Experiment No. 1

AIM:

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

Name: Ankit Singh Chauhan

Roll no: 64

Class: T.Y.BSc.IT

Subject: Business Intelligence

Marks\Grade:

Sign:

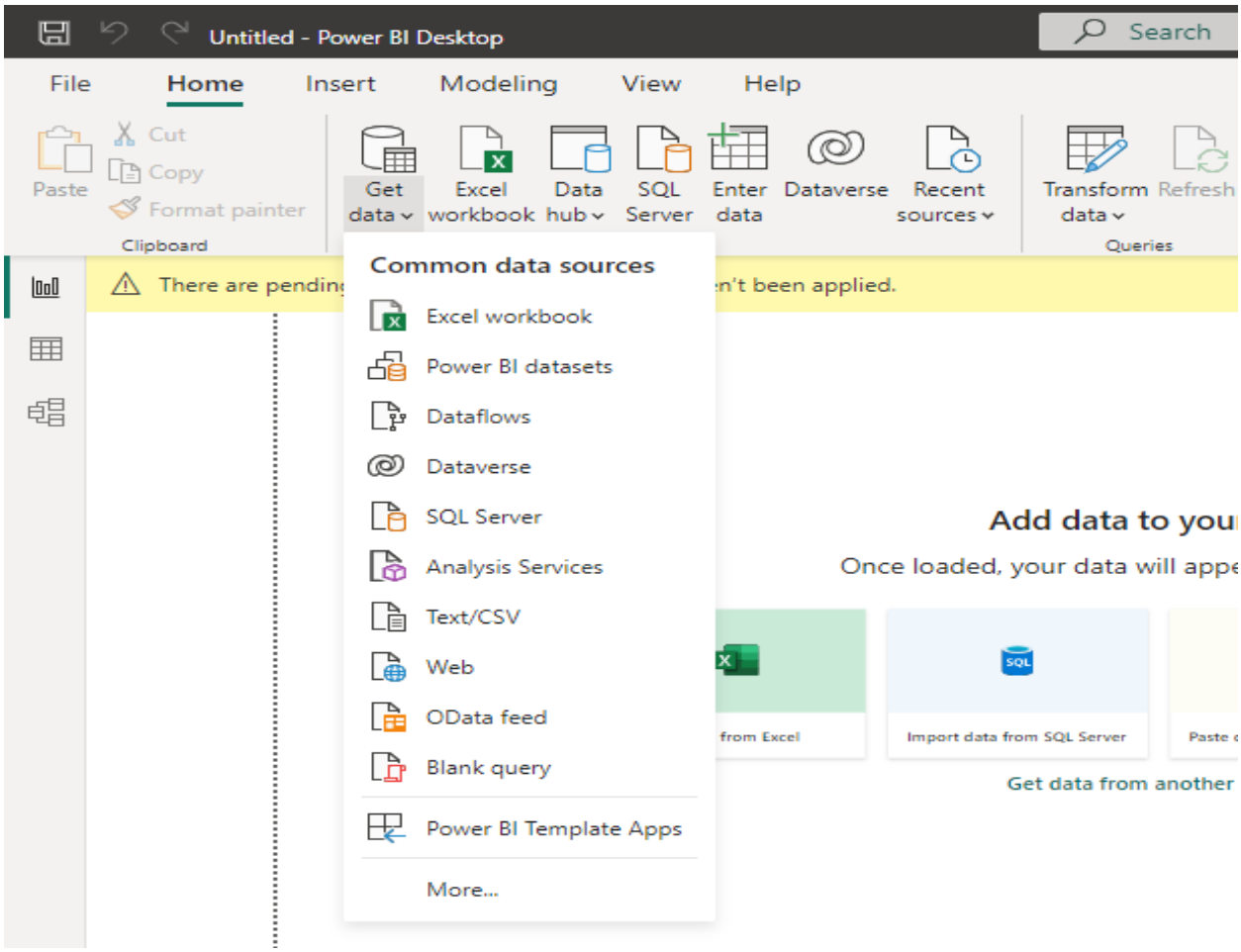
Name: Ankit Singh Chauhan
Roll No. 64

OUTPUT:

Importing Excel Data

1) Launch Power BI Desktop.

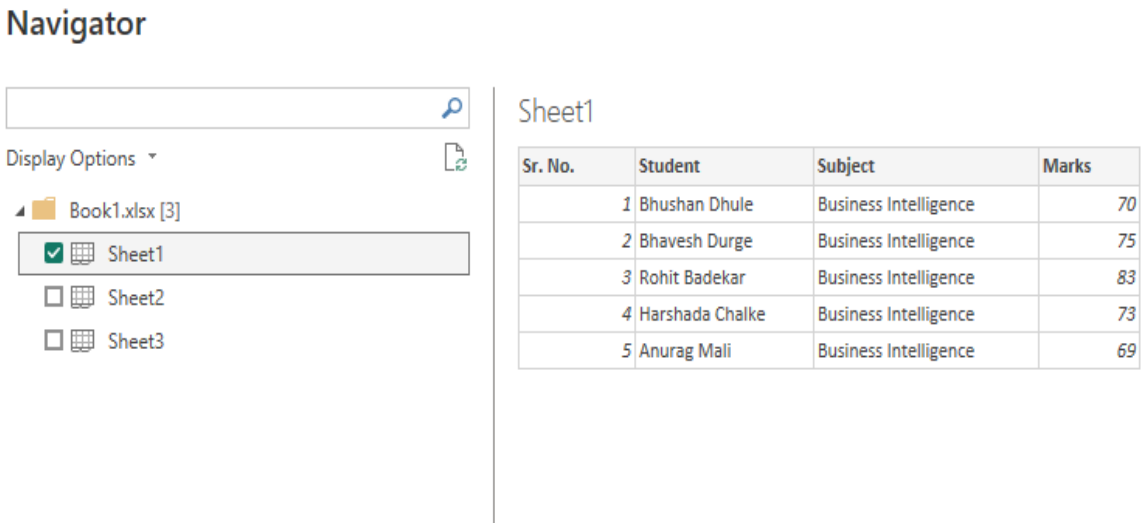
2) From the Home ribbon, select Get Data. Excel is one of the Most Common data connections, so you can select it directly from the Get Data menu.



3) If you select the Get Data button directly, you can also select FIle > Excel and select Connect.

4) In the Open File dialog box, select the Products.xlsx file.

5) In the Navigator pane, select the Products table and then select Edit.



Sr. No.	Student	Subject	Marks
1	Bhushan Dhule	Business Intelligence	70
2	Bhavesh Durge	Business Intelligence	75
3	Rohit Badekar	Business Intelligence	83
4	Harshada Chalke	Business Intelligence	73
5	Anurag Mali	Business Intelligence	69

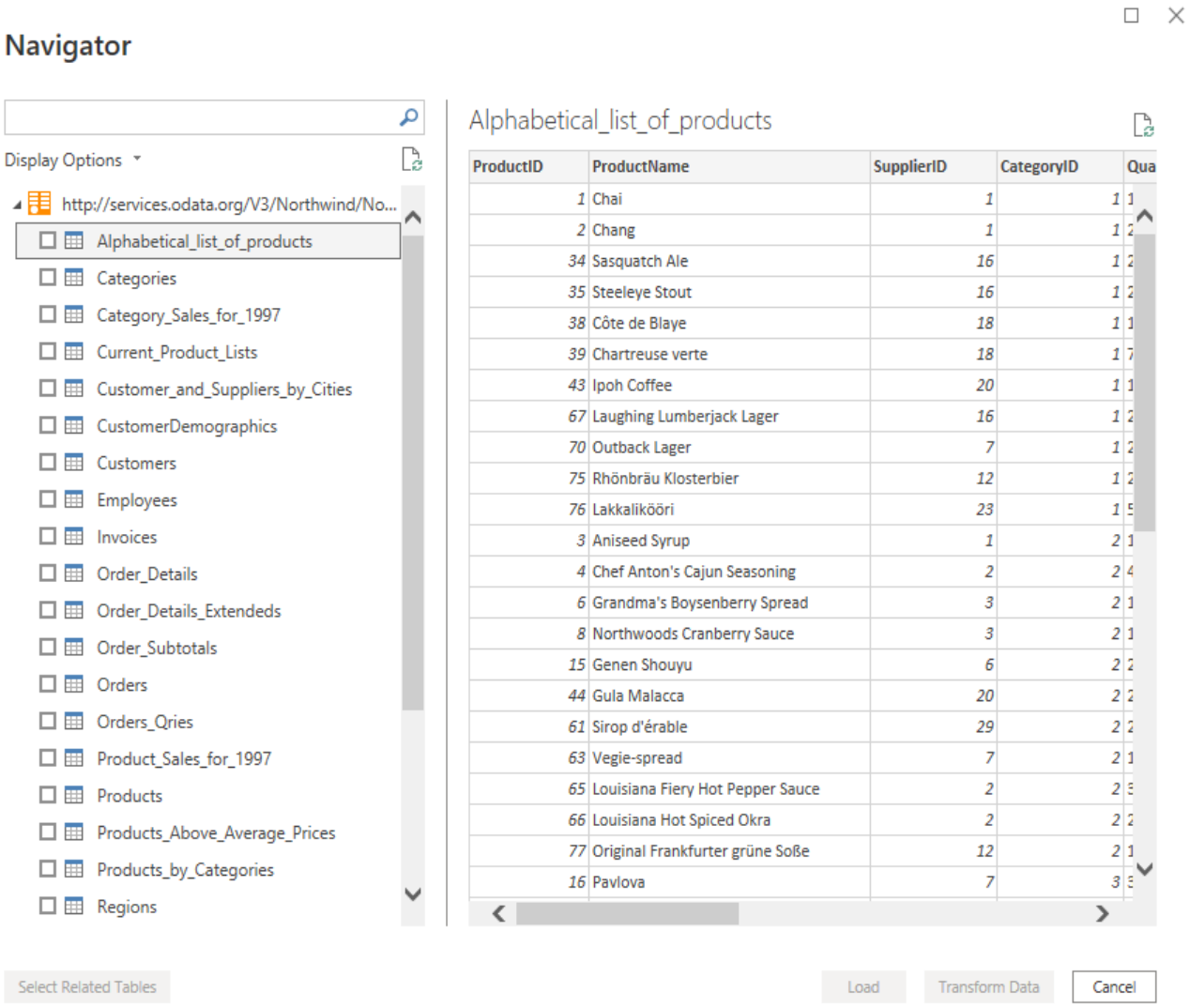
Importing Data from OData Feed

In this task, you'll bring in order data. This step represents connecting to a sales system. You import data into Power BI Desktop from the sample Northwind OData feed at the following URL, which you can copy (and then paste) in the steps below:
http://services.odata.org/V3/Northwind/Northwind.svc/

Connect to an OData feed:

1) From the Home ribbon tab in Query Editor, select Get Data.
2) Browse to the OData Feed data source.
3) In the OData Feed dialog box, paste the URL for the Northwind OData feed.
4) Select OK.
5) In the Navigator pane, select the Orders table, and then select Edit.

Name: Ankit Singh Chauhan
Roll No. 64



Experiment No.2

Aim:

Perform the Extraction Transformation and Loading (ETL) process to construct the database in Power BI.

Name: Ankit Singh Chauhan

Roll no: 64

Class: T.Y.BSc.IT

Subject: Business Intelligence

Marks\Grade:

Sign:

Name: Ankit Singh Chauhan

Roll No: 64

ETL Process in Power BI

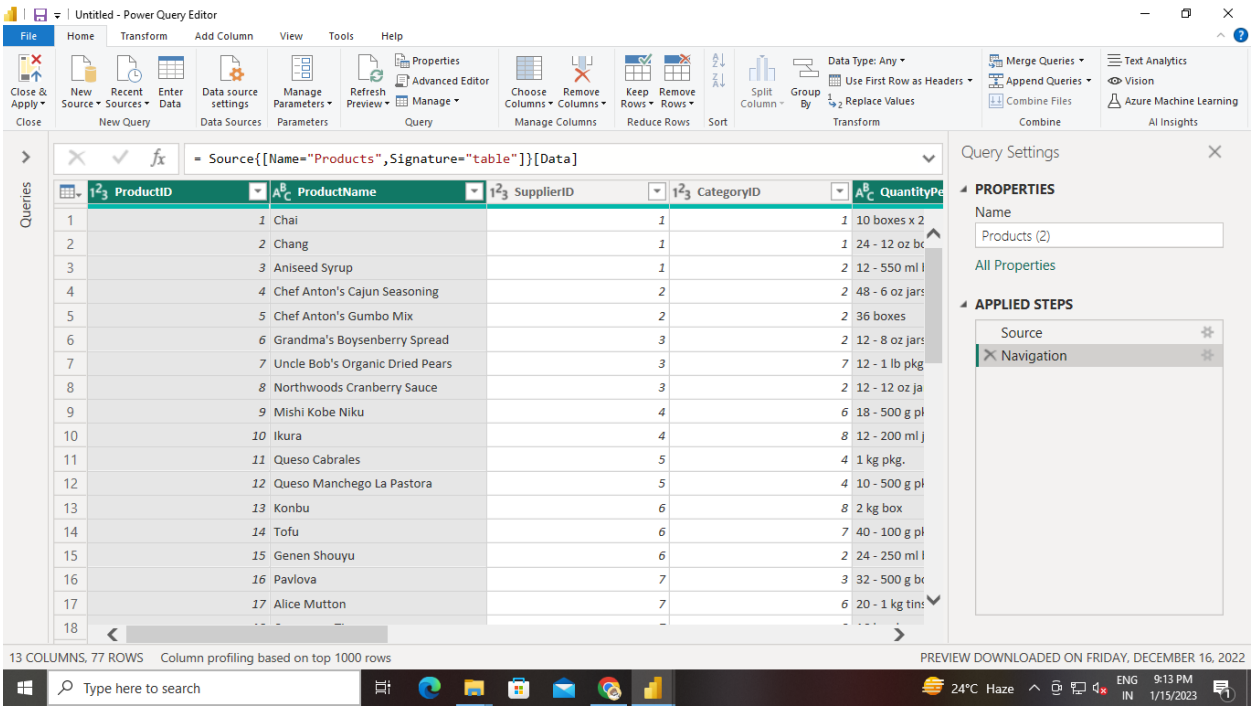
1) Remove other columns to only display columns of interest

In this step you 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. The following steps are performed in Query Editor.

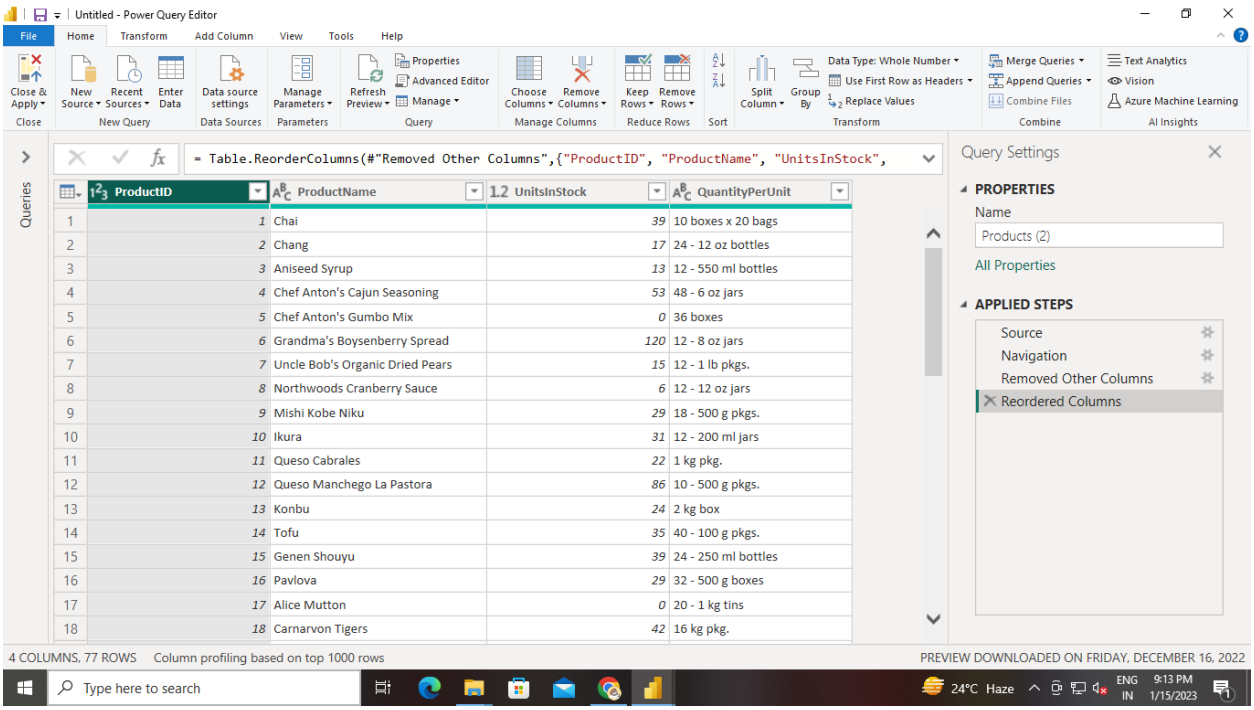
1. In Query Editor, select the ProductID, ProductName, QuantityPerUnit, and UnitsInStock columns (use Ctrl+Click to select more than one column, or Shift+Click to select columns that are beside each other).

2. Select Remove Columns > Remove Other Columns from the ribbon, or right-click on a column header and click Remove Other Columns.



Name: Ankit Singh Chauhan

Roll No: 64



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

3. Change the data type of the UnitsInStock column

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, so in this step you confirm the UnitsInStock column's datatype is Whole Number.

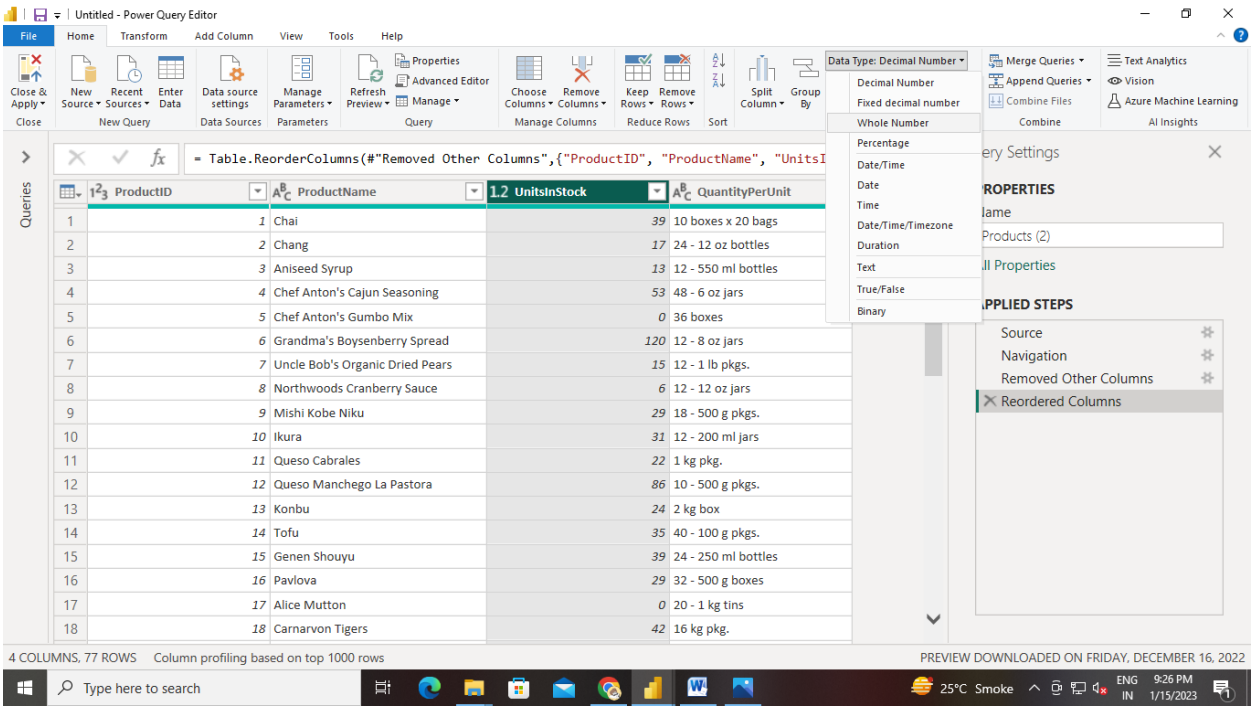
1. Select the UnitsInStock column.

2. Select the Data Type drop-down button in the Home ribbon.

3. If not already a Whole Number, select Whole Number for data type from the drop down (the Data Type: button also displays the data type for the current selection).

Name: Ankit Singh Chauhan

Roll No: 64



3. Expand the Order_Details 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 multiples tables (such as a relational database) you can use these references to build up your query

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.

This is a representation of the data in these tables:

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.

1. In the Query View, scroll to the Order_Details column.

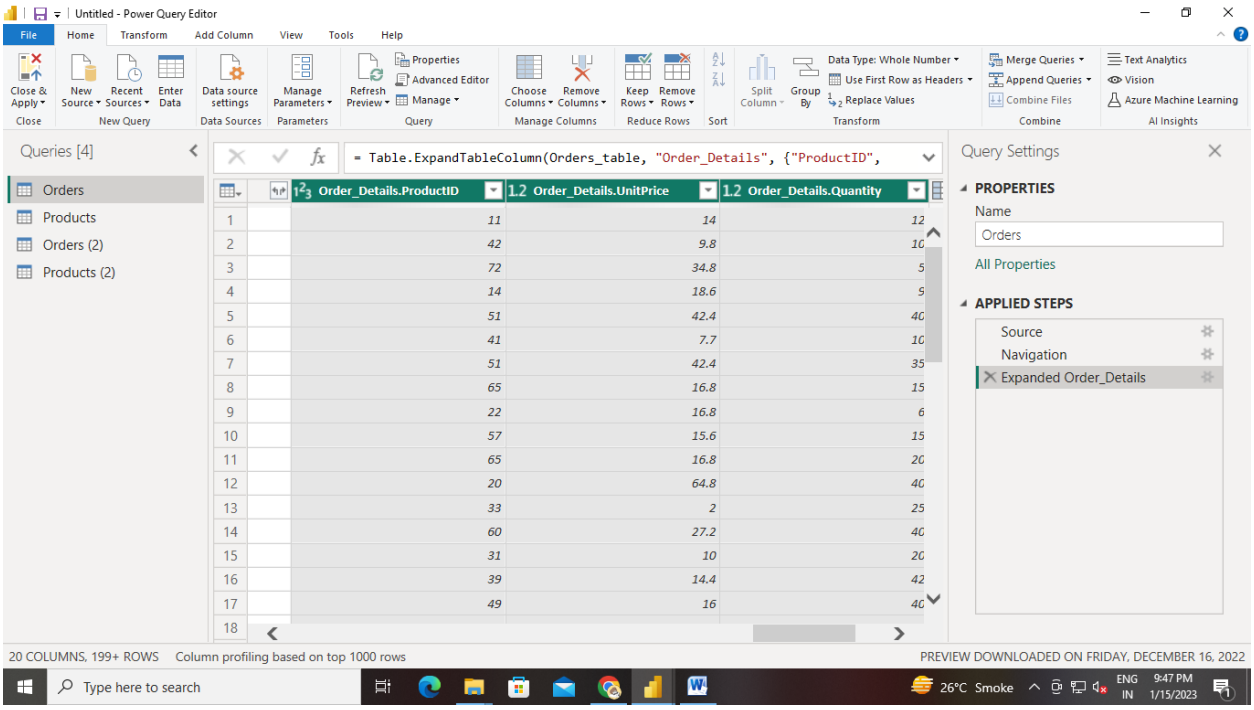
2. In the Order_Details column, select the expand icon ( ).

3. In the Expand drop-down:

a. Select (Select All Columns) to clear all columns.

b. Select ProductID, UnitPrice, and Quantity.

c. Click OK.



4. Calculate the line total for each Order_Details row

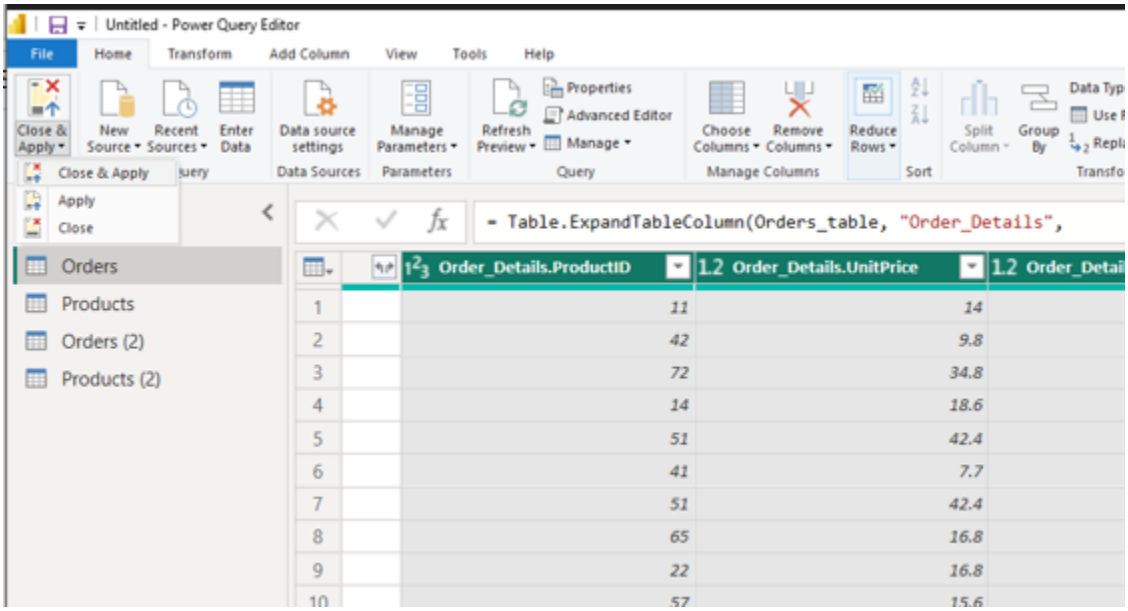
Power BI Desktop lets you to create calculations based on the columns you are importing, so you can enrich the data that you connect to. In this step, you create a Custom Column to calculate the line total for each Order_Details row.

Calculate the line total for each Order_Details row:

1. In the Add Column ribbon tab, click Add Custom Column.

Name: Ankit Singh Chauhan

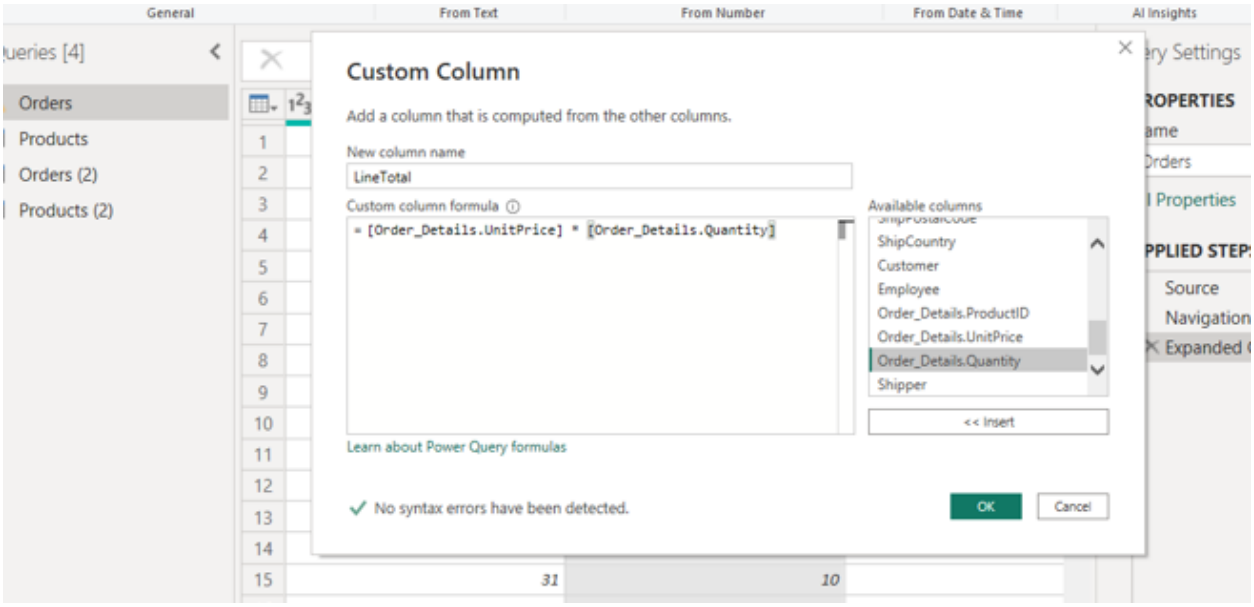
Roll No: 64



2. In the Add Custom Column dialog box, in the Custom Column Formula textbox, enter [Order_Details.UnitPrice] * [Order_Details.Quantity].

3. In the New column name textbox, enter LineTotal.

4. Click OK.



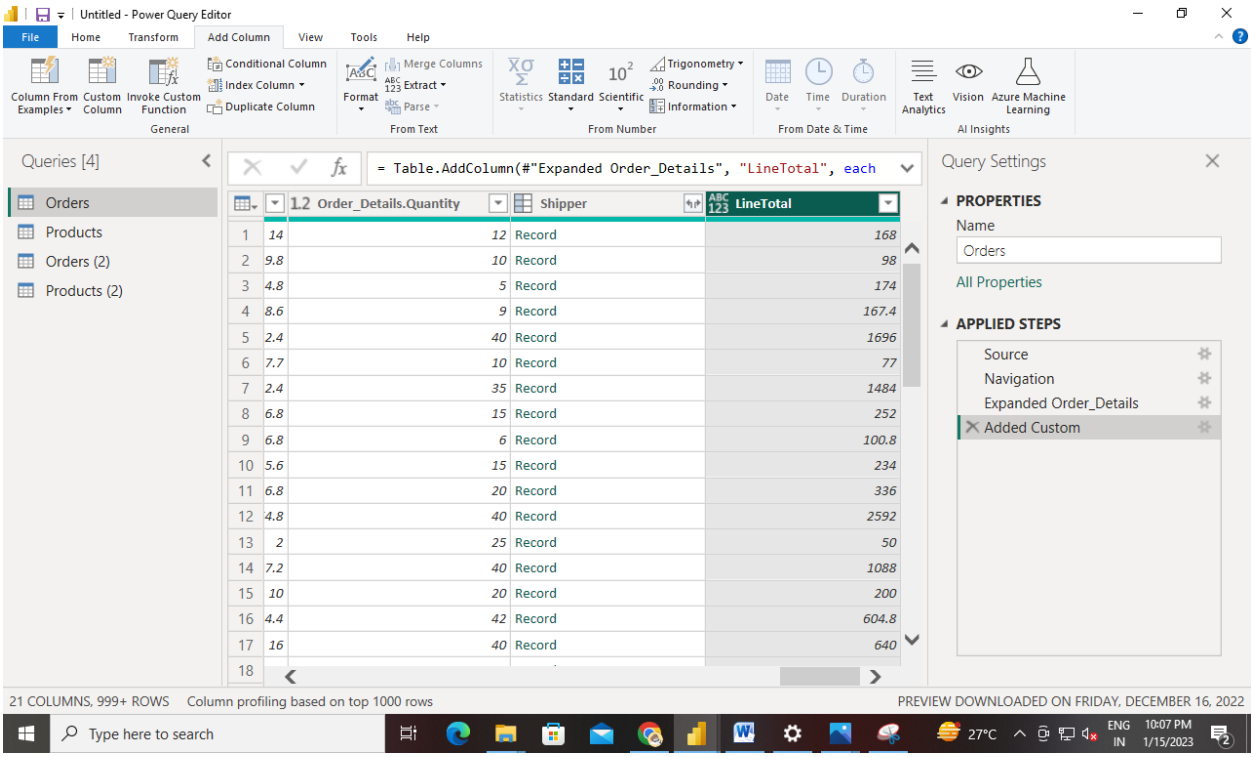
Name: Ankit Singh Chauhan

Roll No: 64

5. Rename and reorder columns in the query

In this step you finish making the model easy to work with when creating reports, by renaming the final columns and changing their order.

1. In Query Editor, drag the LineTotal column to the left, after ShipCountry.



2. Remove the Order_Details. prefix from the Order_Details.ProductID, Order_Details.UnitPrice and Order_Details.Quantity columns, by double-clicking on each column header, and then deleting that text from the column name.

6. Combine the Products and Total Sales queries 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.

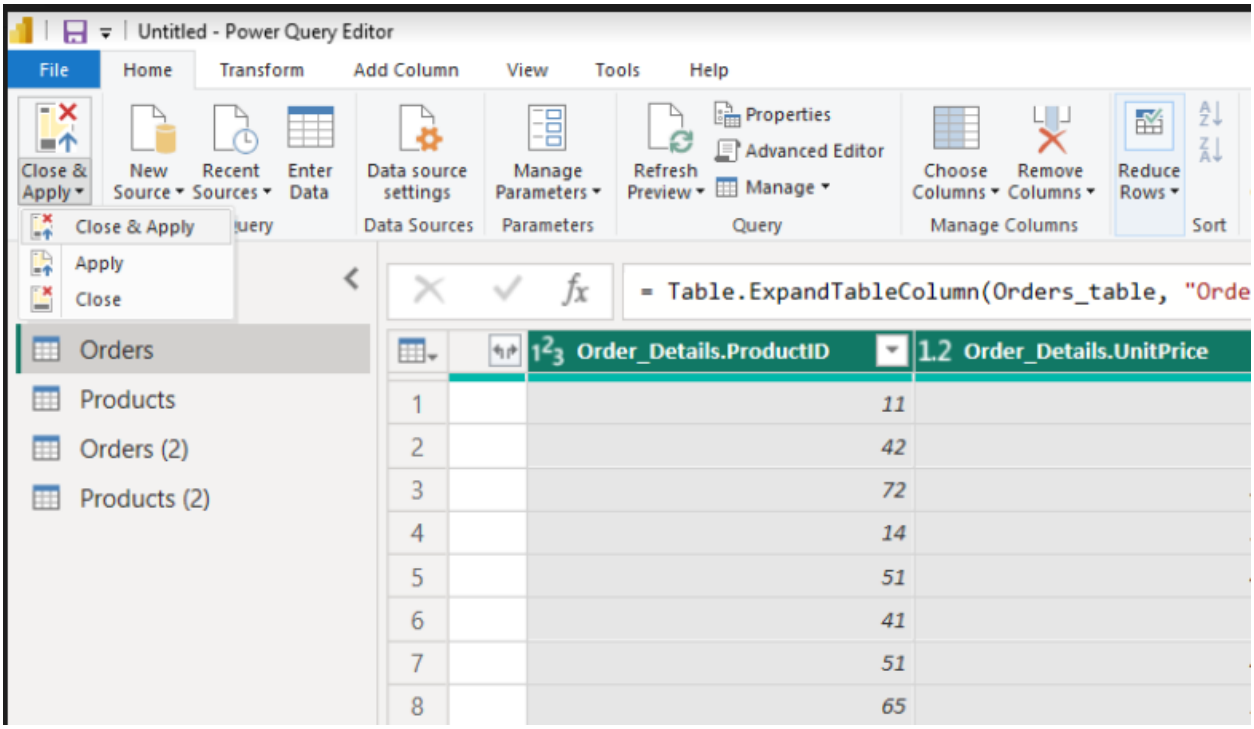
In this task, you confirm that a relationship is established in Power BI Desktop between the Products and Total Sales queries

Name: Ankit Singh Chauhan

Roll No: 64

Step 1: Confirm the relationship between Products and Total Sales

1. First, we need to load the model that we created in Query Editor into Power BI Desktop. From the Home ribbon of Query Editor, select Close & Load

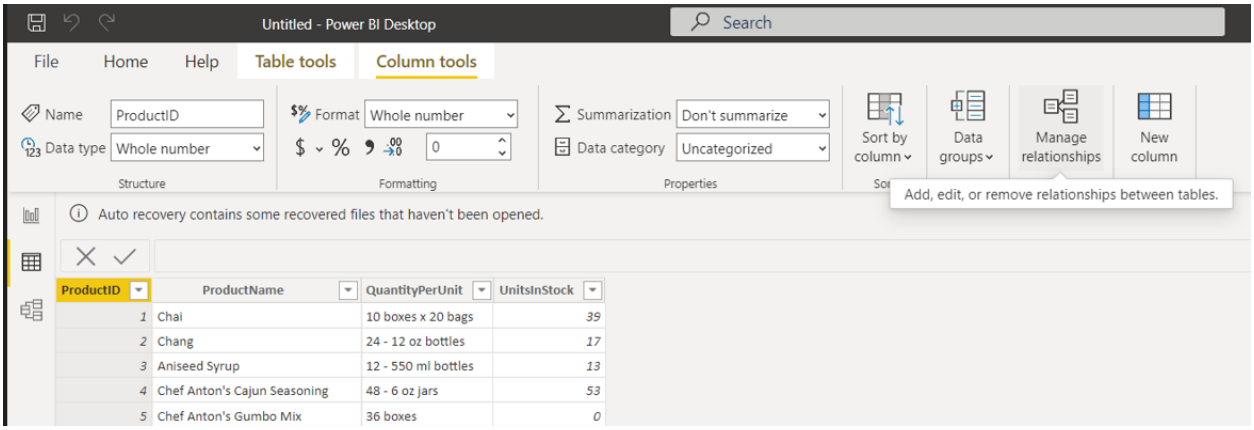


2. Power BI Desktop loads the data from the two queries.

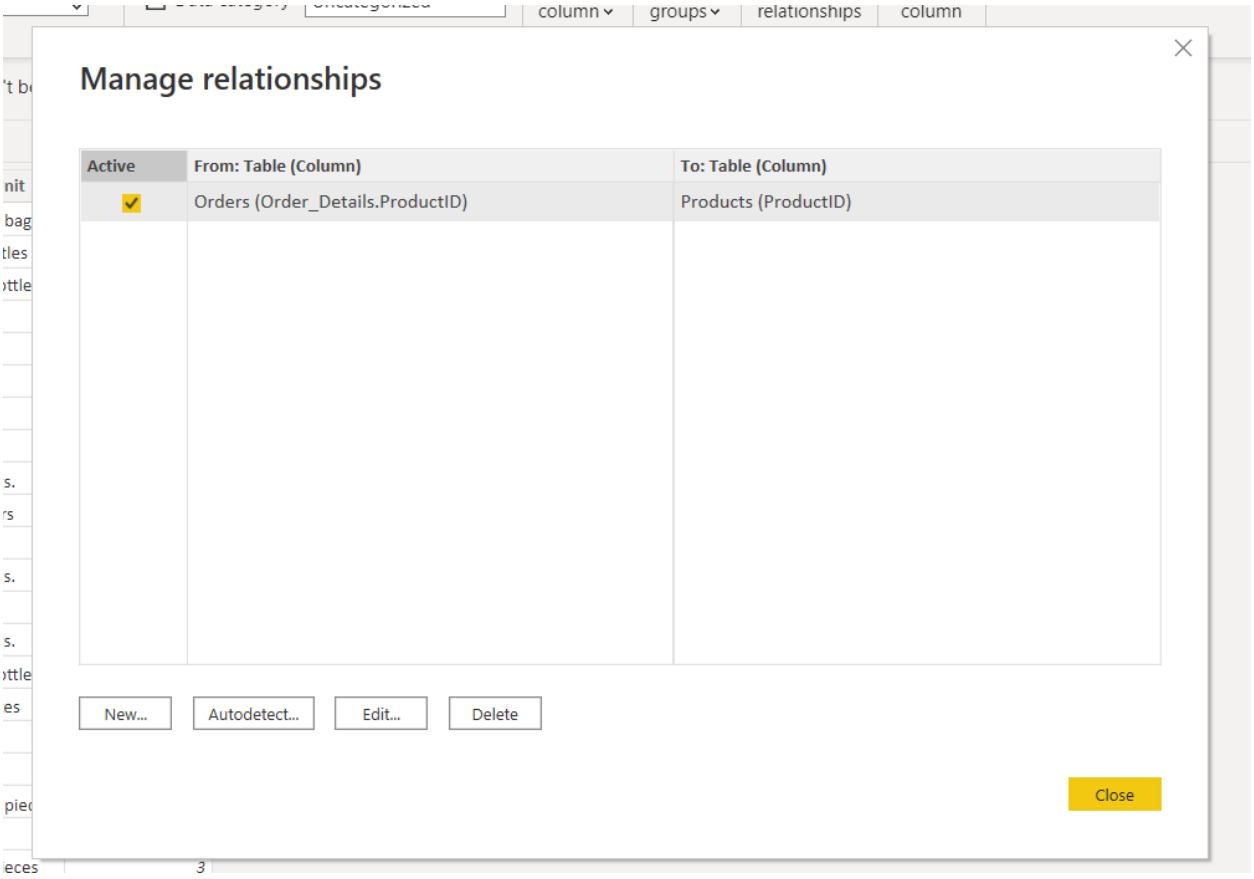
3. Once the data is loaded, select the Manage Relationships button Home ribbon.

Name: Ankit Singh Chauhan

Roll No: 64



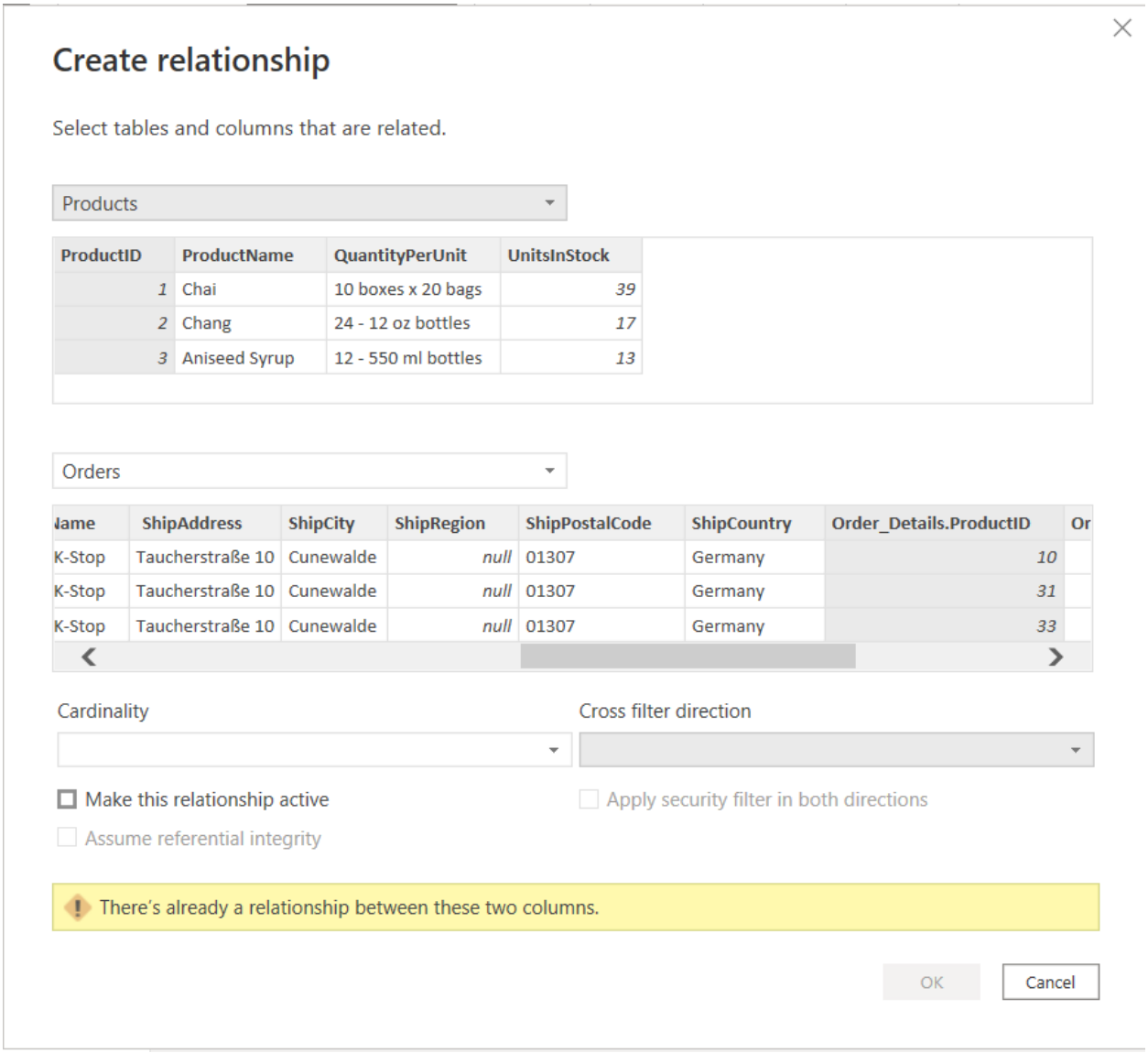
4. Select the new button



Name: Ankit Singh Chauhan

Roll No: 64

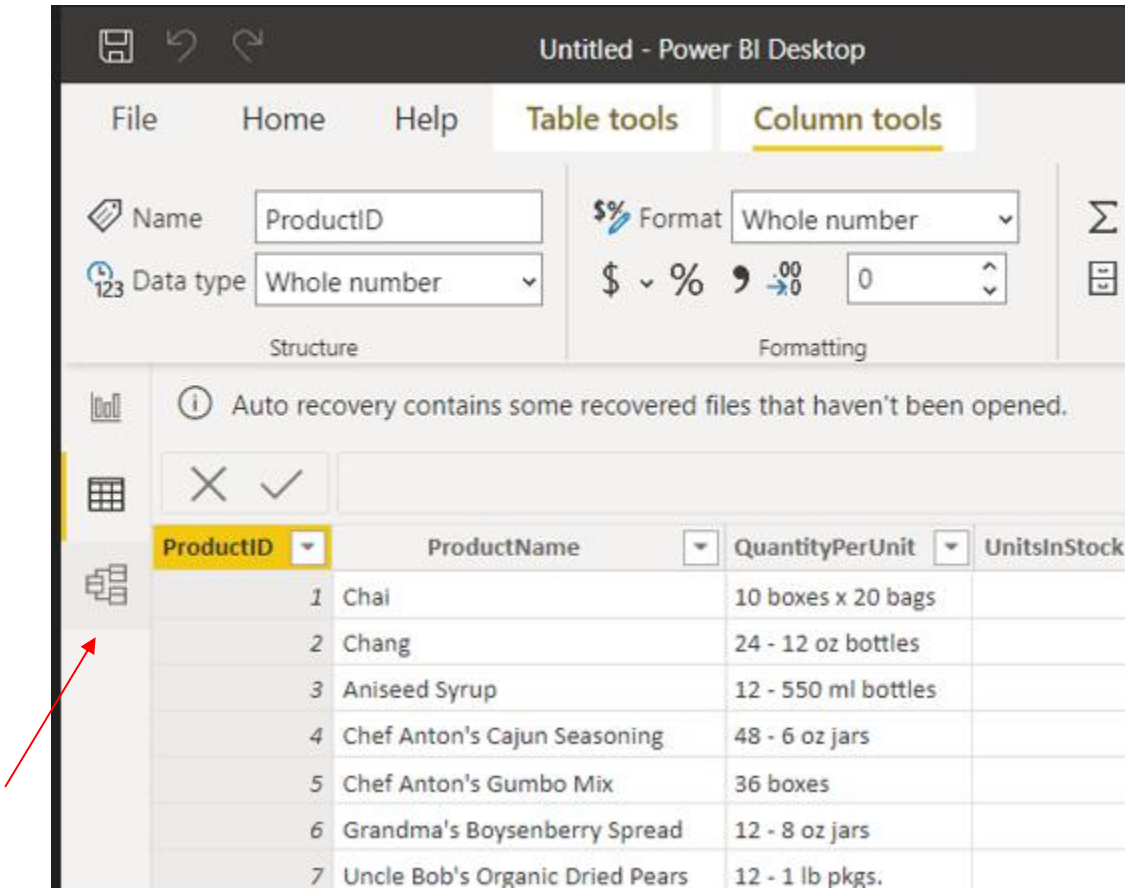
5.When we attempt to create the relationship, we see that one already exists! As shown in the Create Relationship dialog (by the shaded columns), the ProductsID fields in each query already have an established relationship.



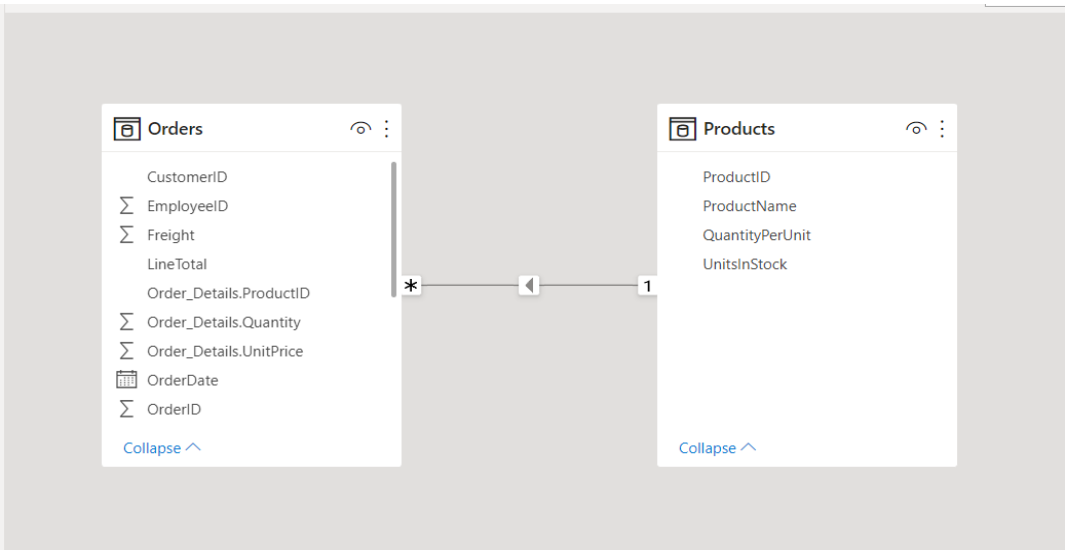
Name: Ankit Singh Chauhan

Roll No: 64

5. Select cancel and then select relationship view in power bi desktop



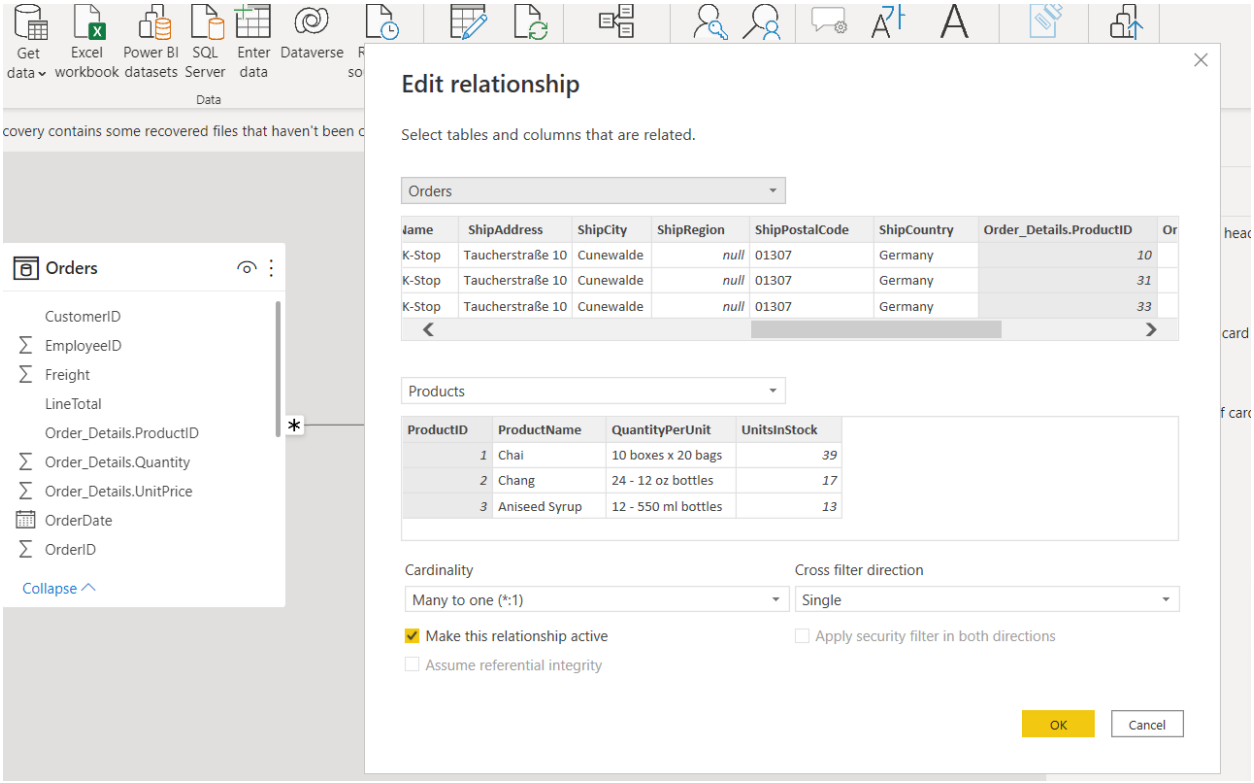
6. We see the following, which visualizes the relationship between the queries.



Name: Ankit Singh Chauhan

Roll No: 64

7. When you double-click the arrow on the line that connects the to queries, an Edit Relationship dialog appears.



8. No need to make any changes, so we'll just select Cancel to close the Edit Relationship dialog.

Experiment No. 3

AIM:

A) Data Visualization from ETL Process.

Name: Ankit Singh Chauhan

Roll no: 64

Class: T.Y.BSc.IT

Subject: Business Intelligence

Marks\Grade:

Sign:

Name: Ankit Singh Chauhan
Roll No. 64

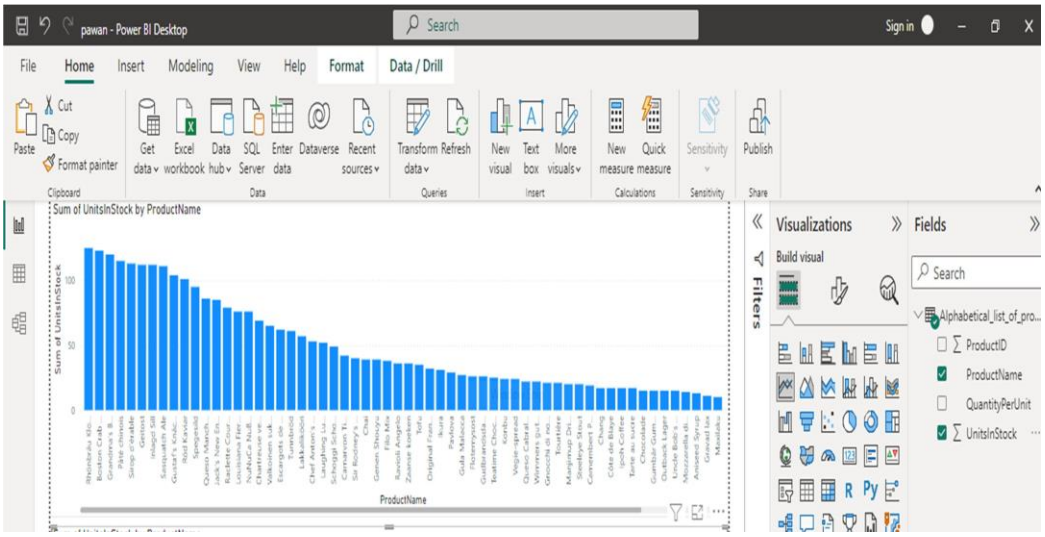
AIM:

A) Data Visualization from ETL Process.

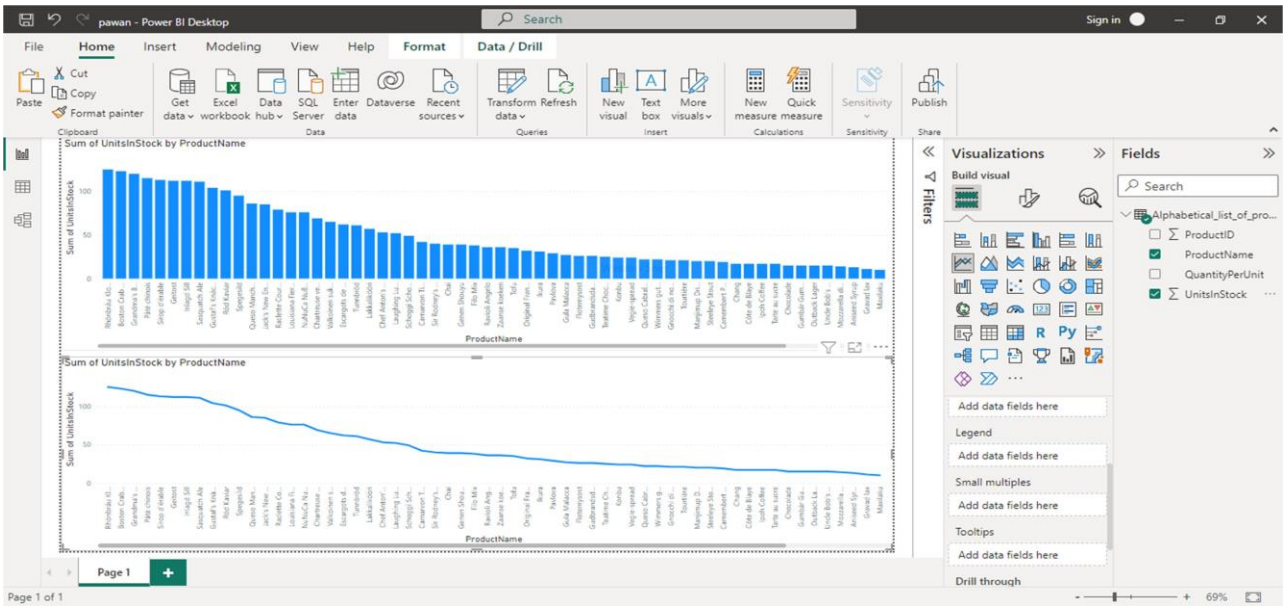
Power BI Desktop lets you create a variety of visualizations to gain insights from your data. You can build reports with multiple pages and each page can have multiple visuals. You can interact with your visualizations to help analyze and understand your data in this task; you create a report based on the data previously loaded. You use the Fields pane to select the columns from which you create the visualizations.

Step 1: Create charts showing Units in Stock by Product and Total Sales by Year.

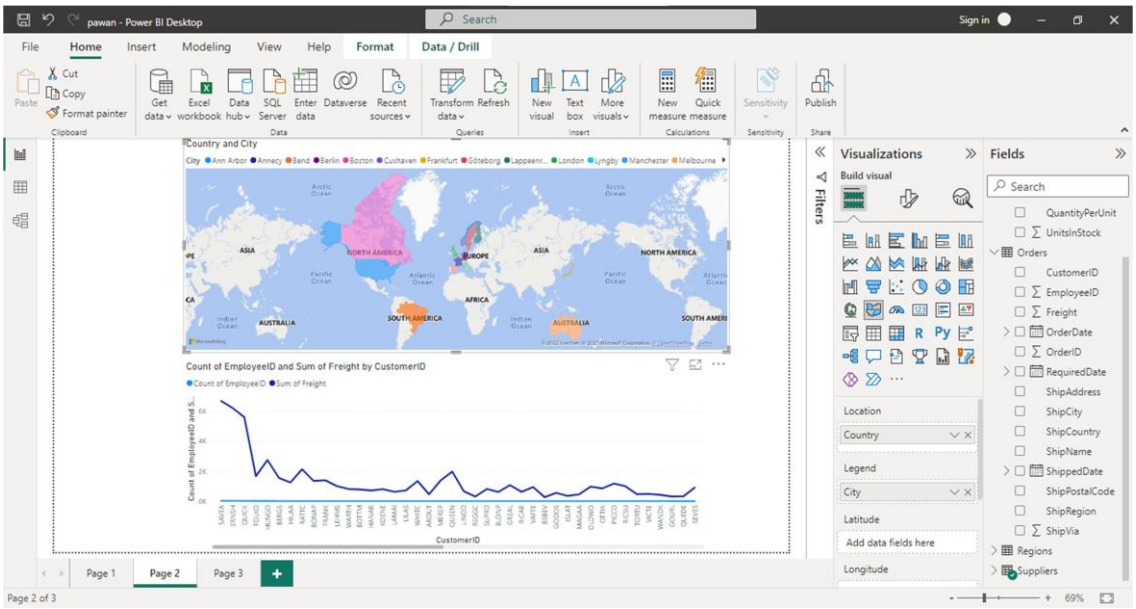
1. Drag Units in Stock from the Field pane (the Fields pane is along the right of the screen) onto a blank space on the canvas. Table visualization is created. Next, drag Product Name to the Axis box, found in the bottom half of the Visualizations pane. Then we then select Sort By > Units in Stock using the skittles in the top right corner of the visualization.



2. Drag Order Date to the canvas beneath the first chart, then drag Line Total (again, from the Fields pane) onto the visual, then select Line Chart. The following visualization is created.



3. Next, drag Ship Country to a space on the canvas in the top right. Because you selected a geographic field, a map was created automatically. Now drag Line Total to the Values field; the circles on the map for each country are now relative in size to the Line Total for orders shipped to that country.

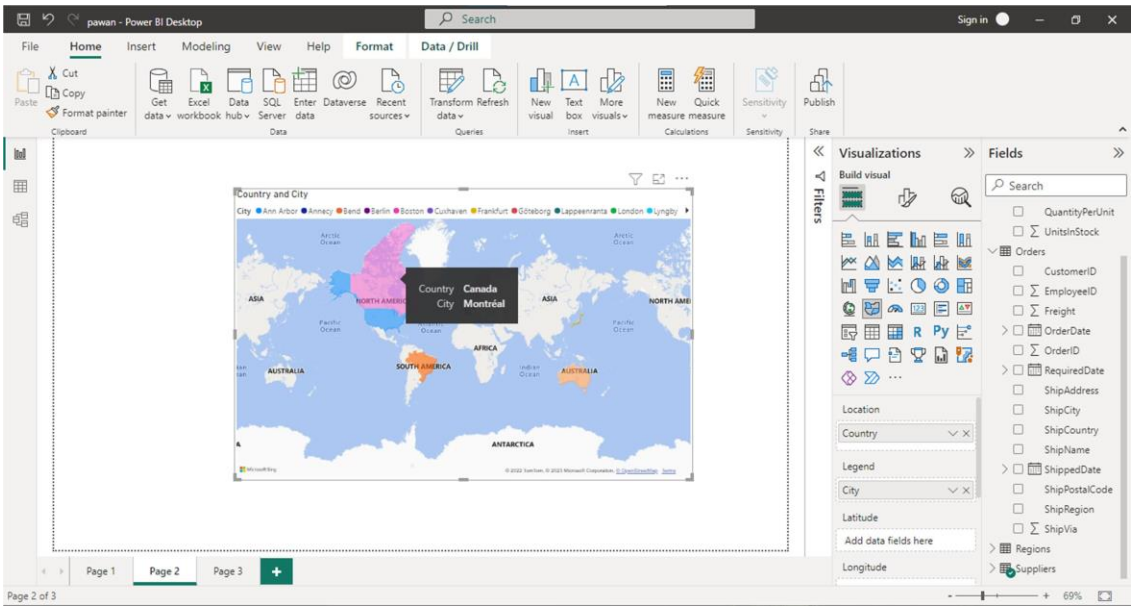


Name: Ankit Singh Chauhan
Roll No. 64

Step 2: Interact with your report visuals to analyze further

Power BI Desktop lets you interact with visuals that cross-highlight and filter each other to uncover further trends.

1. Click on the light blue circle centered in Canada. Note how the other visuals are filtered to show Stock (Ship Country) and Total Orders (Line Total) just for Canada.



Experiment No.5

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

Name: Ankit Singh Chauhan

Roll no: 64

Class: T.Y.BSc.IT

Subject: Business Intelligence

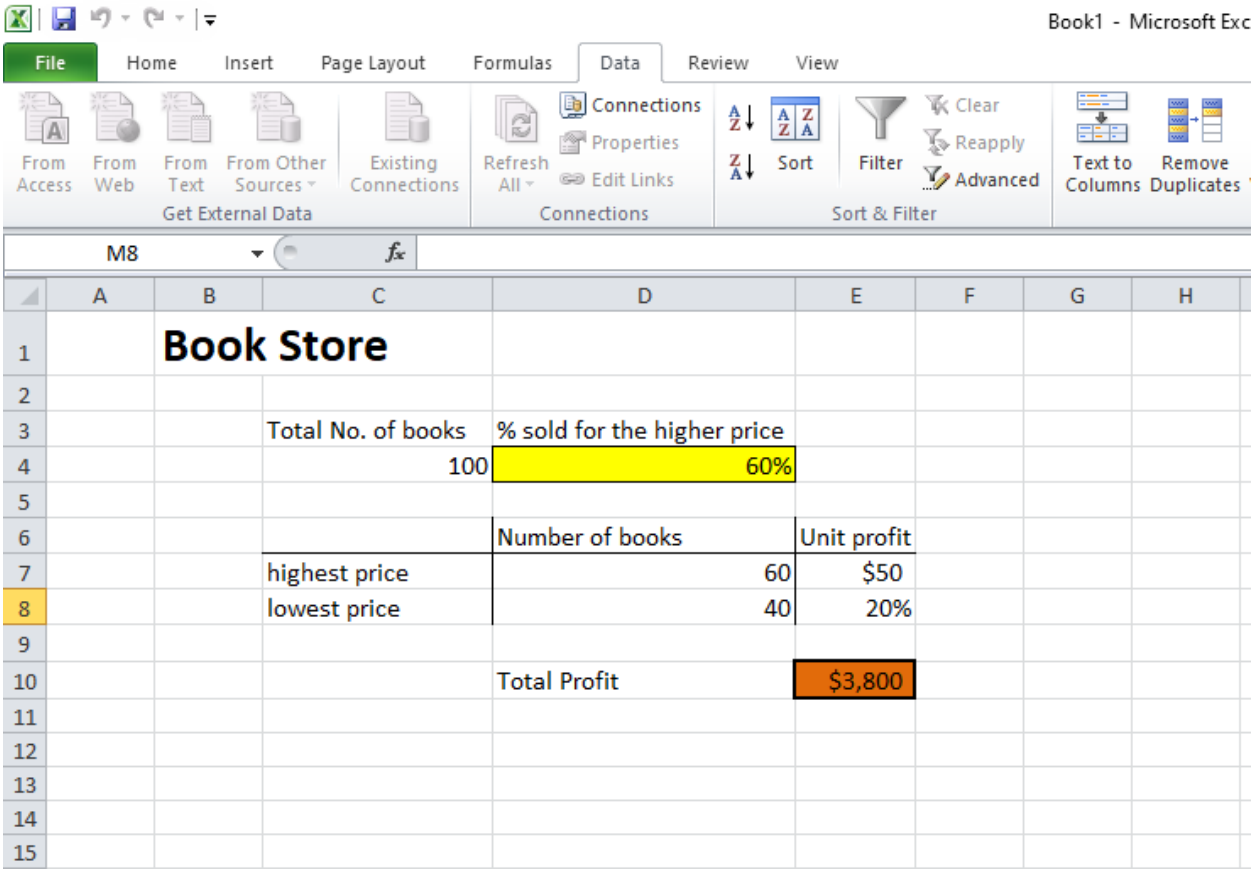
Marks\Grade:

Sign:

Name: Ankit Singh Chauhan

Roll No: 64

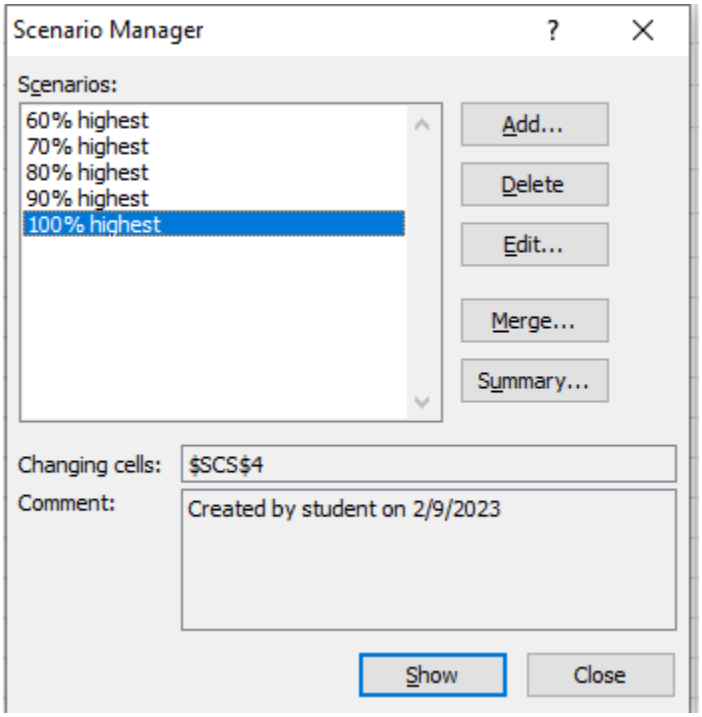
A bookstore and have 100 books in storage. You sell a certain % for the highest price of $50 and a certain % for the lower price of $20.



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.

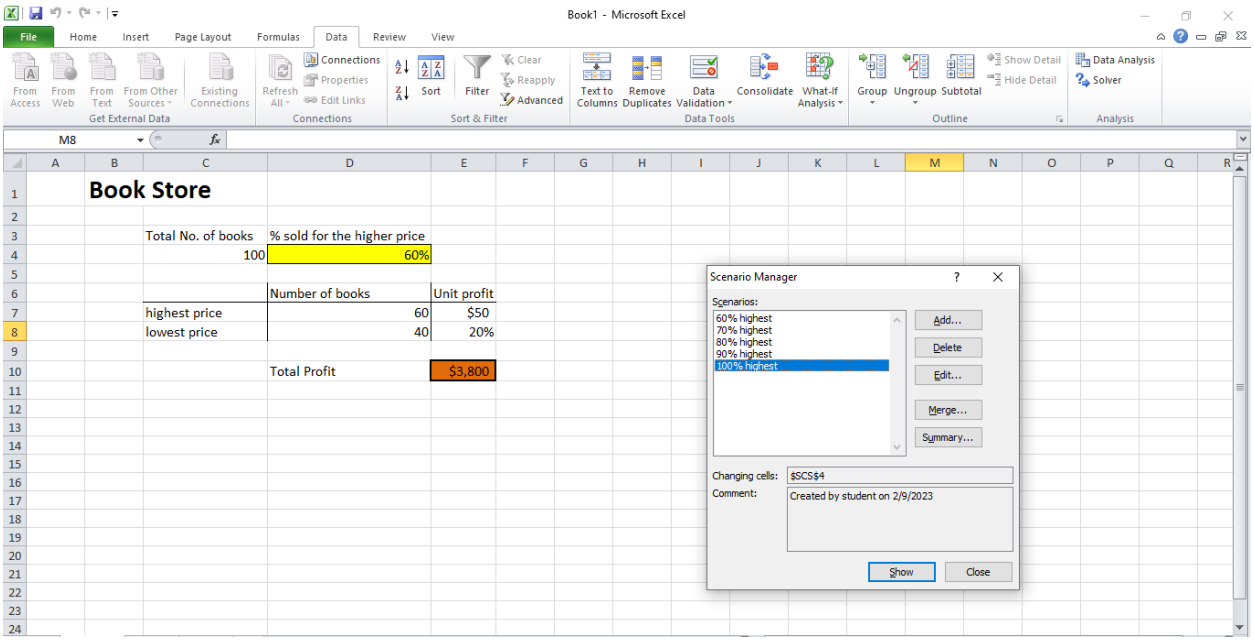
1. On the Data tab, in the Forecast group, click What-If Analysis.
2. Click Scenario Manager.
   The Scenario Manager Dialog box appears.
3. Add a scenario by clicking on Add.
4. Type a name (60% highest), select cell C4 (% sold for the highest price) for the Changing cells and click on OK

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



6. Next, add 4 other scenarios (70%, 80%, 90% and 100%).

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



Experiment No. 6

Aim: Implementation of Classification algorithm in R Programming.

Name: Ankit Singh Chauhan

Roll no: 64

Class: T.Y.BSc.IT

Subject: Business Intelligence

Marks\Grade:

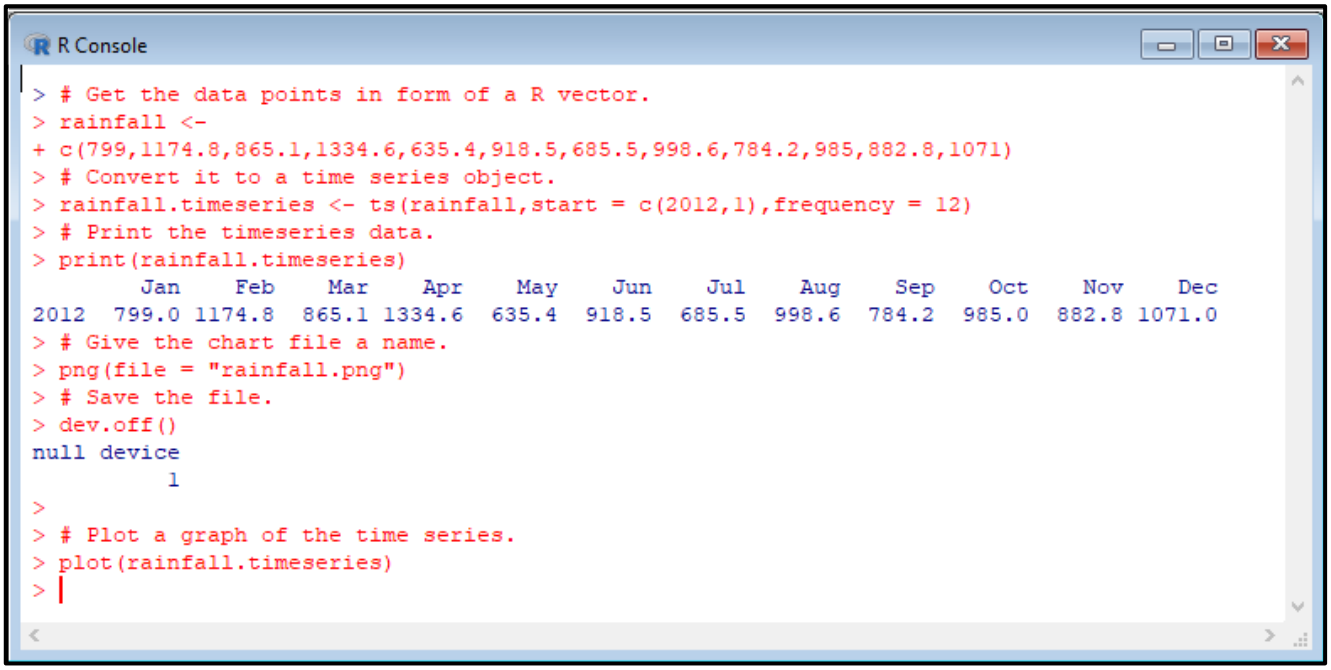
Sign:

Name: Ankit Singh Chauhan
Roll No: 64

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.

Code:

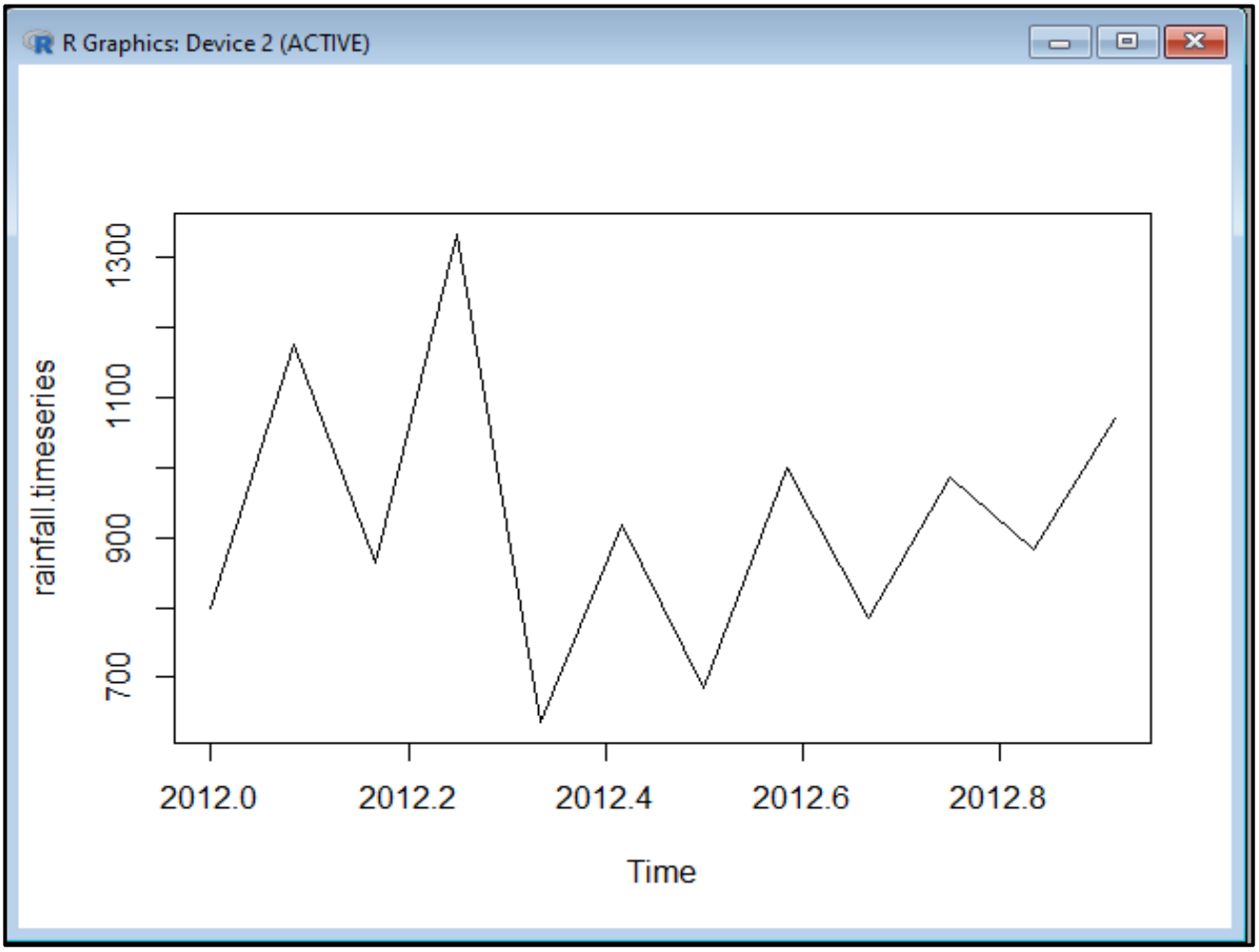
```
# Get the data points in form of a R vector.
rainfall <-
c(799,1174.8,865.1,1334.6,635.4,918.5,685.5,998.6,784.2,985,882.8,1071)
# Convert it to a time series object.
rainfall.timeseries <- ts(rainfall,start = c(2012,1),frequency = 12)
# Print the timeseries data.
print(rainfall.timeseries)
# Give the chart file a name.
png(file = "rainfall.png")
# Plot a graph of the time series.
plot(rainfall.timeseries)
# Save the file.
dev.off()
```



Name: Ankit Singh Chauhan
Roll No: 64

Output:

```
> print(rainfall.timeseries)
        Jan    Feb   Mar    Apr   May   Jun   Jul   Aug   Sep   Oct   Nov    Dec
2012 799.0 1174.8 865.1 1334.6 635.4 918.5 685.5 998.6 784.2 985.0 882.8 1071.0
```



Experiment No.7

Aim: Practical Implementation of Decision Tree using R Tool.

Name: Ankit Singh Chauhan

Roll no: 64

Class: T.Y.BSc.IT

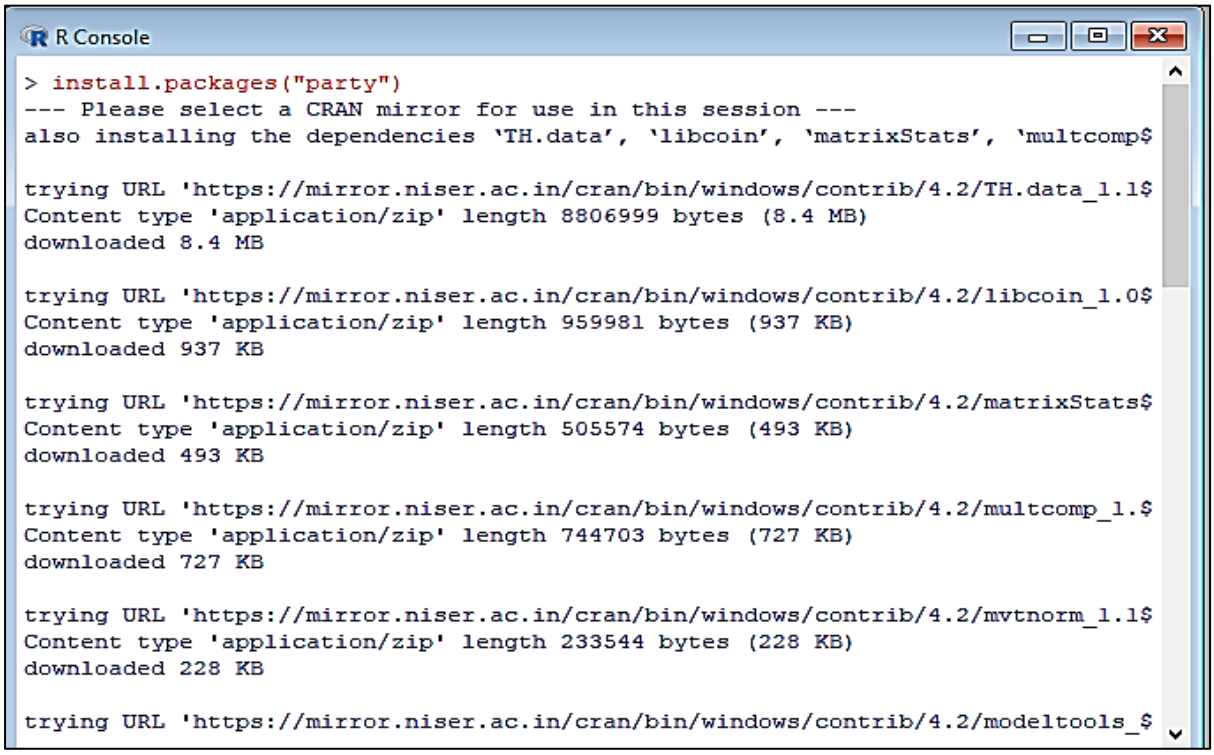
Subject: Business Intelligence

Marks\Grade:

Sign:

install.packages("party")

The package "party" has the function **ctree**() which is used to create and analyze decison 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.

Here is the sample data.

Load the party package. It will automatically load other

dependent packages.

library(party)

```
> library(party)
Loading required package: grid
Loading required package: mvtnorm
Loading required package: modeltools
Loading required package: stats4
Loading required package: strucchange
Loading required package: zoo

Attaching package: 'zoo'

The following objects are masked from 'package:base':

    as.Date, as.Date.numeric

Loading required package: sandwich
>
```

Print some records from data set readingSkills.

print(head(readingSkills))

```
> print(head(readingSkills))
  nativeSpeaker age shoeSize    score
1           yes   5 24.83189 32.29385
2           yes   6 25.95238 36.63105
3            no  11 30.42170 49.60593
4           yes   7 28.66450 40.28456
5           yes  11 31.88207 55.46085
6           yes  10 30.07843 52.83124
>
```

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

```
  nativeSpeaker age shoeSize    score
1           yes   5 24.83189 32.29385
2           yes   6 25.95238 36.63105
3            no  11 30.42170 49.60593
4           yes   7 28.66450 40.28456
5           yes  11 31.88207 55.46085
6           yes  10 30.07843 52.83124
>
```

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

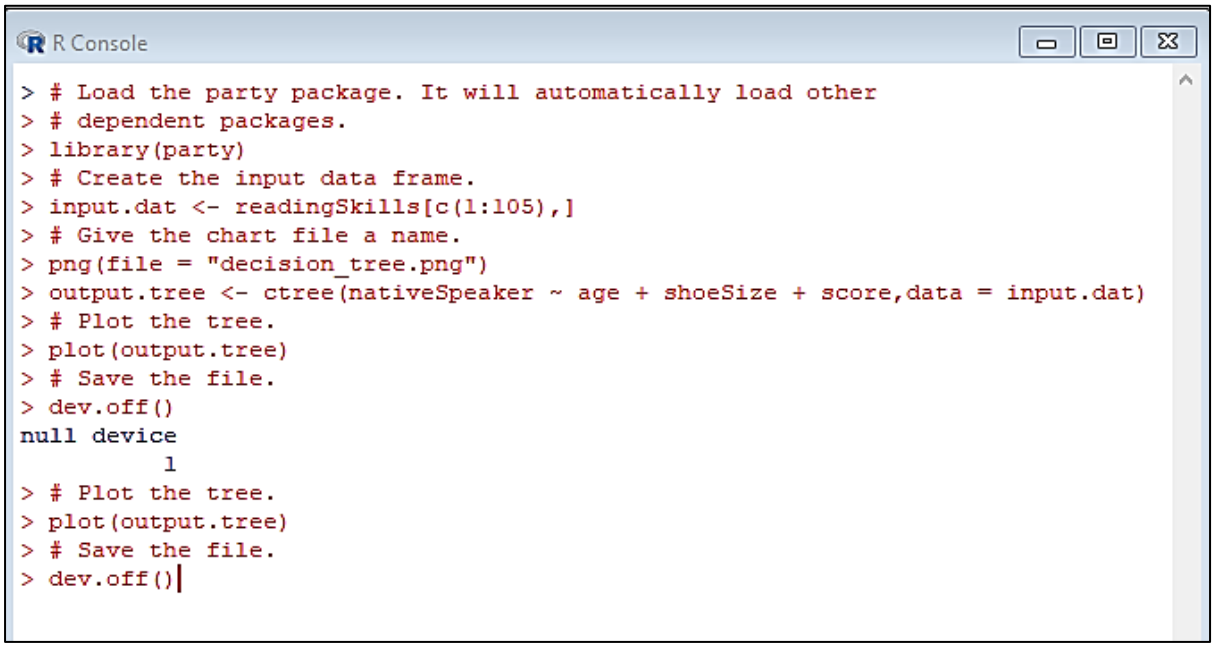
Load the party package. It will automatically load other

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:-

null device

Name: Ankit Singh Chauhan
Roll No: 64

1

Loading required package: methods

Loading required package: grid

Loading required package: mvtnorm

Loading required package: modeltools

Loading required package: stats4

Loading required package: strucchange

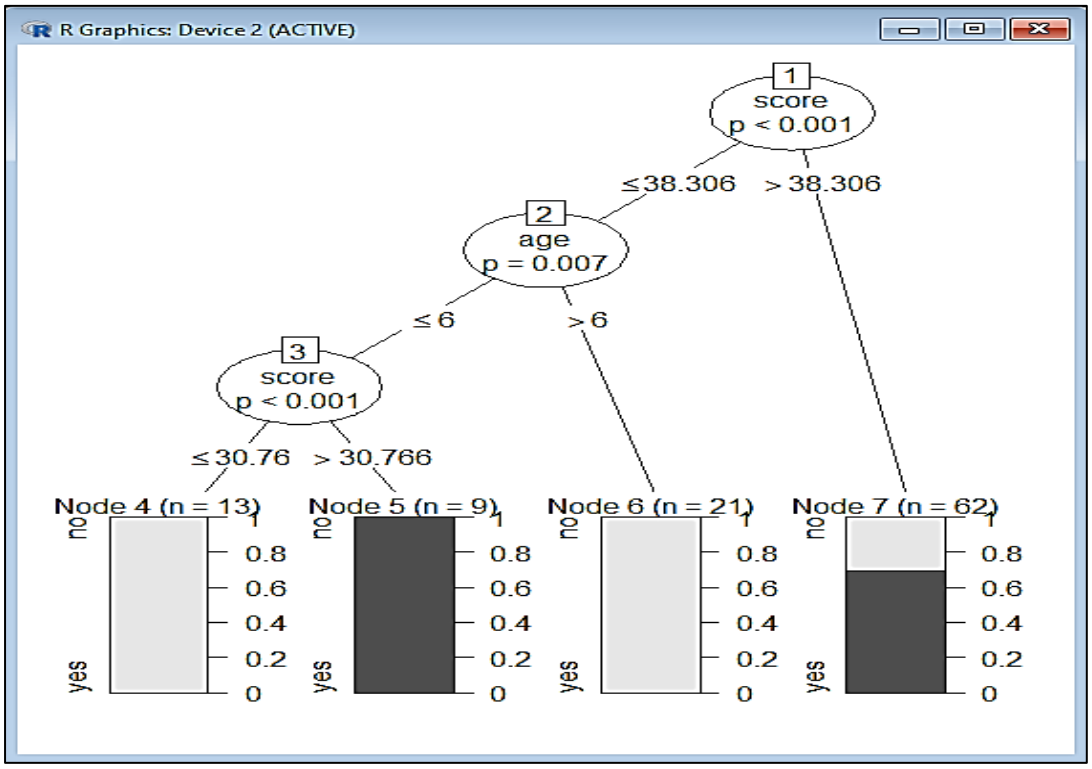
Loading required package: zoo

Attaching package: ‘zoo’

The following objects are masked from ‘package:base’:

as.Date, as.Date.numeric

Loading required package: sandwich



Experiment No. 8

Aim: k-means clustering using R.

Name: Ankit Singh Chauhan

Roll no: 64

Class: T.Y.BSc.IT

Subject: Business Intelligence

Marks\Grade:

Sign:

Name: Ankit Singh Chauhan
Roll No: 64

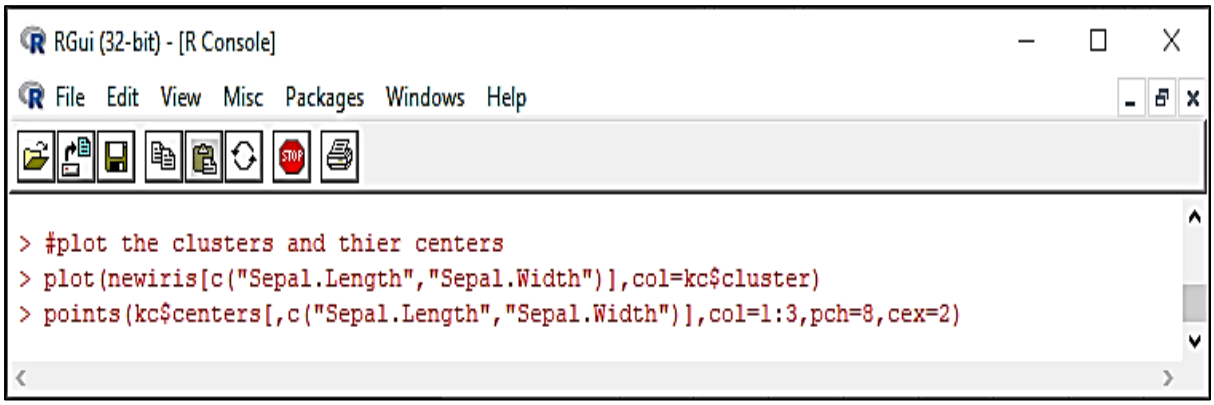
```
RGui (32-bit) - [R Console]
File  Edit  View  Misc  Packages  Windows  Help

> #Apply K mean to iris and store result
> newiris <- iris
> newiris$Species <- NULL
> (kc <-kmeans(newiris,3))
K-means clustering with 3 clusters of sizes 50, 62, 38

Cluster means:
  Sepal.Length Sepal.Width Petal.Length Petal.Width
1     5.006000    3.428000     1.462000    0.246000
2     5.901613    2.748387     4.393548    1.433871
3     6.850000    3.073684     5.742105    2.071053

Clustering vector:
  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 [38] 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 [75] 2 2 2 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 2 3 3 3 3 2 3 3 3 3
[112] 3 3 2 2 3 3 3 3 2 3 2 3 2 3 3 2 2 3 3 3 3 3 2 3 3 3 3 2 3 3 3 2 3 3 3 2 3
[149] 3 2

Within cluster sum of squares by cluster:
[1] 15.15100 39.82097 23.87947
 (between_SS / total_SS =  88.4 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
[6] "betweenss"    "size"         "iter"         "ifault"
>
```

Compare the Species label with the clustering result

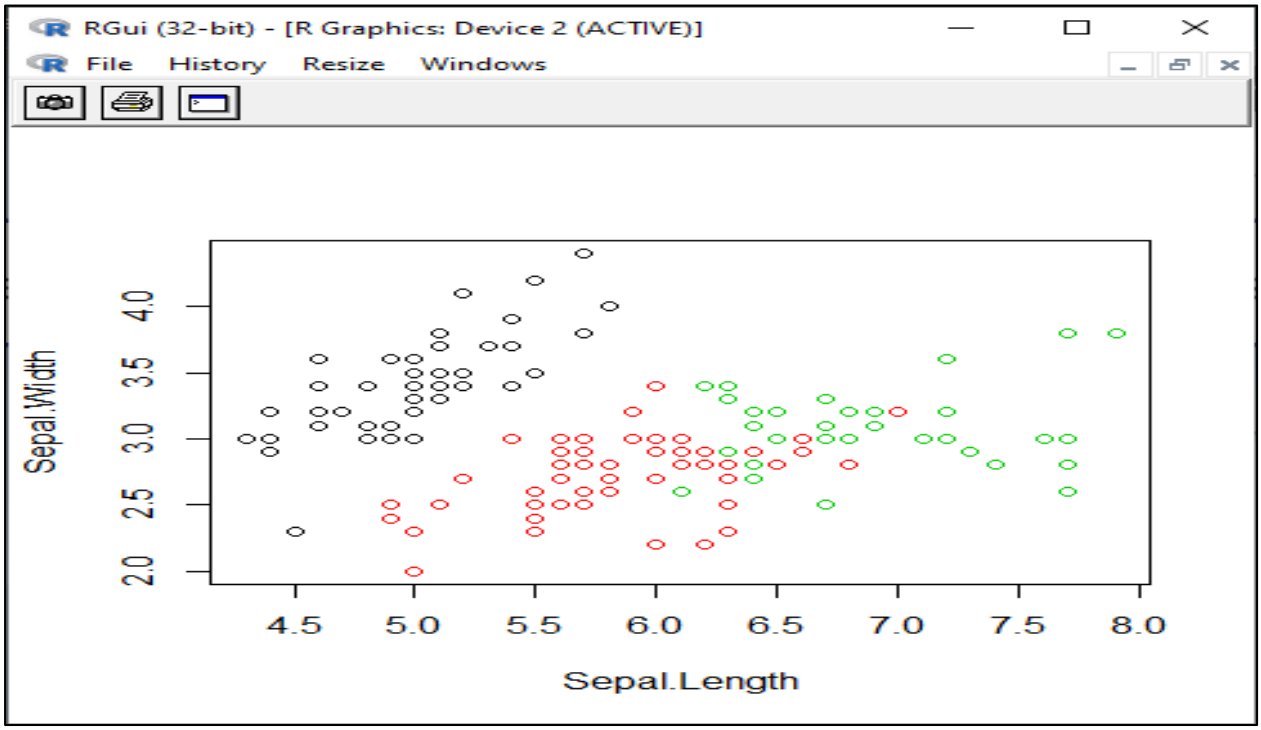
```
RGui (32-bit) - [R Console]
File  Edit  View  Misc  Packages  Windows  Help

> #Compare the Species label with the clustering result
> table(iris$Species,kc$cluster)

              1  2  3
  setosa     50  0  0
  versicolor  0 48  2
  virginica   0 14 36
>
```

Name: Ankit Singh Chauhan
Roll No: 64

Plot the clusters and their centers



Output:



Experiment No. 9

Aim: Prediction Using Linear Regression.

Name: Ankit Singh Chauhan

Roll no: 64

Class: T.Y.BSc.IT

Subject: Business Intelligence

Marks\Grade:

Sign:

Name: Ankit Singh Chauhan
Roll no: 64

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.

Name: Ankit Singh Chauhan
Roll no: 64

Syntax

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

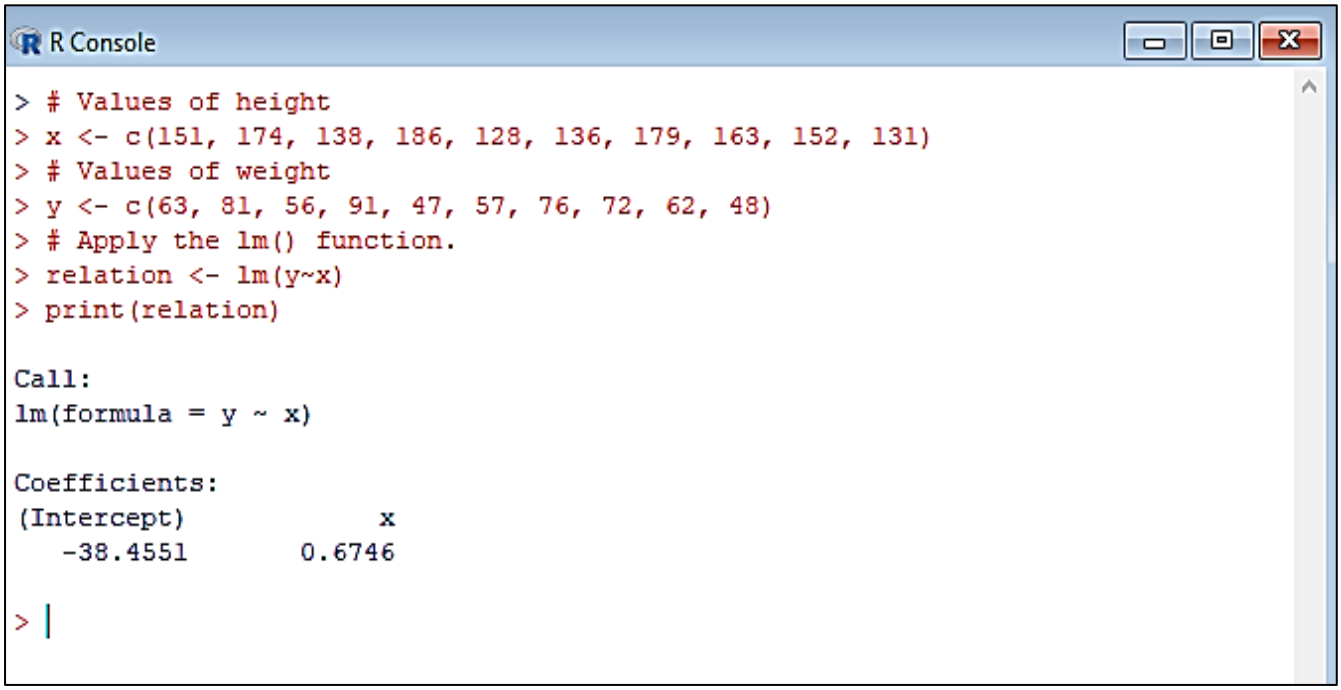
lm(formula,data)

Following is the description of the parameters used −

• **formula** is a symbol presenting the relation between x and y.

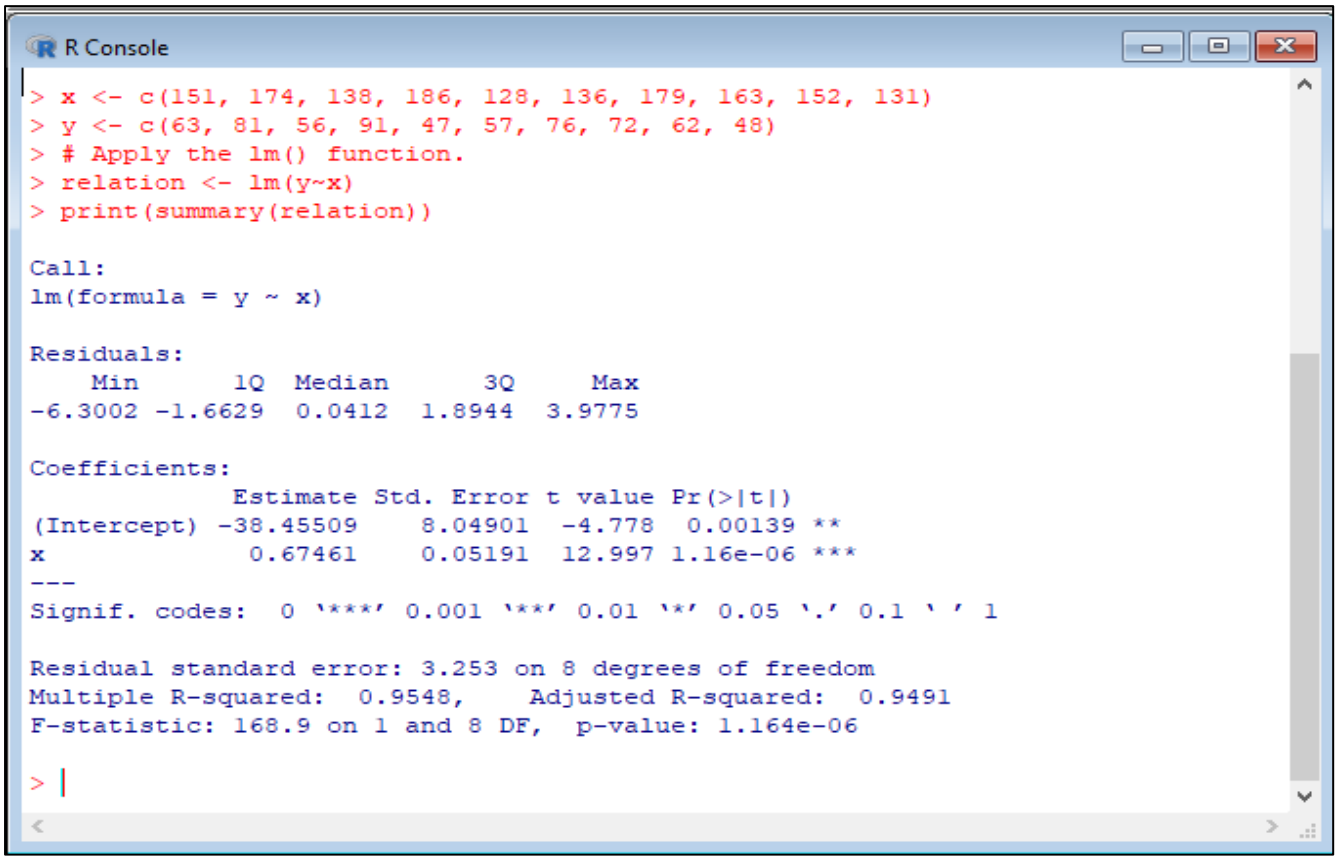
• **data** is the vector on which the formula will be applied.

Create Relationship Model & get the Coefficients :



Name: Ankit Singh Chauhan
Roll no: 64

Get the Summary of the Relationship



predict() Function

Syntax

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

predict(object, newdata)

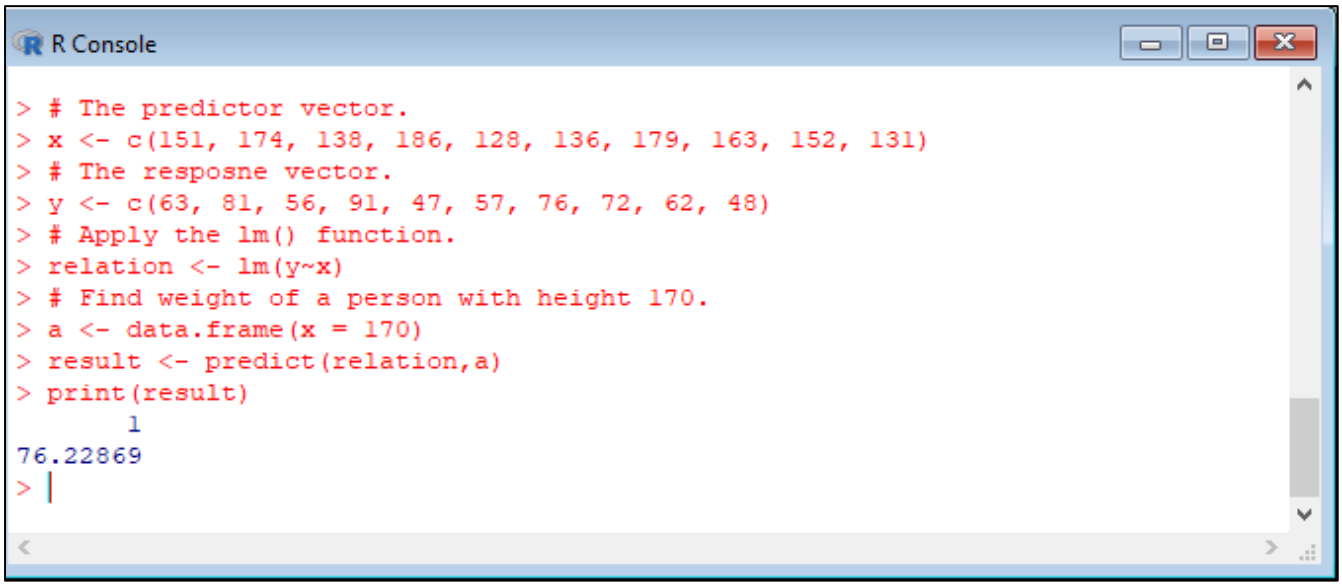
Following is the description of the parameters used −

• **object** is the formula which is already created using the lm() function.

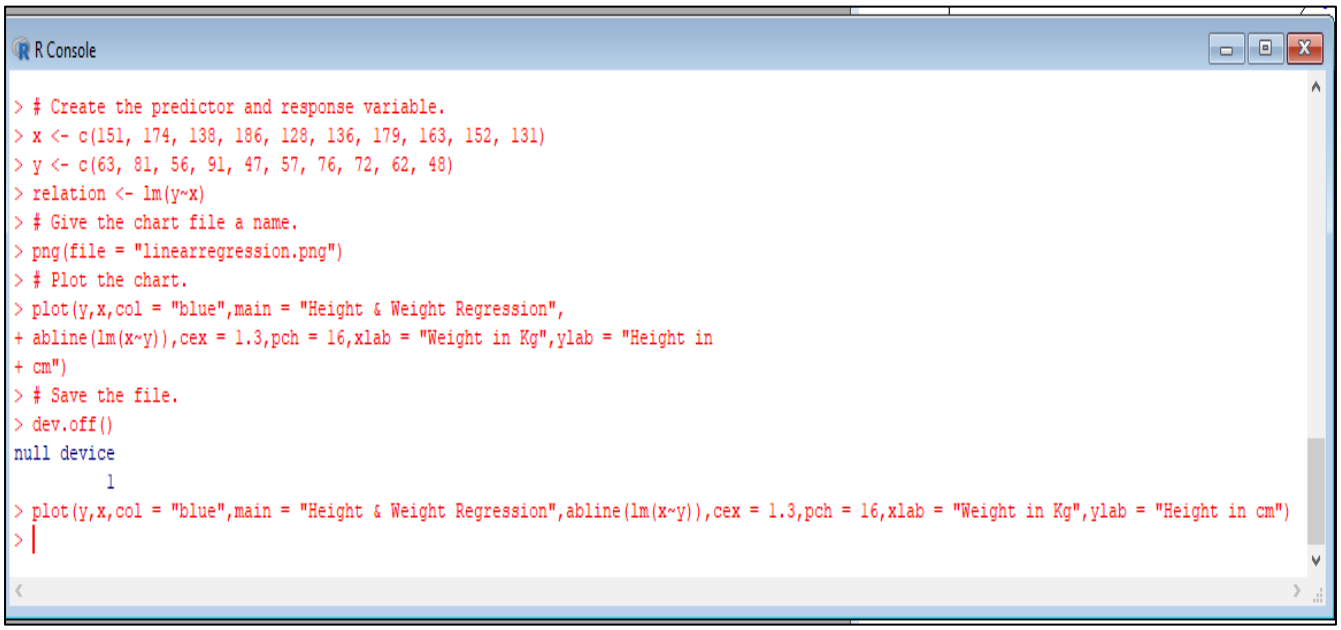
• **newdata** is the vector containing the new value for predictor variable.

Name: Ankit Singh Chauhan
Roll no: 64

Predict the weight of new persons



Visualize the Regression Graphically



Name: Ankit Singh Chauhan
Roll no: 64