24 - 12 oz bottles	17
3	Aniseed Syrup	12 - 550 ml bottles	13
4	Chef Anton's Cajun Seasoning	48 - 6 oz jars	53
5	Chef Anton's Gumbo Mix	36 boxes	0
6	Grandma's Boysenberry Spread	12 - 8 oz jars	120
7	Uncle Bob's Organic Dried Pears	12 - 1 lb pkgs.	15
8	Northwoods Cranberry Sauce	12 - 12 oz jars	6
9	Mishi Kobe Niku	18 - 500 g pkgs.	29
10	Ikura	12 - 200 ml jars	31
11	Queso Cabrales	1 kg pkg.	22
12	Queso Manchego La Pastora	10 - 500 g pkgs.	86
13	Konbu	2 kg box	24
14	Tofu	40 - 100 g pkgs.	35
15	Genen Shouyu	24 - 250 ml bottles	39
16	Pavlova	32 - 500 g boxes	29
17	Alice Mutton	20 - 1 kg tins	0
18	Carnarvon Tigers	16 kg pkg.	42
19	Teatime Chocolate Biscuits	10 boxes x 12 pieces	25
20	Sir Rodney's Marmalade	30 gift boxes	40

C) Expand the Order_Details table

In this step, you expand the Order_Details table that is related to the Orders table, to combine the ProductID, UnitPrice, and Quantity columns from Order_Details into the Orders table.

The Expand operation combines columns from a related table into a subject table. When the query runs, rows from the related table (Order_Details) are combined into rows from the subject table (Orders).

After you expand the Order_Details table, three new columns and additional rows are added to the Orders table, one for each row in the nested or related table.

The Orders table contains a reference to a Details table, which contains the individual products that were included in each Order. When you connect to data sources with multiple tables (such as a relational database) you can use these references to build up your query.

Step 1:

- From the “Home” ribbon, select “Transform data” option

OrderID	CustomerID	EmployeeID	OrderDate	RequiredDate	ShippedDate	ShipVia	Freight
1	1	1	1996-07-10 00:00:00	1996-08-16 00:00:00	1996-08-27 00:00:00	1	11.00
2	2	1	1996-07-10 00:00:00	1996-08-16 00:00:00	1996-08-26 00:00:00	1	55.40
3	3	1	1996-07-10 00:00:00	1996-08-16 00:00:00	1996-08-06 00:00:00	1	208.00
4	4	3	1996-07-10 00:00:00	1996-08-02 00:00:00	1996-08-12 00:00:00	3	76.07
5	5	3	1996-07-10 00:00:00	1996-06-06 00:00:00	1996-07-13 00:00:00	3	125.77
6	6	2	1996-07-10 00:00:00	1996-09-06 00:00:00	1996-08-16 00:00:00	2	25.83
7	7	1	1996-07-10 00:00:00	1996-09-06 00:00:00	1996-08-27 00:00:00	1	76.56
8	8	2	1996-07-10 00:00:00	1996-09-17 00:00:00	1996-08-26 00:00:00	2	76.83
9	9	3	1996-07-10 00:00:00	1996-08-18 00:00:00	1996-08-30 00:00:00	3	229.24
10	10	2	1996-07-10 00:00:00	1996-07-07 00:00:00	1996-07-17 00:00:00	2	45.08
11	11	2	1996-07-10 00:00:00	1996-07-21 00:00:00	1996-08-03 00:00:00	2	40.26
12	12	2	1996-07-10 00:00:00	1996-07-22 00:00:00	1996-08-04 00:00:00	2	1.96
13	13	1	1996-07-10 00:00:00	1996-04-11 00:00:00	1996-10-14 00:00:00	1	4.88
14	14	3	1996-07-10 00:00:00	1996-03-23 00:00:00	1996-10-14 00:00:00	3	64.86
15	15	3	1996-07-10 00:00:00	1996-02-21 00:00:00	1996-10-29 00:00:00	3	108.26
16	16	2	1996-07-10 00:00:00	1996-03-13 00:00:00	1996-10-04 00:00:00	2	54.83
17	17	1	1996-07-10 00:00:00	1996-02-28 00:00:00	1996-10-06 00:00:00	1	110.37
18	18	2	1996-07-10 00:00:00	1996-02-01 00:00:00	1996-10-11 00:00:00	2	249.06
19	19	2	1996-07-10 00:00:00	1996-05-01 00:00:00	1996-10-15 00:00:00	2	0.78
20	20	2	1996-07-10 00:00:00	1996-06-16 00:00:00	1996-10-27 00:00:00	2	36.71

ProductID	ProductName	SupplierID	CategoryID	Unit	UnitPrice	UnitsInStock	UnitsOnOrder	ReorderLevel	Discontinued
1	Chai	1	1	10	18	39	0	10	0
2	Sauerkraut	1	1	10	20	45	0	10	0
3	Biryani	1	1	10	15	25	0	10	0
4	Chorizo	1	1	10	15	35	0	10	0
5	Pastrami	1	1	10	15	35	0	10	0
6	Capricorn	1	1	10	15	35	0	10	0
7	Pastrami	1	1	10	15	35	0	10	0
8	Pastrami	1	1	10	15	35	0	10	0
9	Pastrami	1	1	10	15	35	0	10	0
10	Pastrami	1	1	10	15	35	0	10	0

Step 2:

- From the columns displayed → go to Order_Details → click on the “expand” icon beside it → select the column of interest (ProductID, UnitPrice, Quantity) → click on “OK”

The screenshot shows the Power BI Data Editor interface. A context menu is open over the 'Order_Details' table, specifically over the 'Quantity' column. The 'Expand' option is selected in the 'Search Columns to Expand' dropdown. The 'Quantity' checkbox is checked under the 'Select All Columns' section. Other options like 'Discount', 'Order', and 'Product' are also listed. The 'OK' button is visible at the bottom right of the dialog.

- All the columns will be displayed

The screenshot shows the Power BI Data Editor interface after expanding the 'Order_Details' table. The expanded table now includes five columns: 'ShipPostalCode', 'ShipCountry', 'ProductID', 'UnitPrice', and 'Quantity'. The 'ProductID' column is the primary key, indicated by a bold font. The 'UnitPrice' and 'Quantity' columns are also clearly visible. The rest of the table structure remains the same, including rows for various orders and their details.

D) Calculate the line total for each Order_Details row

In this step, you create a Custom Column to calculate the line total for each Order_Details row.

You can create calculations based on the columns you are importing, so you can enrich the data that you connect to.

Step 1:

- From the “Add column” ribbon, select “Custom Column”.

Step 2:

- Change the New column name as “Line Total”.
- For formula, select “Order_Details.UnitPrice” from available columns list and click on “insert”. Insert an “asterisk (*)”. Select “Order_Details.Quantity” from available columns list and click on “insert”.
- Select “OK”

➤ “Line total” column will be displayed.

The screenshot shows the Microsoft Power BI Data Editor interface. The top ribbon has tabs for File, Home, Transform, Add Column, View, Tools, and Help. The Home tab is selected. The ribbon also includes Close & Apply, New Source, Recent Sources, Enter Data, Data source settings, Manage Parameters, Refresh, Advanced Editor, Properties, Choose Columns, Remove Columns, Keep Rows, Remove Rows, Sort, Split Column, Group By, Replace Values, Data Type: Any, Use First Row as Headers, Merge Queries, Append Queries, Text Analytics, Vision, Combine Files, Azure Machine Learning, and AI Insights. Below the ribbon is a 'Queries' pane on the left containing a list of queries. The main area displays a preview of a table named 'Table.ReorderColumns(#"Added Custom", {"OrderID", "CustomerID", "EmployeeID", "OrderDate", "RequiredDate", "ShippedDate", "ShipVia", "Freight", "ShipName", "Order_Details.ProductID", "Order_Details.UnitPrice", "Order_Details.Quantity", "Customer", "Employee", "Shipper", "Line Total"}'. The preview shows 21 columns and 999+ rows. The 'Line Total' column is the last column on the right. The table data includes various product IDs, unit prices, quantities, and their calculated totals. The bottom status bar indicates '21 COLUMNS, 999+ ROWS' and 'Column profiling based on top 1000 rows'.

	OrderID	CustomerID	EmployeeID	OrderDate	RequiredDate	ShippedDate	ShipVia	Freight	ShipName	ProductID	UnitPrice	Quantity	Customer	Employee	Shipper	Line Total
1	11									12	Record	Record	Record	Record	Record	168
2	42									10	Record	Record	Record	Record	Record	98
3	72									5	Record	Record	Record	Record	Record	174
4	14									9	Record	Record	Record	Record	Record	167.4
5	51									40	Record	Record	Record	Record	Record	1696
6	41									10	Record	Record	Record	Record	Record	77
7	51									35	Record	Record	Record	Record	Record	1484
8	65									15	Record	Record	Record	Record	Record	252
9	22									6	Record	Record	Record	Record	Record	100.8
10	57									15	Record	Record	Record	Record	Record	234
11	65									20	Record	Record	Record	Record	Record	336
12	20									40	Record	Record	Record	Record	Record	2592
13	33									25	Record	Record	Record	Record	Record	50
14	60									40	Record	Record	Record	Record	Record	1088
15	31									20	Record	Record	Record	Record	Record	200
16	39									42	Record	Record	Record	Record	Record	604.8
17	49									40	Record	Record	Record	Record	Record	640
18	24									15	Record	Record	Record	Record	Record	54
19	55									21	Record	Record	Record	Record	Record	403.2
20	74									21	Record	Record	Record	Record	Record	168
21	-									-	-	-	-	-	-	

E) Rename and reorder columns in the query

In this step, we will rename the necessary columns and change their orders for making a better model.

Step 1:

- Drag “Line Total” column towards the left side to reorder the data.

The screenshot shows the Microsoft Power BI Data Editor interface. A table named 'Table.ReorderColumns' is displayed with the following schema:

	ShipPostalCode	ShipCountry	Line Total	ProductID	UnitPrice	Quantity	Customer	Employee
1	u/l 51100	France	168	11	14	12	Record	
2	u/l 51100	France	98	42	9.8	10	Record	
3	u/l 51100	France	174	72	34.8	5	Record	
4	u/l 44087	Germany	167.4	14	18.6	9	Record	
5	u/l 44087	Germany	1696	51	42.4	40	Record	
6	05454-876	Brazil	77	41	7.7	10	Record	
7	05454-876	Brazil	1484	51	42.4	35	Record	
8	05454-876	Brazil	252	65	16.8	15	Record	
9	u/l 69004	France	100.8	22	16.8	6	Record	
10	u/l 69004	France	234	57	15.6	15	Record	
11	u/l 69004	France	336	65	16.8	20	Record	
12	u/l B-6000	Belgium	2592	20	64.8	40	Record	
13	u/l B-6000	Belgium	50	33	2	25	Record	
14	u/l B-6000	Belgium	1088	60	27.2	40	Record	
15	05454-876	Brazil	200	31	10	20	Record	
16	05454-876	Brazil	604.8	39	14.4	42	Record	
17	05454-876	Brazil	640	49	16	40	Record	
18	u/l 3012	Switzerland	54	24	3.6	15	Record	
19	u/l 3012	Switzerland	403.2	55	19.2	21	Record	
20	u/l 3012	Switzerland	168	74	8	21	Record	
21								

Column profiling based on top 1000 rows. PREVIEW DOWNLOADED ON TUESDAY.

Step 2:

- Rename Order_Details columns by removing the prefix – Order_Details from their heading i.e., changing Order_Details.ProductID to ProductID, Order_Details.UnitPrice to UnitPrice and Order_Details.Quantity to Quantity.

The screenshot shows the Microsoft Power BI Data Editor interface. A table named 'Table.RenameColumns' is displayed with the following schema:

	ShipPostalCode	ShipCountry	Line Total	ProductID	UnitPrice	Quantity	Customer	Employee
1	u/l 51100	France	168	11	14	12	Record	
2	u/l 51100	France	98	42	9.8	10	Record	
3	u/l 51100	France	174	72	34.8	5	Record	
4	u/l 44087	Germany	167.4	14	18.6	9	Record	
5	u/l 44087	Germany	1696	51	42.4	40	Record	
6	05454-876	Brazil	77	41	7.7	10	Record	
7	05454-876	Brazil	1484	51	42.4	35	Record	
8	05454-876	Brazil	252	65	16.8	15	Record	
9	u/l 69004	France	100.8	22	16.8	6	Record	
10	u/l 69004	France	234	57	15.6	15	Record	
11	u/l 69004	France	336	65	16.8	20	Record	
12	u/l B-6000	Belgium	2592	20	64.8	40	Record	
13	u/l B-6000	Belgium	50	33	2	25	Record	
14	u/l B-6000	Belgium	1088	60	27.2	40	Record	
15	05454-876	Brazil	200	31	10	20	Record	
16	05454-876	Brazil	604.8	39	14.4	42	Record	
17	05454-876	Brazil	640	49	16	40	Record	
18	u/l 3012	Switzerland	54	24	3.6	15	Record	
19	u/l 3012	Switzerland	403.2	55	19.2	21	Record	
20	u/l 3012	Switzerland	168	74	8	21	Record	
21								

Column profiling based on top 1000 rows. PREVIEW DOWNLOADED ON TUESDAY.

- Applying the necessity changes by selecting “Close & Apply” option from “Home” ribbon.

The screenshot shows the Microsoft Power BI desktop interface. The ribbon at the top has the 'Home' tab selected. Below the ribbon is a table with the following data:

	ShipPostalCode	ShipCountry	Line Total	ProductID	UnitPrice	Quantity	Customer	Employee
1	u11 51100	France	168	11	14	12	Record	Record
2	u11 51100	France	98	42	9.8	10	Record	Record
3	u11 51100	France	174	72	34.8	5	Record	Record
4	u11 44087	Germany	167.4	14	18.6	9	Record	Record
5	u11 44087	Germany	1696	51	42.4	40	Record	Record
6	05454-876	Brazil	77	41	7.7	10	Record	Record
7	05454-876	Brazil	1484	51	42.4	35	Record	Record
8	05454-876	Brazil	252	65	16.8	15	Record	Record
9	u11 69004	France	100.8	22	16.8	6	Record	Record
10	u11 69004	France	234	57	15.6	15	Record	Record
11	u11 69004	France	336	65	16.8	20	Record	Record
12	u11 B-6000	Belgium	2592	20	64.8	40	Record	Record
13	u11 B-6000	Belgium	50	33	2	25	Record	Record
14	u11 B-6000	Belgium	1088	60	27.2	40	Record	Record
15	05454-876	Brazil	200	31	10	20	Record	Record
16	05454-876	Brazil	604.8	39	14.4	42	Record	Record
17	05454-876	Brazil	640	49	16	40	Record	Record
18	u11 3012	Switzerland	54	24	3.6	15	Record	Record
19	u11 3012	Switzerland	403.2	55	19.2	21	Record	Record
20	u11 3012	Switzerland	168	74	8	21	Record	Record
21

21 COLUMNS, 999+ ROWS Column profiling based on top 1000 rows PREVIEW DOWNLOADED ON TUESDAY

F) Combine the Products and Total Sales queries

In this step, you confirm that a relationship is established between the Products and Total Sales

Power BI Desktop does not require you to combine queries to report on them. Instead, you can create Relationships between datasets. These relationships can be created on any column that is common to your datasets we have Orders and Products data that share a common 'ProductID' field, so we need to ensure there's a relationship between them in the model we're using with Power BI Desktop. Simply specify in Power BI Desktop that the columns from each table are related (i.e. columns that have the same values). Power BI Desktop works out the direction and cardinality of the relationship for you. In some cases, it will even detect the relationships automatically.

Step 1:

- Select “Manage Relationships” from the “Home” ribbon.

The screenshot shows the Microsoft Power BI Desktop interface. The top navigation bar has tabs: File, Home (which is selected), Help, and Table tools. Under the Home tab, there are icons for Paste, Cut, Copy, Get data from Excel, Data, SQL Server, Enter data, Data Transform Refresh, Manage relationships, New measure, Quick measure column, New table, Manage roles, View as, Sensitivity, Publish, Security, and Share. A search bar labeled 'Search' is also present. Below the ribbon, a preview of the 'Orders' table is shown with 2,155 rows. The table has columns: OrderID, CustomerID, EmployeeID, OrderDate, RequiredDate, ShippedDate, ShipVia, Freight, ShipName, ShipAddress, ShipCity, and ShipRegion. The preview shows several rows of data, such as OrderID 10273 for customer QUICK with various shipping details.

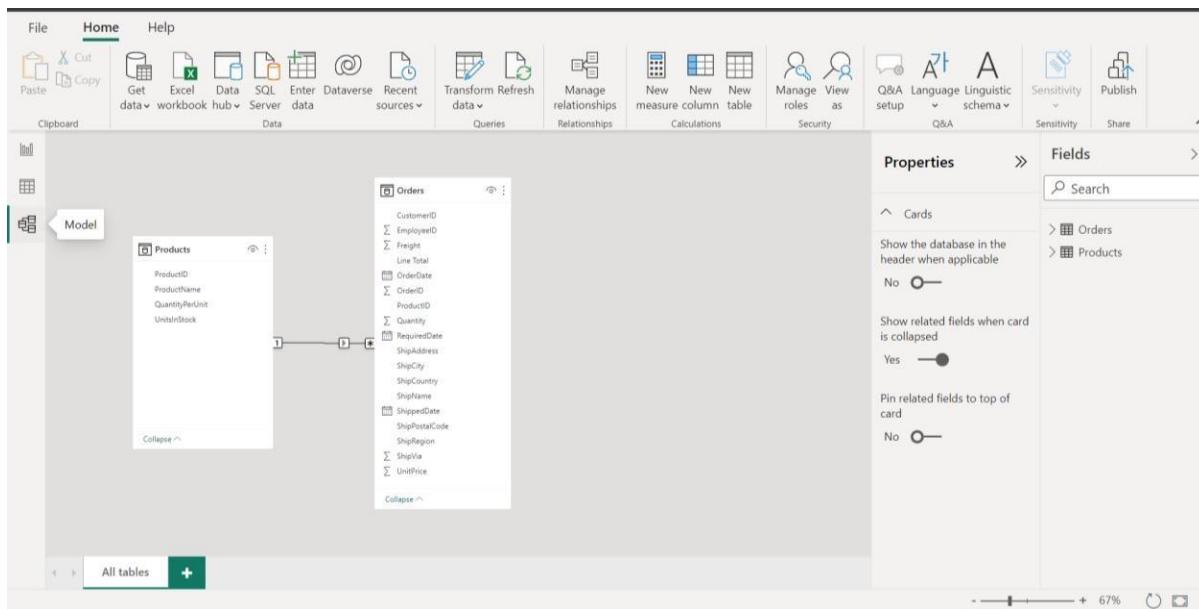
- As displayed a relationship is created by default between “Orders and Products”.
- To create a relation between two data, click on “New”

The screenshot shows the 'Manage relationships' dialog box in Excel. It displays a table with two columns: 'From: Table (Column)' and 'To: Table (Column)'. The 'From' column has a single entry: 'Orders (ProductID)'. The 'To' column also has a single entry: 'Products (ProductID)'. A green checkmark is placed next to this row. At the bottom of the dialog are buttons for 'New...', 'Autodetect...', 'Edit...', and 'Delete'. A 'Close' button is located at the bottom right. In the background, a table titled 'Orders' is visible, showing 2,155 rows of data.

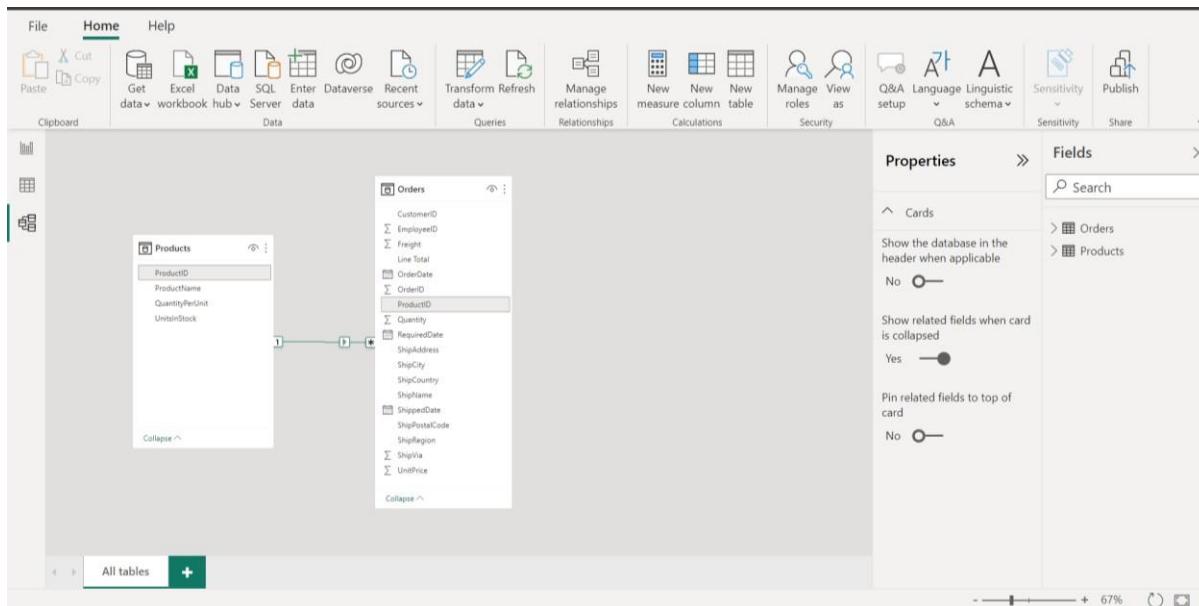
- Select “Products” in from table and “Orders” in to table.
- Here, the relationship is already created by default so we cannot create the same relationship.

The screenshot shows the 'Create relationship' dialog box in Excel. It has two main sections: 'From' and 'To'. The 'From' section is set to 'Products' and shows a list of three products: 1. Chai, 2. Chang, 3. Aniseed Syrup. The 'To' section is set to 'Orders' and shows a list of three orders: K-Stop, K-Stop, K-Stop. Below these lists is a 'Cardinality' dropdown. Underneath the tables are two checkboxes: 'Make this relationship active' and 'Assume referential integrity'. A yellow warning bar at the bottom states: 'There's already a relationship between these two columns.' At the bottom right are 'OK' and 'Cancel' buttons.

- From the left panel, select “Model”
- Here, you can view the relationship created.



- When you click on relationship, you can view that “ProductID” is in common to create this relationship model.



Practical 3

Implementation of Classification Algorithm in R Programming.

- ❖ Consider the annual rainfall details at a place starting from January 2012. We create an R time series object for a period of 12 months and plot it.

```
# Get the data points in form of a R vector.
```

```
rainfall <-c(799,1174.8,865.1,1334.6,635.4,918.5,685.5,998.6,784.2,985,882.8,1071)
```

```
# Convert it to a time series object.
```

```
rainfall.timeseries <- ts(rainfall,start = c(2012,1),frequency = 12)
```

```
# Print the timeseries data.
```

```
print(rainfall.timeseries)
```

```
# Give the chart file a name.
```

```
png(file = "rainfall.png")
```

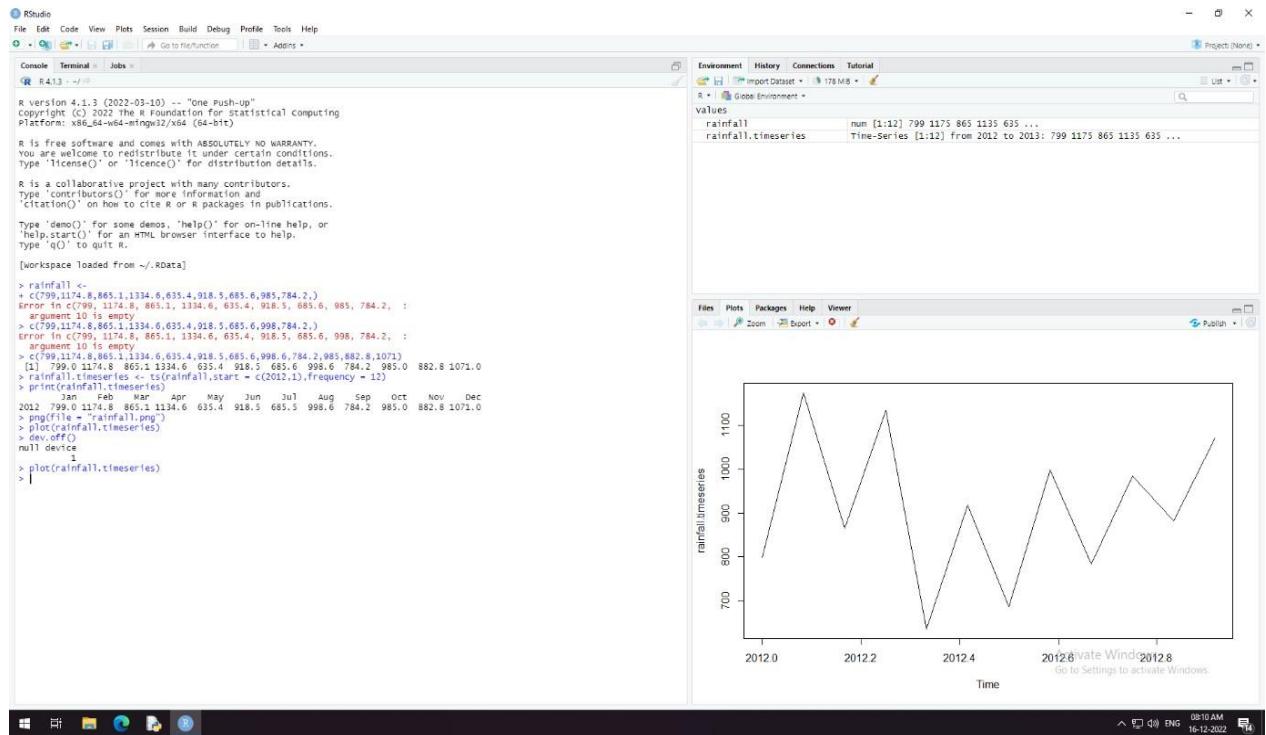
```
# Plot a graph of the time series.
```

```
plot(rainfall.timeseries)
```

```
# Save the file.
```

```
dev.off()
```

Output:



Practical 4

K-means Clustering using R Programming.

K-means clustering (MacQueen 1967) is one of the most commonly used unsupervised machine learning algorithm for partitioning a given data set into a set of k groups (i.e. k clusters), where k represents the number of groups pre-specified by the analyst. It classifies objects in multiple groups (i.e., clusters), such that objects within the same cluster are as similar as possible (i.e., high intra-class similarity), whereas objects from different clusters are as dissimilar as possible (i.e., low inter-class similarity). In k-means clustering, each cluster is represented by its center (i.e, centroid) which corresponds to the mean of points assigned to the cluster. The algorithm tries to find groups by minimizing the distance between the observations, called local optimal solutions. The distances are measured based on the coordinates of the observations. For instance, in a two-dimensional space, the coordinates are simple

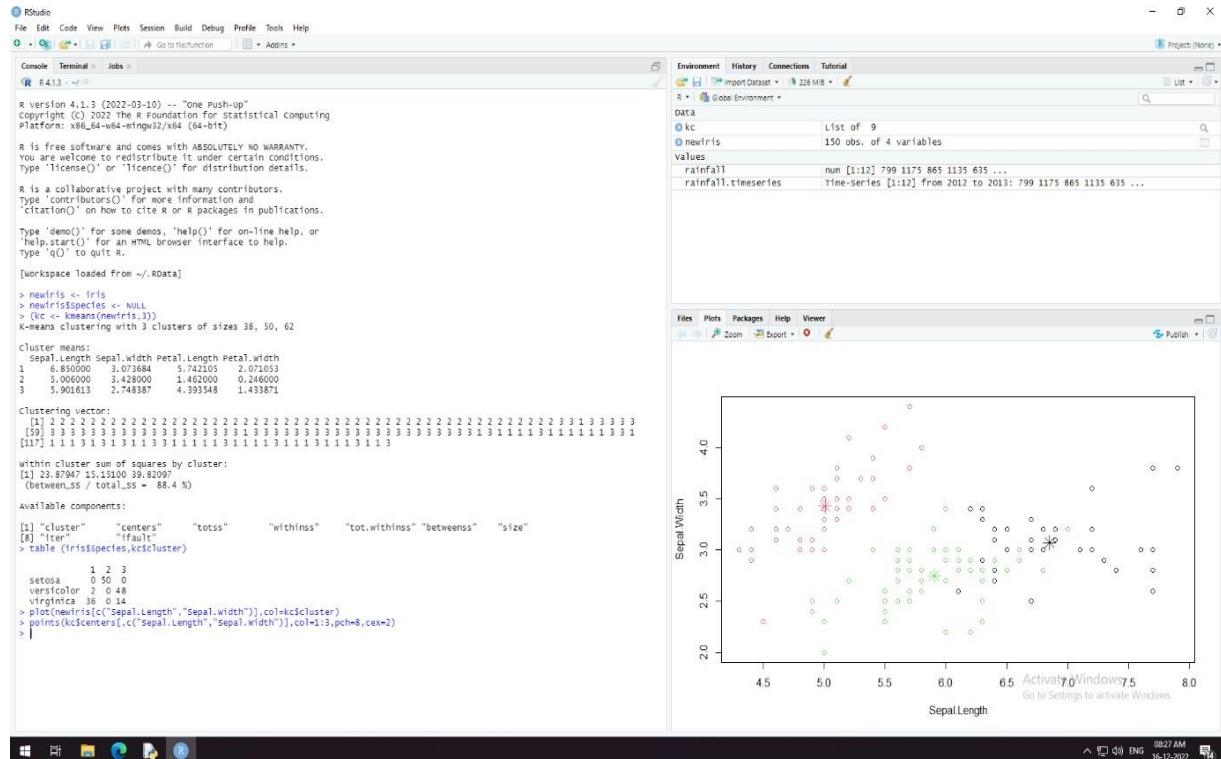
Theory

- K-Means clustering groups the data on similar groups. The algorithm is as follows:
- Choose the number K clusters.
- Select at random K points, the centroids(Not necessarily from the given data).
- Assign each data point to closest centroid that forms K clusters.
- Compute and place the new centroid of each centroid.
- Reassign each data point to new cluster.

The algorithm works as follow:

- Step 1: Choose groups in the feature plan randomly
- Step 2: Minimize the distance between the cluster center and the different observations (centroid). It results in groups with observations
- Step 3: Shift the initial centroid to the mean of the coordinates within a group.
- Step 4: Minimize the distance according to the new centroids. New boundaries are created. Thus, observations will move from one group to another.
- Repeat until no observation changes groups

Output:



Practical 5

Prediction Using Linear Regression in R Programming.

In Linear Regression these two variables are related through an equation, where exponent (power) of both these variables is 1. Mathematically a linear relationship represents a straight line when plotted as a graph. A non-linear relationship where the exponent of any variable is not equal to 1 creates a curve.

$y = ax + b$ is an equation for linear regression.

Where, y is the response variable, x is the predictor variable and a and b are constants which are called the coefficients.

A simple example of regression is predicting weight of a person when his height is known. To do this we need to have the relationship between height and weight of a person.

The steps to create the relationship is –

- ❖ Carry out the experiment of gathering a sample of observed values of height and corresponding weight.
- ❖ Create a relationship model using the lm() functions in R.
- ❖ Find the coefficients from the model created and create the mathematical equation using these
- ❖ Get a summary of the relationship model to know the average error in prediction. Also called residuals.
- ❖ To predict the weight of new persons, use the predict() function in R.

Input Data

Below is the sample data representing the observations –

```
# Values of height
151, 174, 138, 186, 128, 136, 179, 163, 152, 131
```

```
# Values of weight.
63, 81, 56, 91, 47, 57, 76, 72, 62, 48
```

lm() Function

This function creates the relationship model between the predictor and the response variable.

Syntax

The basic syntax for lm() function in linear regression is –

```
lm(formula,data)
```

Following is the description of the parameters used –

- formula is a symbol presenting the relation between x and y.
- data is the vector on which the formula will be applied.

Create Relationship Model & get the Coefficients

```
x <- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)
```

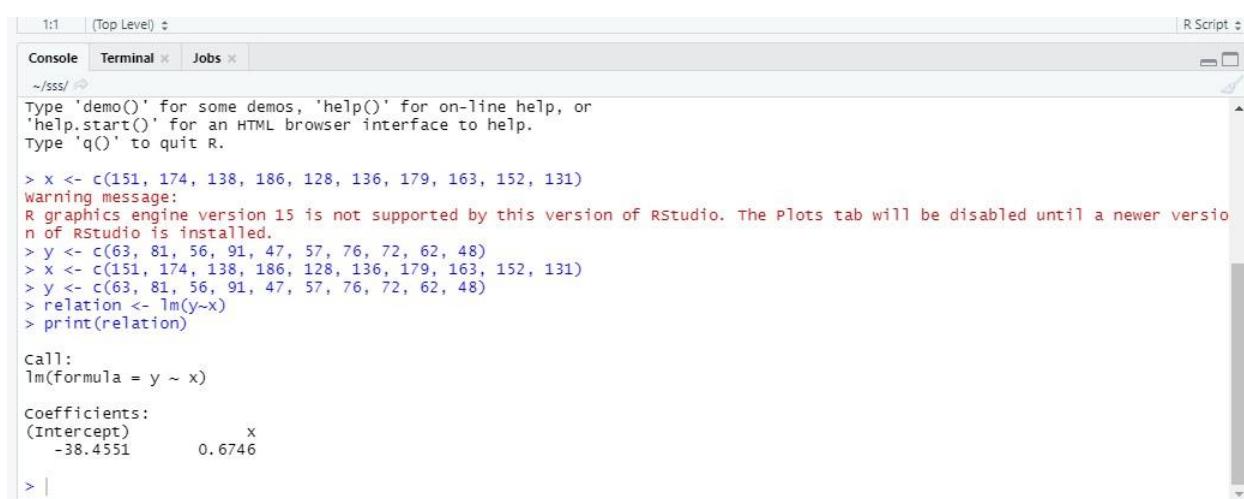
```
y <- c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)
```

```
# Apply the lm() function.
```

```
relation <- lm(y~x)
```

```
print(relation)
```

When we execute the above code, it produces the following result –



The screenshot shows the RStudio interface with the console tab selected. The console window displays the following R code and its output:

```
1:1 | (Top Level) $ R Script ▾
Console Terminal Jobs ~ /SSS/ ↻
Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> x <- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)
Warning message:
R graphics engine version 15 is not supported by this version of RStudio. The Plots tab will be disabled until a newer version of RStudio is installed.
> y <- c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)
> x <- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)
> y <- c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)
> relation <- lm(y~x)
> print(relation)

Call:
lm(formula = y ~ x)

Coefficients:
(Intercept)          x
-38.4551        0.6746
> |
```

Get the Summary of the Relationship

```
x <- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)
```

```
y <- c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)
```

```
# Apply the lm() function.
```

```
relation <- lm(y~x)
```

```
print(summary(relation))
```

When we execute the above code, it produces the following result –

```
> y <- c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)
> x <- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)
> relation <- lm(y~x)
> print(summary(relation))

call:
lm(formula = y ~ x)

Residuals:
    Min      1Q  Median      3Q     Max 
-6.3002 -1.6629  0.0412  1.8944  3.9775 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) -38.45509   8.04901  -4.778  0.00139 ***
x            0.67461   0.05191  12.997 1.16e-06 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.253 on 8 degrees of freedom
Multiple R-squared:  0.9548,    Adjusted R-squared:  0.9491 
F-statistic: 168.9 on 1 and 8 DF,  p-value: 1.164e-06

> |
```

predict() Function

Syntax

The basic syntax for predict() in linear regression is –

```
predict(object, newdata)
```

Following is the description of the parameters used –

- object is the formula which is already created using the lm() function.
- newdata is the vector containing the new value for predictor variable.

Predict the weight of new persons

```
# The predictor vector.
```

```
x <- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)
```

```
# The response vector.
```

```
y <- c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)
```

```
# Apply the lm() function.
```

```
relation <- lm(y~x)
```

```
# Find weight of a person with height 170.
```

```
a <- data.frame(x = 170)
```

```
result <- predict(relation,a)
```

```
print(result)
```

Result:

```
1
```

```
76.22869
```

```

1:1 | (Top Level) |
Console Terminal Jobs
~/SSS/ 
(Intercept) -38.45509   8.04901  -4.778  0.00139 ***
x            0.67461    0.05191  12.997  1.16e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Residual standard error: 3.253 on 8 degrees of freedom
Multiple R-squared:  0.9548, Adjusted R-squared:  0.9491
F-statistic: 168.9 on 1 and 8 DF,  p-value: 1.164e-06

> # The predictor vector.
> x <- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)
> # The response vector.
> y <- c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)
> # Apply the lm() function.
> relation <- lm(y~x)
> # Find weight of a person with height 170.
> a <- data.frame(x = 170)
> result <- predict(relation,a)
> print(result)
1
76.22869
>

```

Visualize the Regression Graphically

Create the predictor and response variable.

```

x <- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)
y <- c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)
relation <- lm(y~x)

```

Give the chart file a name.

```
png(file = "linearregression.png")
```

Plot the chart.

```

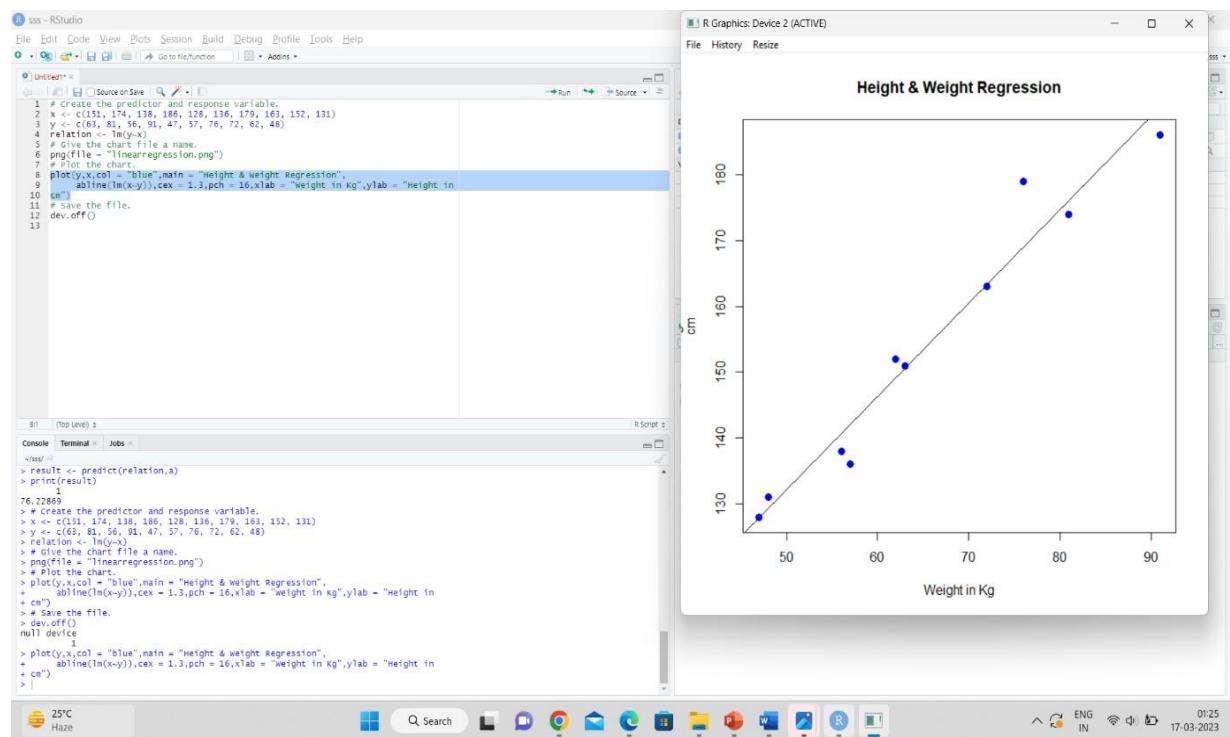
plot(y,x,col = "blue",main = "Height & Weight Regression",
abline(lm(x~y)),cex = 1.3,pch = 16,xlab = "Weight in Kg",ylab = "Height in cm")

```

Save the file.

```
dev.off()
```

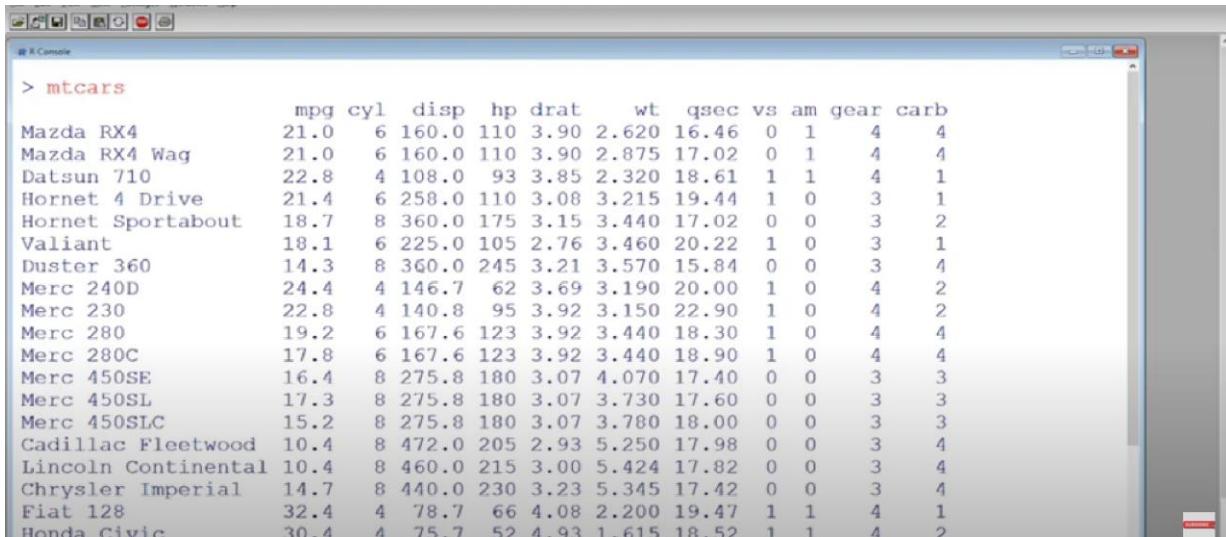
Output:



Practical 6

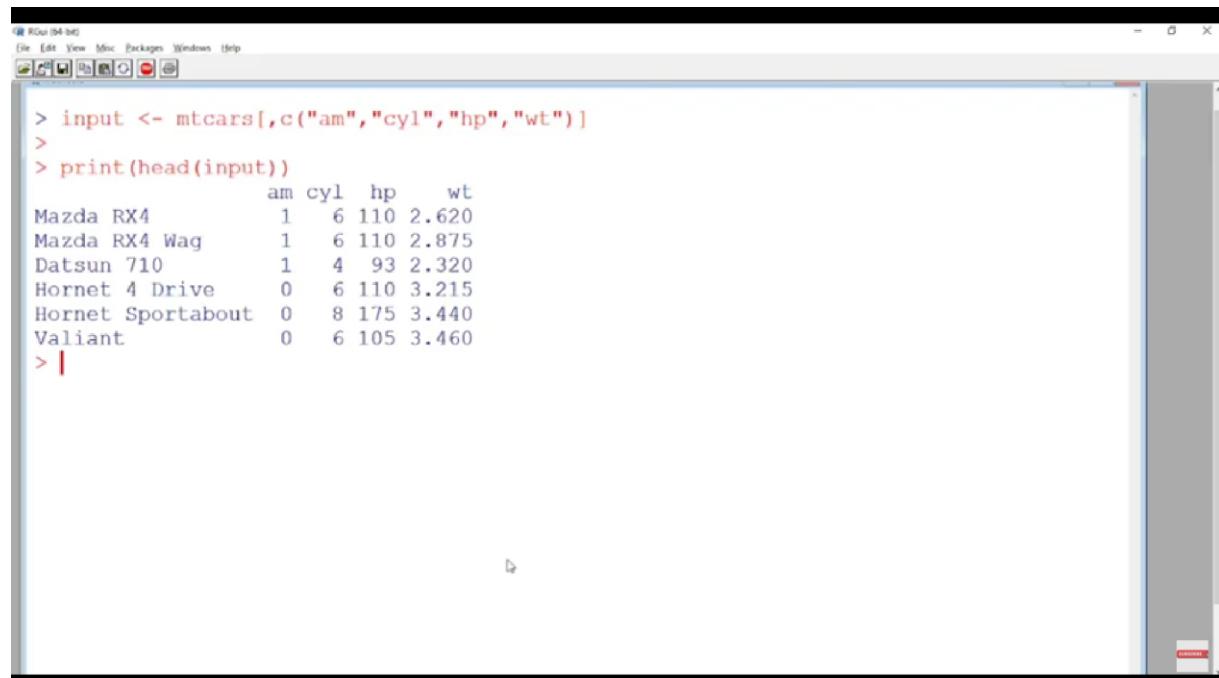
Perform the logistic regression on the given data warehouse data..

Logistic regression is a data analysis technique that uses mathematics to find the relationships between two data factors. It then uses this relationship to predict the value of one of those factors based on the other. The prediction usually has a finite number of outcomes, like yes or no.



The screenshot shows the R console window with the command `> mtcars` and its output. The output displays a data frame with 32 rows and 11 columns. The columns are mpg, cyl, disp, hp, drat, wt, qsec, vs, am, gear, and carb. The data represents various car models with their respective fuel consumption, engine characteristics, and transmission details.

		mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4		21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag		21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
Datsun 710		22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive		21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout		18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2
Valiant		18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1
Duster 360		14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4
Merc 240D		24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2
Merc 230		22.8	4	140.8	95	3.92	3.150	22.90	1	0	4	2
Merc 280		19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4
Merc 280C		17.8	6	167.6	123	3.92	3.440	18.90	1	0	4	4
Merc 450SE		16.4	8	275.8	180	3.07	4.070	17.40	0	0	3	3
Merc 450SL		17.3	8	275.8	180	3.07	3.730	17.60	0	0	3	3
Merc 450SLC		15.2	8	275.8	180	3.07	3.780	18.00	0	0	3	3
Cadillac Fleetwood		10.4	8	472.0	205	2.93	5.250	17.98	0	0	3	4
Lincoln Continental		10.4	8	460.0	215	3.00	5.424	17.82	0	0	3	4
Chrysler Imperial		14.7	8	440.0	230	3.23	5.345	17.42	0	0	3	4
Fiat 128		32.4	4	78.7	66	4.08	2.200	19.47	1	1	4	1
Honda Civic		30.4	4	75.7	52	4.93	1.615	18.52	1	1	4	2



The screenshot shows the R console window with the following code and its output:

```

> input <- mtcars[,c("am","cyl","hp","wt")]
>
> print(head(input))
      am cyl   hp   wt
Mazda RX4     1   6 110 2.620
Mazda RX4 Wag 1   6 110 2.875
Datsun 710    1   4  93 2.320
Hornet 4 Drive 0   6 110 3.215
Hornet Sportabout 0   8 175 3.440
Valiant       0   6 105 3.460
> |

```

```

> input <- mtcars[,c("am","cyl","hp","wt")]
>
> print(head(input))
      am cyl   hp   wt
Mazda RX4     1   6 110 2.620
Mazda RX4 Wag 1   6 110 2.875
Datsun 710    1   4  93 2.320
Hornet 4 Drive 0   6 110 3.215
Hornet Sportabout 0   8 175 3.440
Valiant       0   6 105 3.460
> input <- mtcars[,c("am","cyl","hp","wt")]
> am.data = glm(formula = am ~ cyl + hp + wt, data = input, family = binomial)

```

```

Hornet 4 Drive     0   6 110 3.215
Hornet Sportabout  0   8 175 3.440
Valiant           0   6 105 3.460
> input <- mtcars[,c("am","cyl","hp","wt")]
> am.data = glm(formula = am ~ cyl + hp + wt, data = input, family = binomial)
> print(summary(am.data))

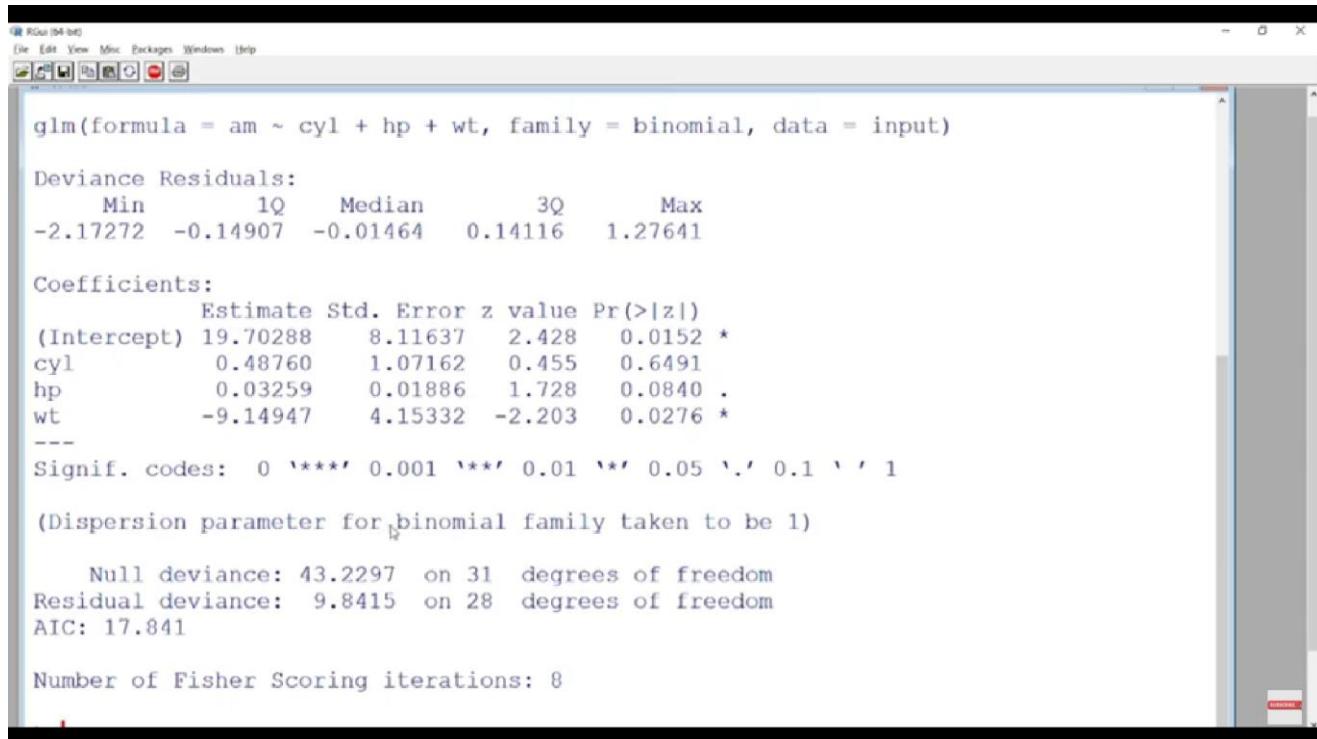
Call:
glm(formula = am ~ cyl + hp + wt, family = binomial, data = input)

Deviance Residuals:
    Min      1Q  Median      3Q      Max 
-2.17272 -0.14907 -0.01464  0.14116  1.27641 

Coefficients:
            Estimate Std. Error z value Pr(>|z|)    
(Intercept) 19.70288   8.11637   2.428   0.0152 *  
cyl          0.48760   1.07162   0.455   0.6491    
hp           0.03259   0.01886   1.728   0.0840 .    
wt          -9.14947   4.15332  -2.203   0.0276 *  
---
Signif. codes:  0 '****' 0.001 '***' 0.01 '**' 0.05 '*' 0.1 ' ' 1


```

Output



The screenshot shows the RStudio interface with the following R code and output:

```
glm(formula = am ~ cyl + hp + wt, family = binomial, data = input)

Deviance Residuals:
    Min      1Q  Median      3Q     Max 
-2.17272 -0.14907 -0.01464  0.14116  1.27641 

Coefficients:
            Estimate Std. Error z value Pr(>|z|)    
(Intercept) 19.70288   8.11637  2.428   0.0152 *  
cyl          0.48760   1.07162  0.455   0.6491    
hp           0.03259   0.01886  1.728   0.0840 .  
wt          -9.14947   4.15332 -2.203   0.0276 *  
---
Signif. codes:  0 '****' 0.001 '***' 0.01 '**' 0.05 '*' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 43.2297 on 31 degrees of freedom
Residual deviance: 9.8415 on 28 degrees of freedom
AIC: 17.841

Number of Fisher Scoring iterations: 8
```

Practical 7

Apply What – if Analysis for data visualization. Design and generate necessary reports based on the data warehouse data.

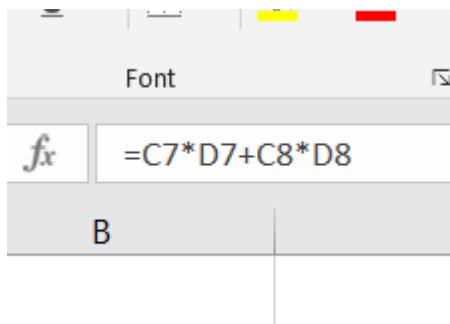
A book store and have 100 books in storage. You sell a certain % for the highest price of \$50 and a certain % for the lower price of \$20.

The screenshot shows a Microsoft Excel spreadsheet with the following data:

		B	C	D	E
1	BOOK STORE				
2					
3	total number of books		% sold for the highest price		
4		100		60%	
5					
6		number of books		unit profit	
7	highest price		60	50	
8	lower price		40	20	
9					
10		total profit		3800	
11					
12					

The formula in cell D10 is =C7*D7+C8*D8.

If you sell 60% for the highest price, cell D10 calculates a total profit of $60 * \$50 + 40 * \$20 = \$3800$.

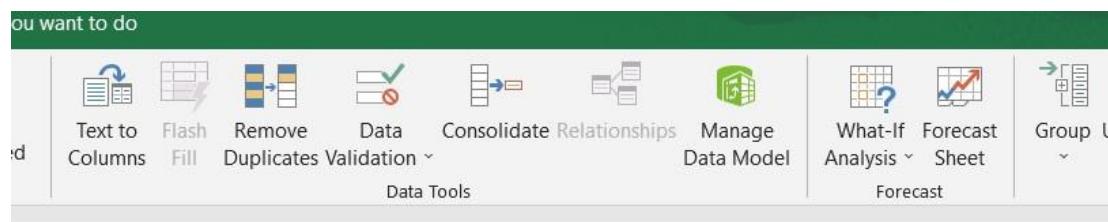


Create Different Scenarios But what if you sell 70% for the highest price? And what if you sell 80% for the highest price? Or 90%, or even 100%? Each different percentage is a different scenario. You can use the Scenario Manager to create these scenarios. Note: You can simply type in a different percentage into cell C4 to see the corresponding result of a scenario in cell D10. However, what-if analysis enables you to easily compare the results of different scenarios. Read on.

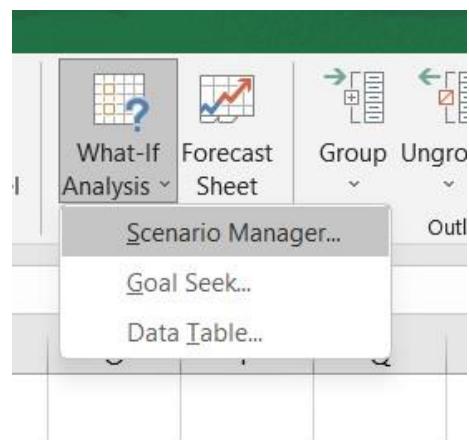
For C8 this formula is applied 100-60(B4-C7)

C8	A	B	C	D	E	F
			=B4-C7			
1	book store					
2						
3		total no of books	% for the highest			
4		100	60%			
5						
6		number of books	unit profit			
7	highest price	60	50			
8	lower price	40	20			
9			total	3800		
10						
11						
12						

1. On the Data tab, in the Forecast group, click What-If Analysis

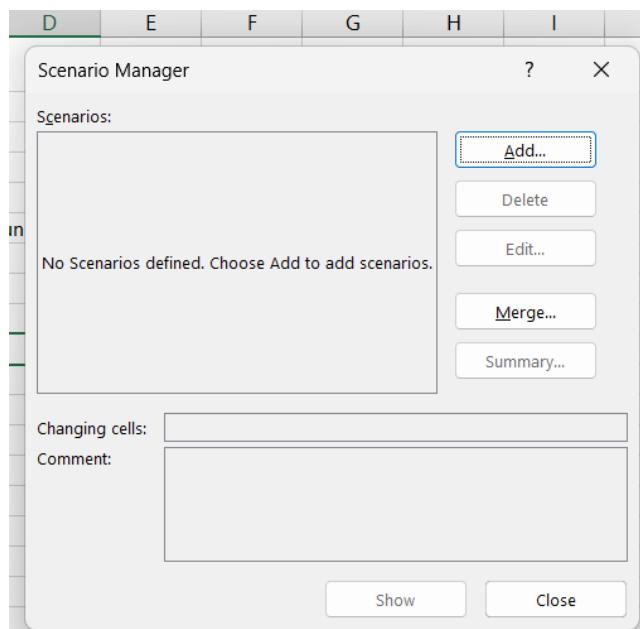


2. Click Scenario Manager.

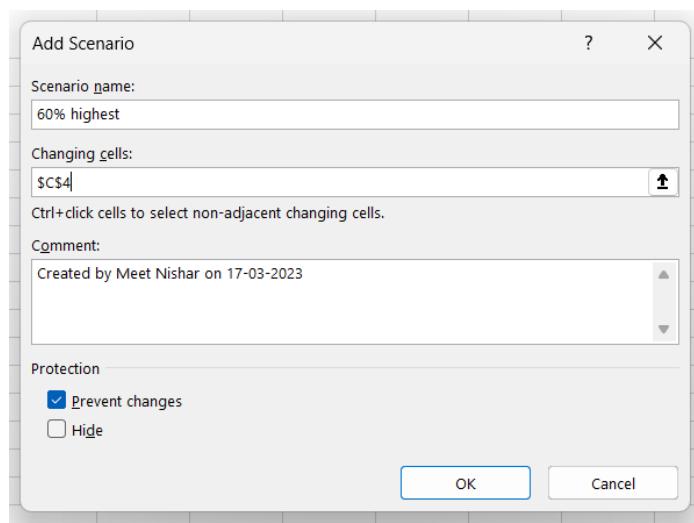


The Scenario Manager dialog box appears.

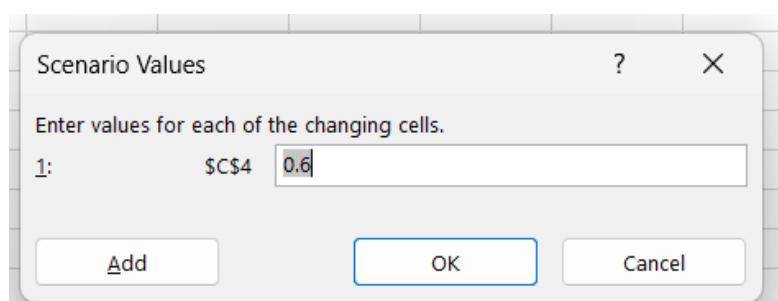
3. Add a scenario by clicking on Add.



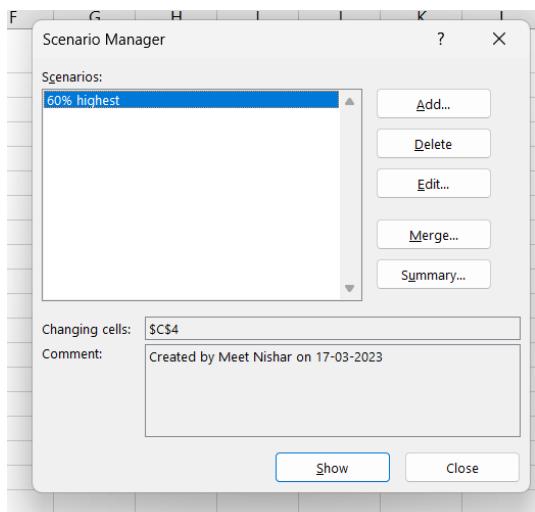
4. Type a name (60% highest), select cell C4 (% sold for the highest price) for the Changing cells and Click on OK.



5. Enter the corresponding value 0.6 and click on OK again.



6. 60% highest will be added to the Dialog box



7. Type one more name (70% highest), select cell C4 (% sold for the highest price) for the Changing cells and Click on OK.
8. Enter the corresponding value 0.7 and click on OK.

Get & Transform Data						Queries & Connections
C8	A	B	C	D	E	F
1	book store					
2						
3	total no of books	% for the highest				
4	100	70%				
5						
6		number of books	unit profit			
7	highest price	70	50			
8	lower price	30	20			
9		total	4100			
10						
11						
12						

9. Next, add 3 other scenarios (80%, 90%, 100%)

Finally, your Scenario Manager should be consistent with the picture below:

The screenshot shows a Microsoft Excel spreadsheet titled "Book3 - Excel". The Data tab is selected, and the Scenario Manager dialog box is open. The dialog lists five scenarios: "60% highest", "70% highest", "80% highest", "90% highest", and "100% highest". The "100% highest" scenario is currently selected. The "Changing cells:" field is set to "\$C\$4". The "Comment:" field contains the text "Created by Meet Nishar on 18-03-2023". In the background, there is a table with the following data:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
1	book store																					
2																						
3	total no of books	% for the highest																				
4		100	100%																			
5																						
6		number of books	unit profit																			
7	highest price	100	50																			
8	lower price	0	20																			
9		total	5000																			
10																						
11																						
12																						
13																						
14																						
15																						
16																						
17																						
18																						
19																						
20																						
21																						
22																						
23																						
24																						
25																						
26																						
27																						
28																						
29																						

The status bar at the bottom shows "Sheet1", "Accessibility: Good to go", "18°C Cloudy", and the date "18-03-2023".

Practical 8

Data Analyzing using Time-series Analysis.

- Time series is a series of data points in which each data point is associated with a timestamp. A simple example is the price of a stock in the stock market at different points of time on a given day. Another example is the amount of rainfall in a region at different months of the year. R language uses many functions to create, manipulate and plot the time series data. The data for the time series is stored in an R object called time-series object. It is also a R data object like a vector or data frame.
- The time series object is created by using the ts() function.
- The basic syntax for ts() function in time series analysis is –
`timeseries.object.name <- ts(data, start, end, frequency)`
- Following is the description of the parameters used –
 - Data is a vector or matrix containing the values used in the time series.
 - Start specifies the start time for the first observation in time series.
 - End specifies the end time for the last observation in time series.
 - Frequency specifies the number of observations per unit time.

Except the parameter "data" all other parameters are optional.

- ❖ Consider the annual rainfall details at a place starting from January 2012. We create an R time series object for a period of 12 months and plot it.

```
# Get the data points in form of a R vector.
```

```
rainfall <-c(799,1174.8,865.1,1334.6,635.4,918.5,685.5,998.6,784.2,985,882.8,1071)
```

```
# Convert it to a time series object.
```

```
rainfall.timeseries <- ts(rainfall,start = c(2012,1),frequency = 12)
```

```
# Print the timeseries data.
```

```
print(rainfall.timeseries)
```

```
# Give the chart file a name.
```

```
png(file = "rainfall.png")
```

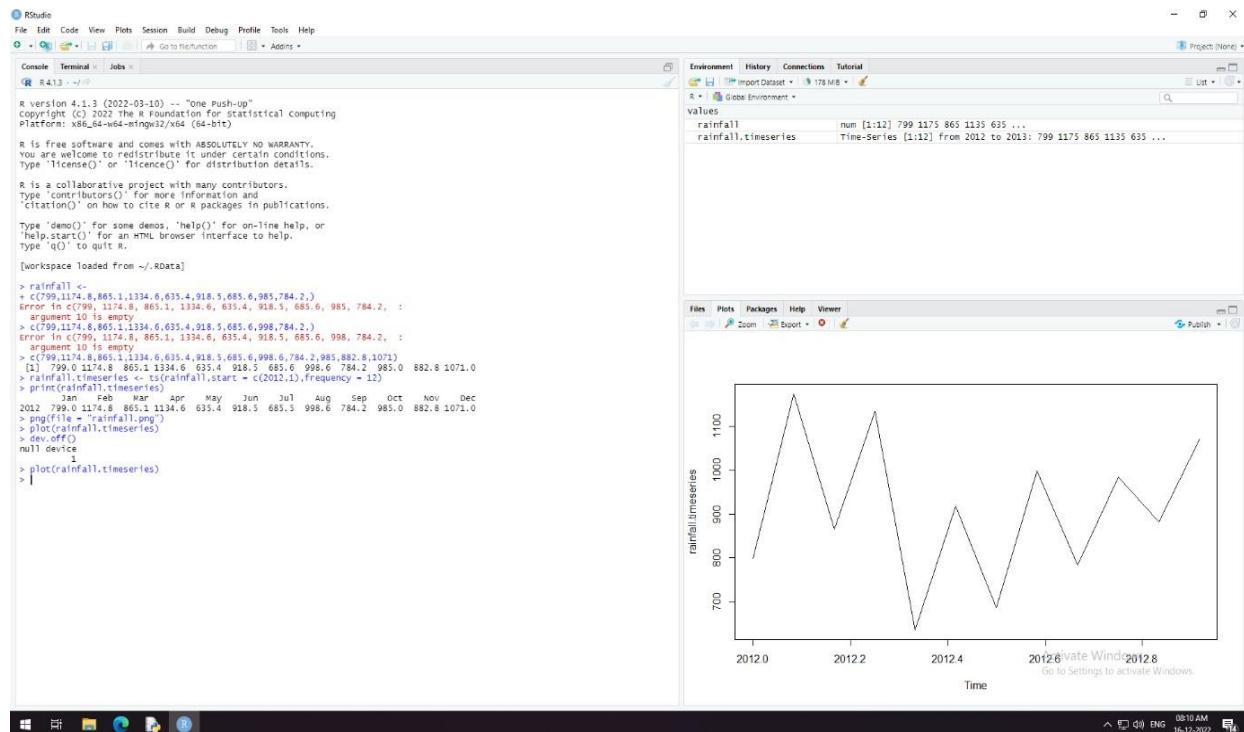
```
# Plot a graph of the time series.
```

```
plot(rainfall.timeseries)
```

```
# Save the file.
```

```
dev.off()
```

Output



Practical 9

Implementation of Decision Tree using R Tool.

Install.packages("party") - The package "party" has the function ctree() which is used to create and analyze decision tree

Syntax:

The basic syntax for creating a decision tree in R is – ctree(formula, data)

Input Data:

We will use the R in-built data set named readingSkills to create a decision tree. It describes the score of someone's readingSkills if we know the variables "age", "shoesize", "score" and whether the person is a native speaker or not.

```
# Load the party package. It will automatically load other dependent packages.
```

```
library(party)
```

```
# Print some records from data set readingSkills.
```

```
print(head(readingSkills))
```

We will use the ctree() function to create the decision tree and see its graph.

```
# Load the party package. It will automatically load other dependent packages.
```

```
library(party)
```

```
# Create the input data frame.
```

```
input.dat <- readingSkills[c(1:105),]
```

```
# Give the chart file a name.

png(file = "decision_tree.png")

# Create the tree.

output.tree <- ctree(nativeSpeaker ~ age + shoeSize + score, data = input.dat)

# Plot the tree.

plot(output.tree)

# Save the file.

dev.off
```

Output:

