|  |  |
| --- | --- |
| Power BI Lab Day 4 Document | |
|  |  | | |
| **Writer:** | I&D Microsoft | | **Initials entity:** |
| **Date creation:** 24/04/2018 | |  | **Last upgrade:** 25/04/2018 |

Power BI LAB DOCUMENT

DAY 4- Lab 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Version | Author | Comment | Reviewed By | Date |
| V 1.0 | I&D Microsoft | Initial draft | Moupiya Das |  |
|  |  |  |  |  |
|  |  |  |  |  |

**Pre-requisites**

Installed and working Power BI Desktop setup.

**Environment Setup**

1. **Installing Power BI Desktop**

Power BI Desktop lets you create a collection of queries, data connections, and reports that can easily be shared with others. Power BI Desktop integrates proven Microsoft technologies – the powerful Query engine, data modeling, and visualizations – and works seamlessly with the online Power BI service. You will need to download and install Power BI desktop to perform the labs in this course.

* 1. **Which version to install?**

• If you have a 32 bit machine, you need to install the 32bit Power BI Desktop.

• If you have a 32 bit Office installed (regardless of your machine), you need to install the 32 bit Power BI Desktop.

• Otherwise, you can install the 64 bit PBI Desktop.

**1.2 Minimum requirements**

• Windows 7 / Windows Server 2008 R2, or later

• .NET 4.5

• Internet Explorer 9 or later

**1.3 Download and install Power BI Desktop**

You can download and install the latest version of Power BI Desktop in two ways,

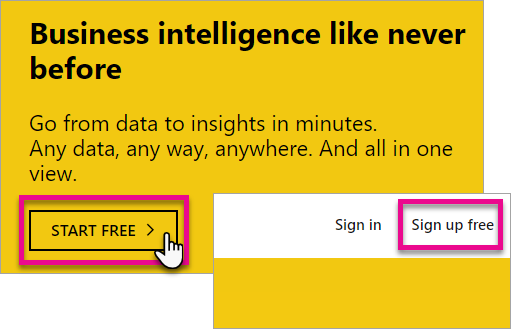
1. Directly from Microsoft link, http://go.microsoft.com/fwlink/?LinkID=521662

**Note: To select which version to download, go to** [https://www.microsoft.com/en-us/download/details.aspx?id=45331](https://www.microsoft.com/en-us/download/details.aspx?id=45331%20) and select appropriate version.

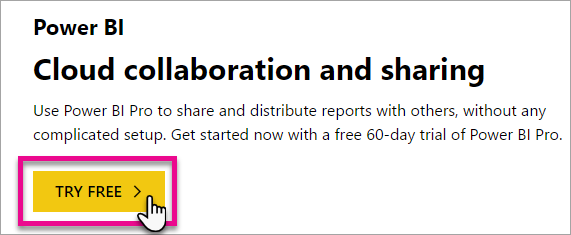
2. Or, from Power BI Service site, <https://app.powerbi.com/>, i.e., in Power BI, click the Downloads > **Power BI Desktop**.



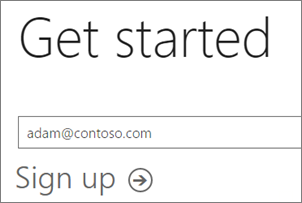
1. **Signing Up for Power BI Service**
2. Browse to [powerbi.com](https://powerbi.microsoft.com/).
3. Select **Start Free** or **Sign up free**.



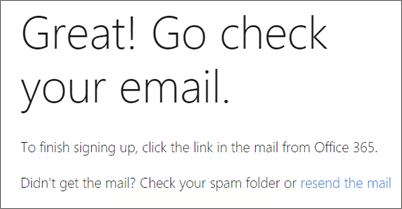
1. On the get started page, select **Try Free >** under Power BI.



1. Enter the email address you are signing up with, and then select **Sign up**. Be sure your email address is allowed for sign up. For more information about what email address you can use, see [What email address can be used with Power BI.](https://docs.microsoft.com/en-us/power-bi/service-self-service-signup-for-power-bi#what-email-address-can-be-used-with-power-bi)



1. You will get a message indicating to check your email.



1. Select the link within the email to verify your email address. This will bring you back into the sign up flow. You may need to supply some additional information about yourself.
2. You will then be taken to [https://app.powerbi.com](https://app.powerbi.com/) and you can begin using Power BI as a free user.

**Lab Overview**

This lab comprises of four sections:

1. In the first section, you will create role based security based on territory region.

2. In the second section, you will basic DAX measures.

3. In the third section, you will create few measures and understand how they are different from calculated columns by using Time Intelligence function.

4. In the fourth section you will understand how calculations work in DAX by means of context and context transition.

# Table of Contents

[Table of Contents 6](#_Toc512863192)

[1. RLS – Role Based Security Lab 7](#_Toc512863193)

[2. Creating Measures Lab 16](#_Toc512863194)

[3. Time Intelligence & Calculate column vs Measures Lab 18](#_Toc512863195)

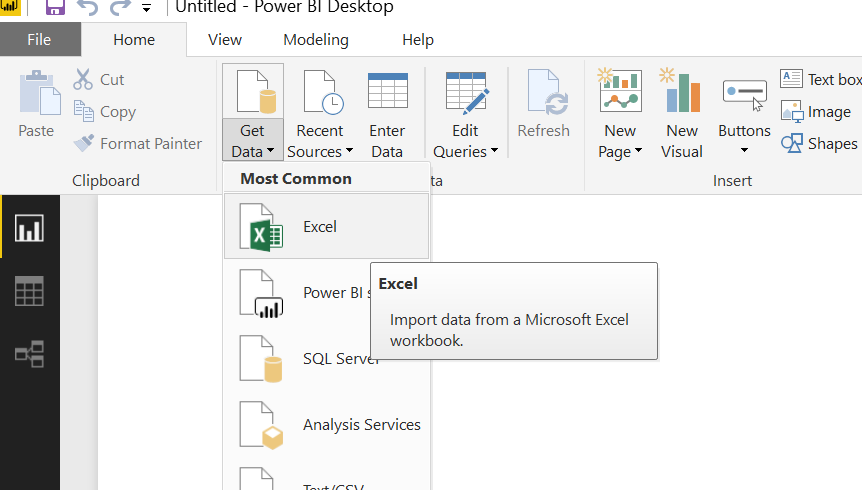
[4. Understanding DAX computation logic, Context Transition Lab 20](#_Toc512863196)

# RLS – Role Based Security Lab

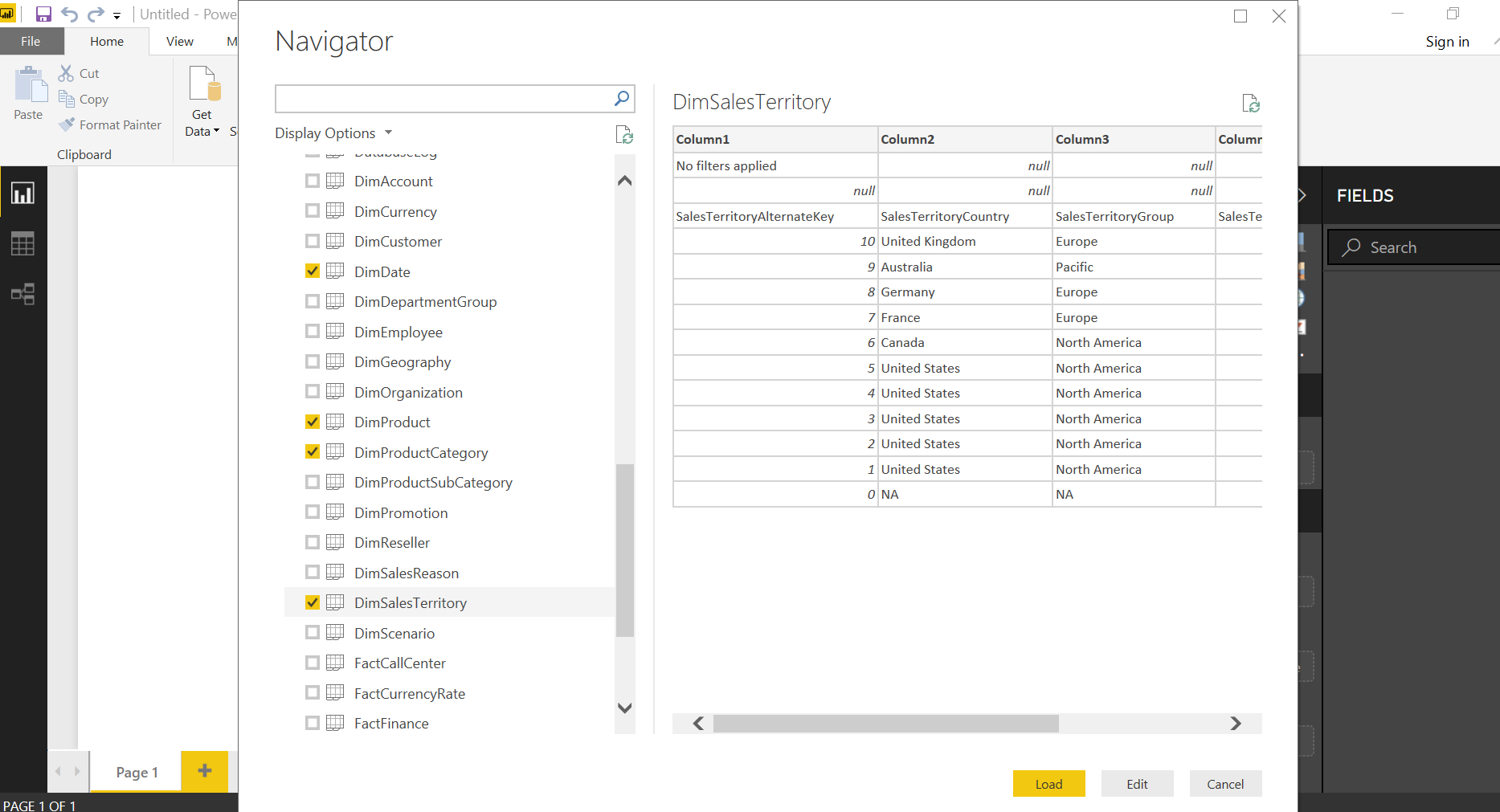
**Case Scenario**

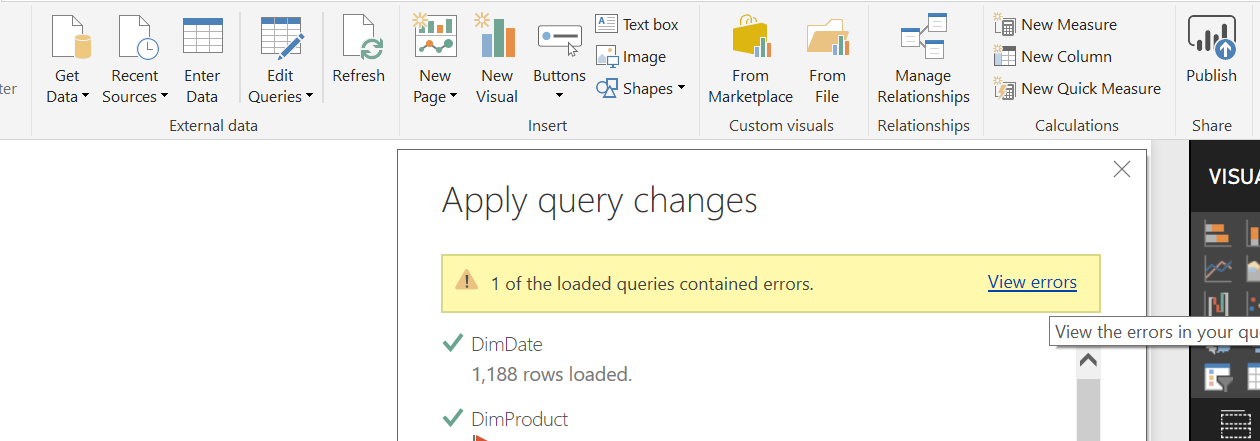
This case requires sales managers to view data from their respective allocated region. To complete this requirement, implement Role based security on the data by creating roles based on region and provide membership to users for their respective region. That way, members belonging to their region will see only their data and not the data related to another region. Now let us implement this using the data from Adventure works sample excel file (You should be able to find this in the file: ***AdventureWorksDatabase.xlsx*** ).

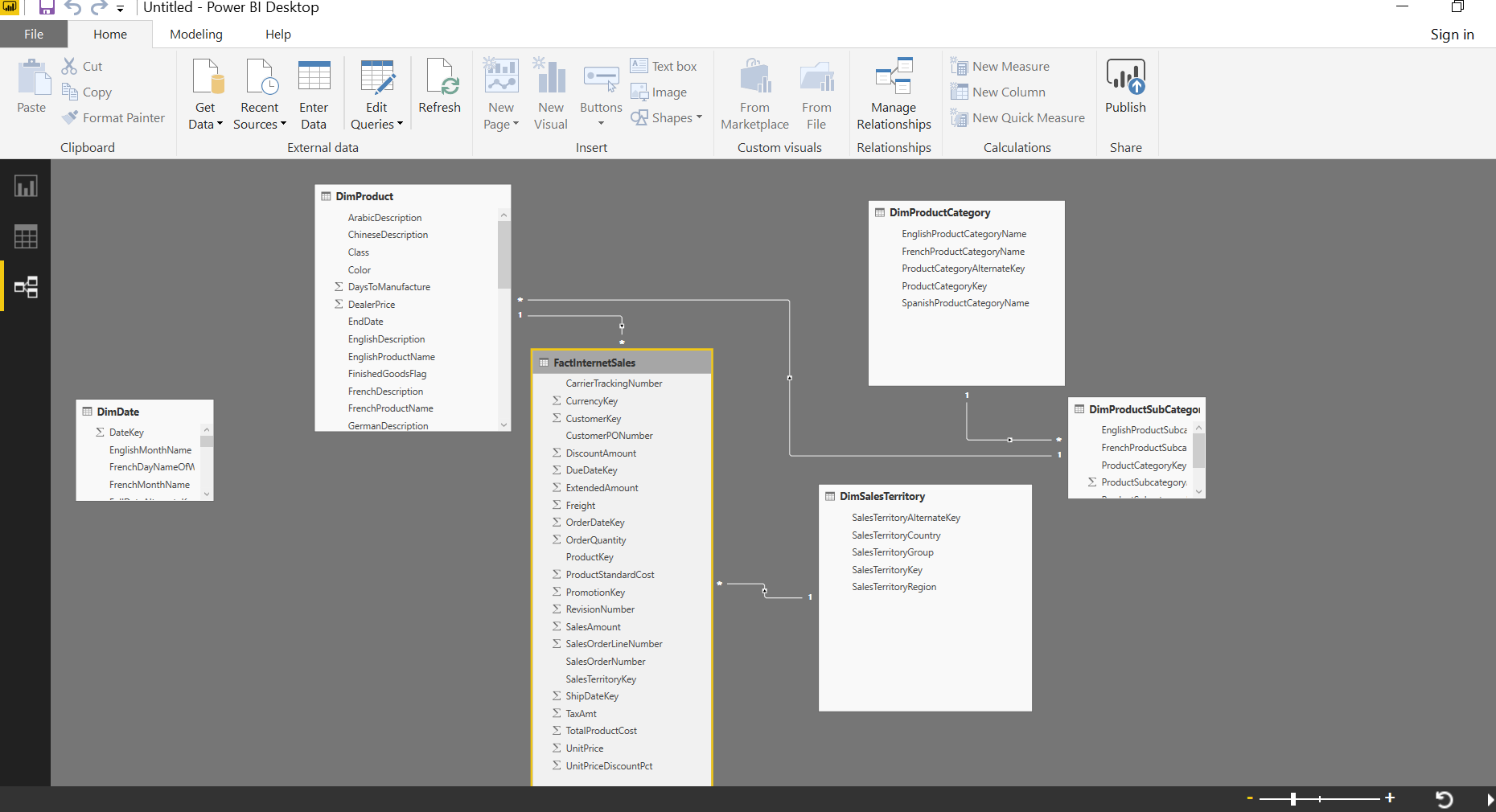
1. Open the **Power BI Desktop**. You can try to sign if it asks for provided you have access to **Power BI Service** as demonstrated in earlier labs. Then, click on Get Data 🡪 Excel and browse the excel file for *AdventureWorksDatabase.xlsx*.



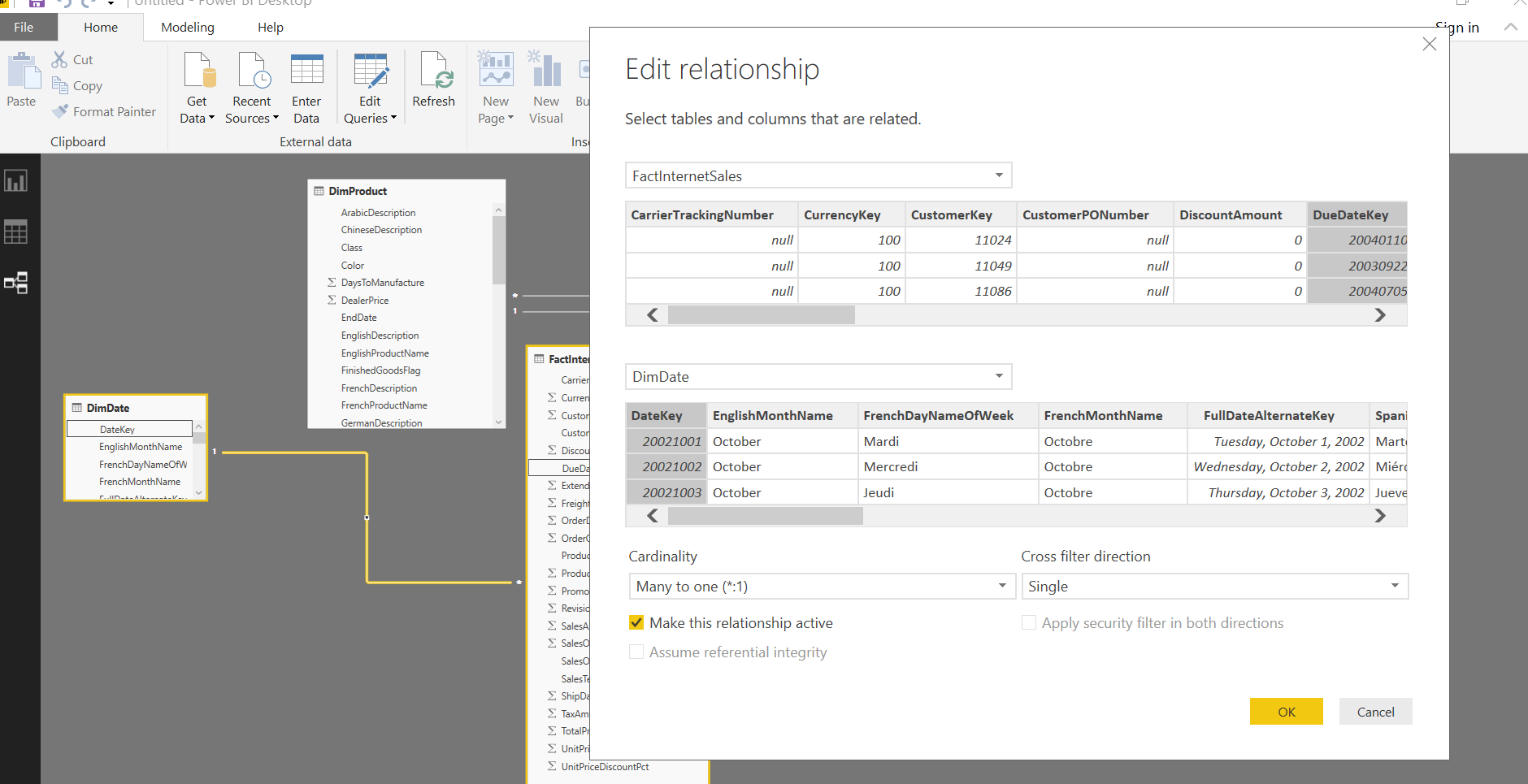
1. Select DimDate, DimProduct, DimProductCategory, DimSalesTerritory and FactInternetSales. Click **Load** to load the data. You must see the below screen in this process.



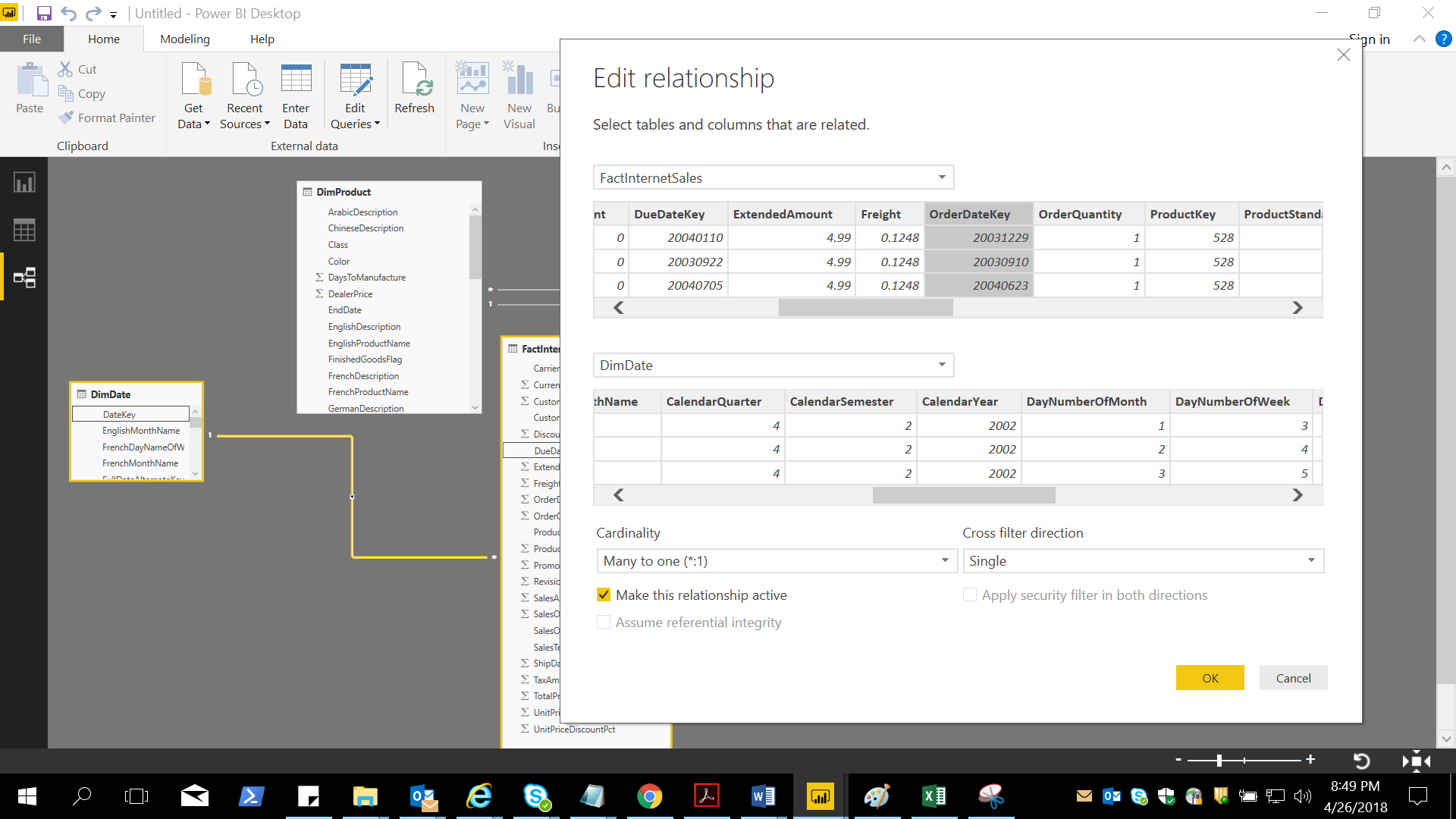
1. Load the data after this operation. You won’t face any error now; however, if you face errors try to resolve by considering the query. 
2. Check the relationship tab and the data model should now look like:



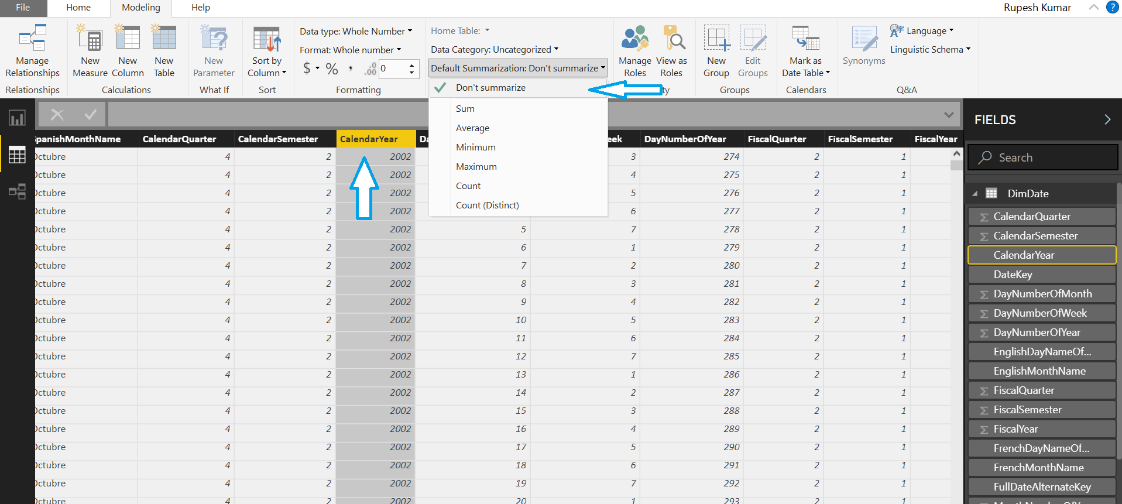
1. Click on Manage relationship and create a new relationship between DimDate and FactInternetSales as FactInternetSales.OrderDateKey = DimDate.DateKey. Select Ok to activate this relationship.



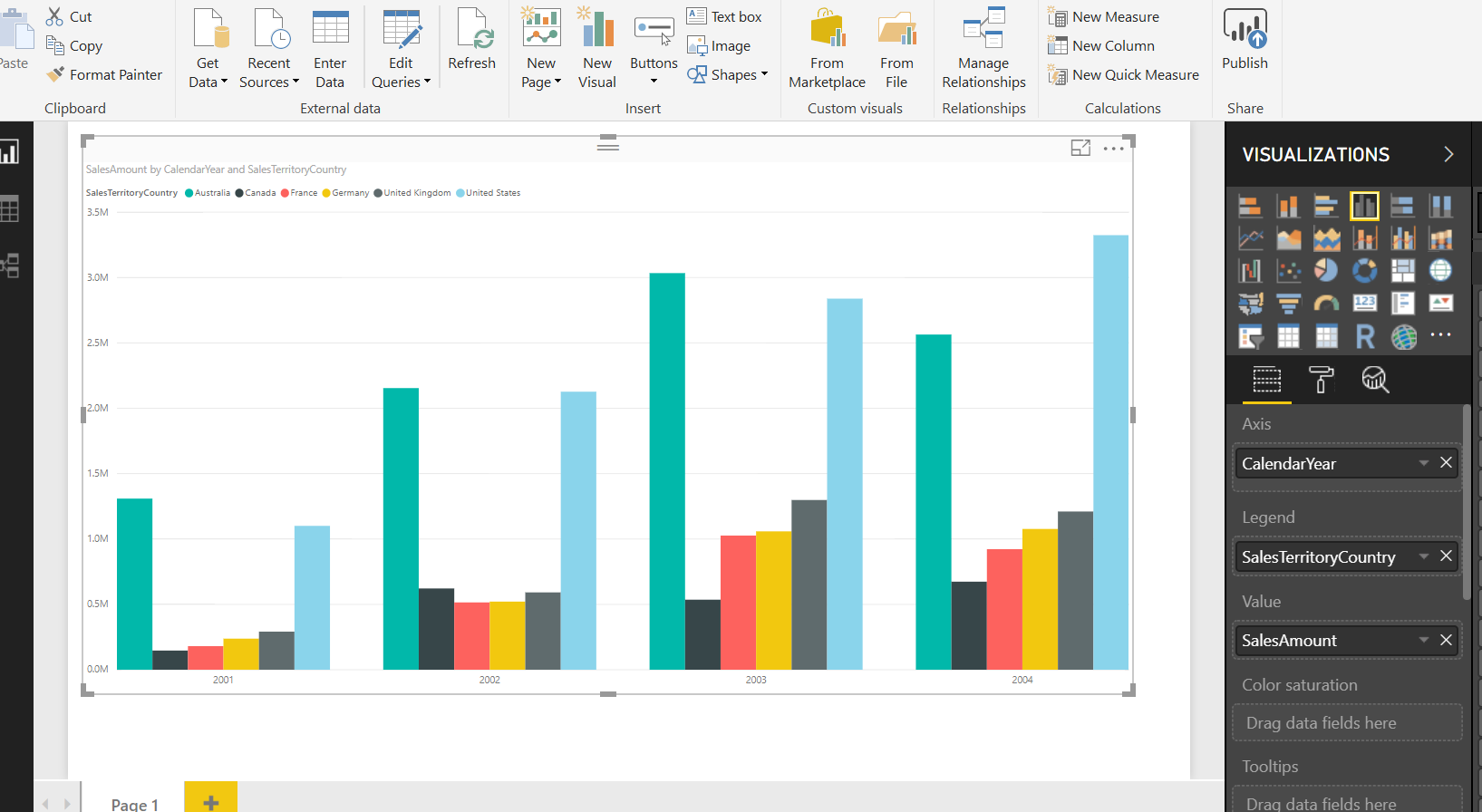
The relationship under the relationships tab should now look like:



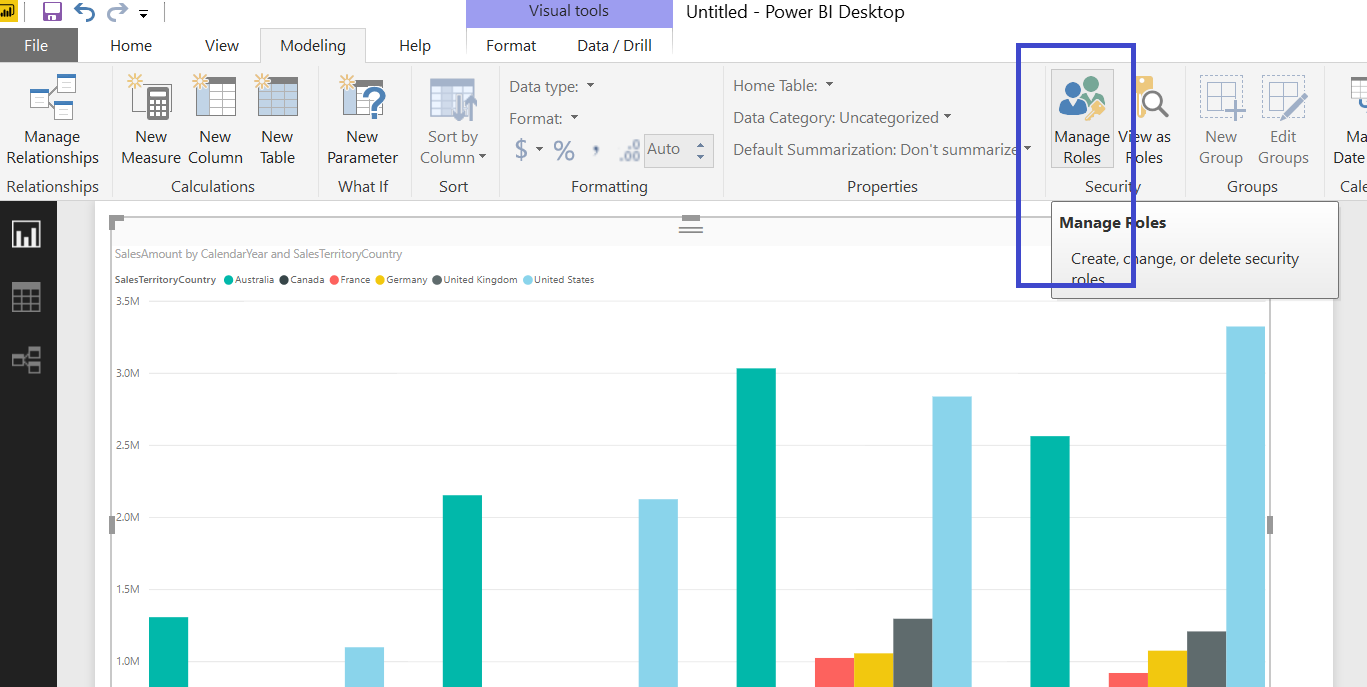
1. Navigate data from different dimensions and fact. Click the DimDate and click on the CalendarYear column. Observe the **Data Type**, **Format**, **Data Categorization** and **Default Summarization** under the **Modeling** tab. Change the **Default Summarization** to Don’t Summarize for CalendarYear column.



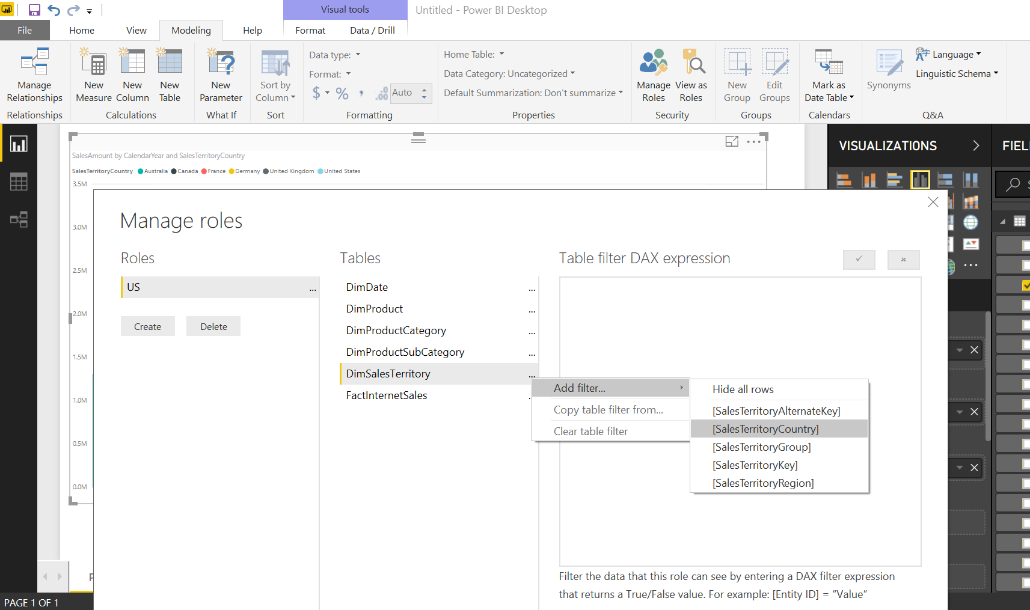
1. Now, create a **Clustered Column chart** visual with the fields: **DimDate.CalendarYear**, **DimSalesTerritory.SalesTerritoryCountry** and **FactInternetSales.SalesAmount**.



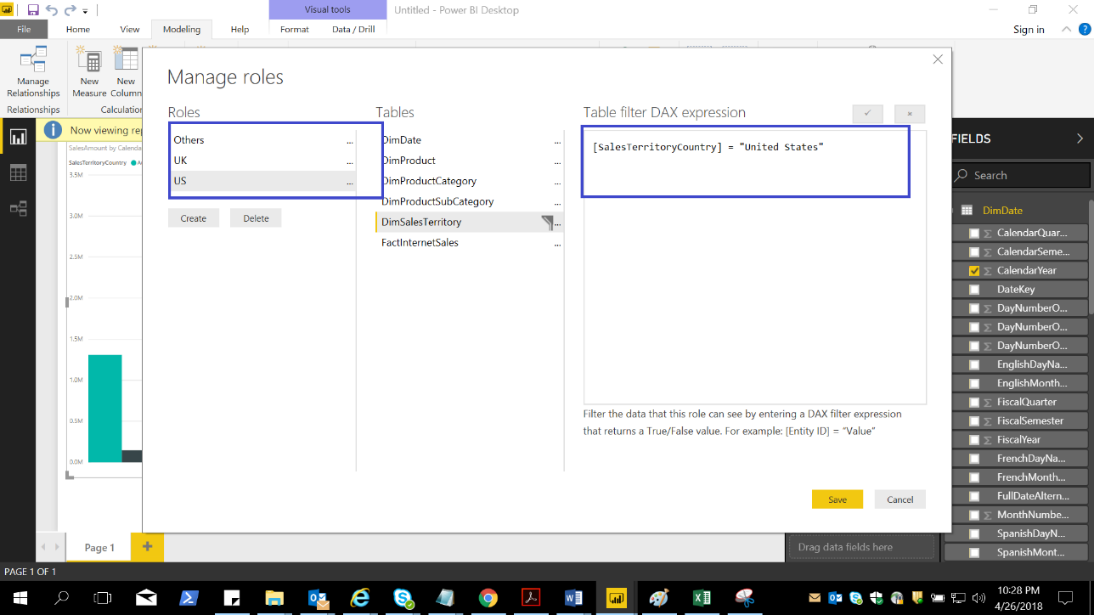
1. After thus, click **Modeling** 🡪 **Manage Roles**



1. A window should now appear like the below. Start creating new roles here by clicking **Create**. Double click on the role names to rename the roles. For the first role, rename it to **US**, then click the 3 ellipses on the right of DimSalesTerritory, hover the mouse to Add filter and click on [SalesCategoryCountry]. This will populate the DAX expression under the Table filter DAX expression window.

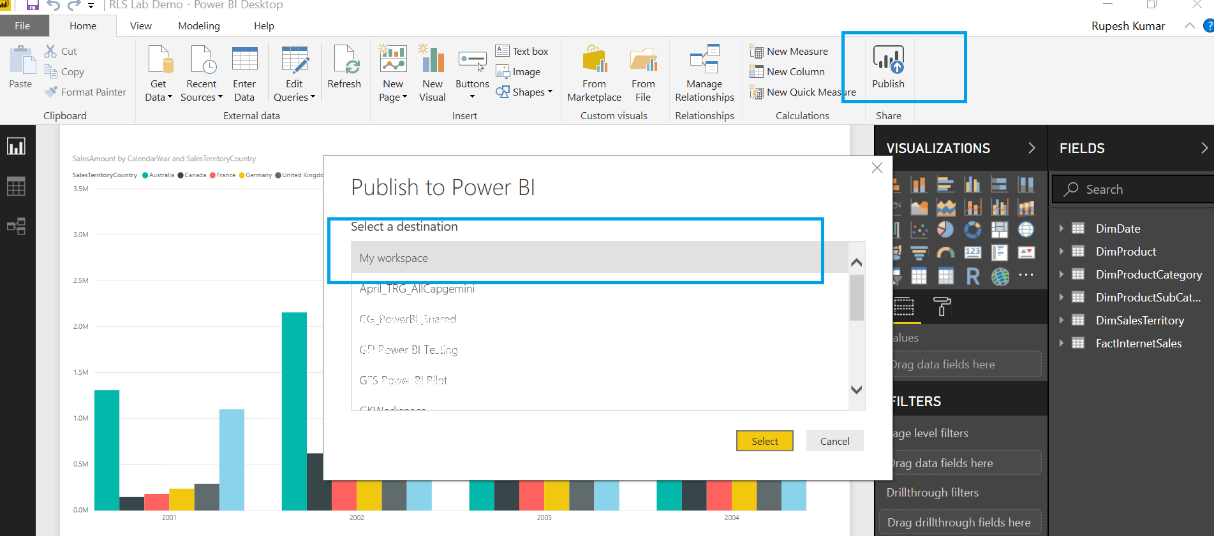


Change the DAX expression so that the value is “**United States**”.

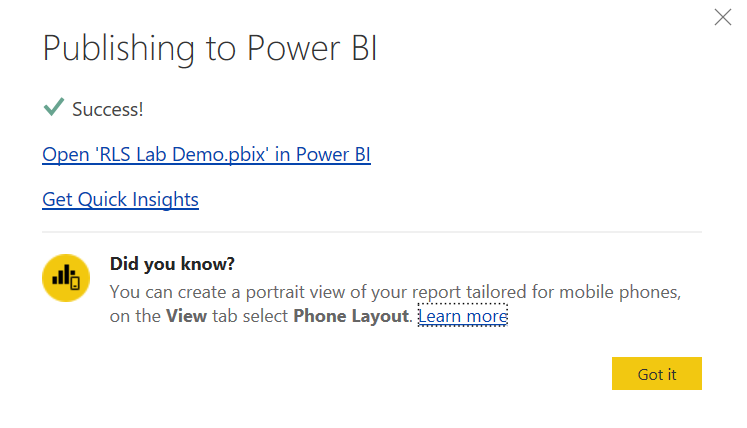


Follow the same rules to create 2 more roles for UK and Others. The **UK** DAX expression should be **[SalesTerritoryCountry] = "United Kingdom"** and **Others** should be **[SalesTerritoryCountry] <> "United States" && [SalesTerritoryCountry] <> "United Kingdom"**

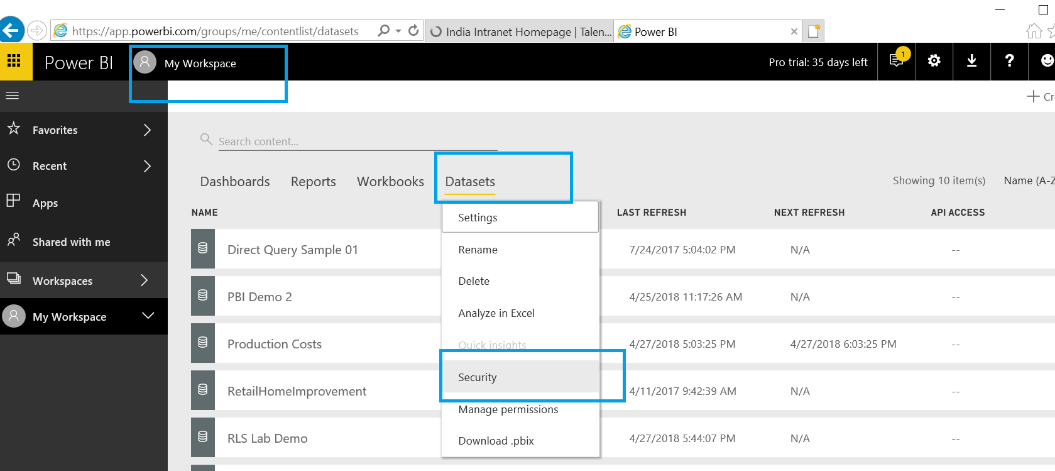
1. Save this PBI file by the name **RLS Lab Demo**.
2. For these next set of tasks make sure you have access to Power BI Service and logged in. Click on the Publish to publish the newly created. Select **My workspace** so that the PBI file gets published to your workspace.



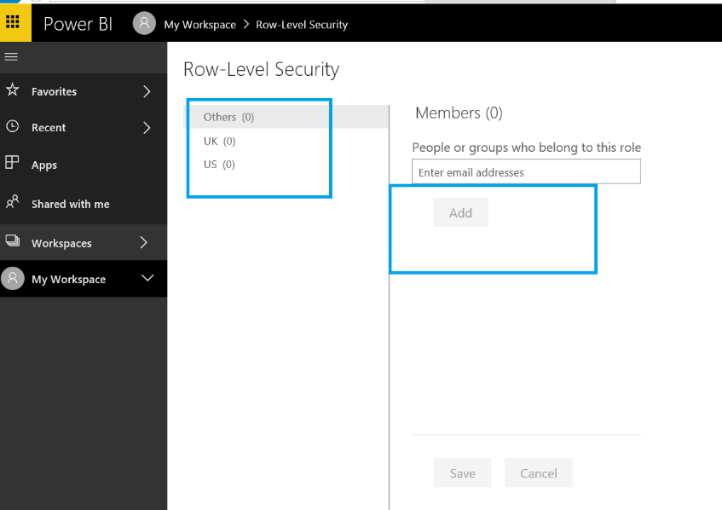
On successful publishing you should get the below message:



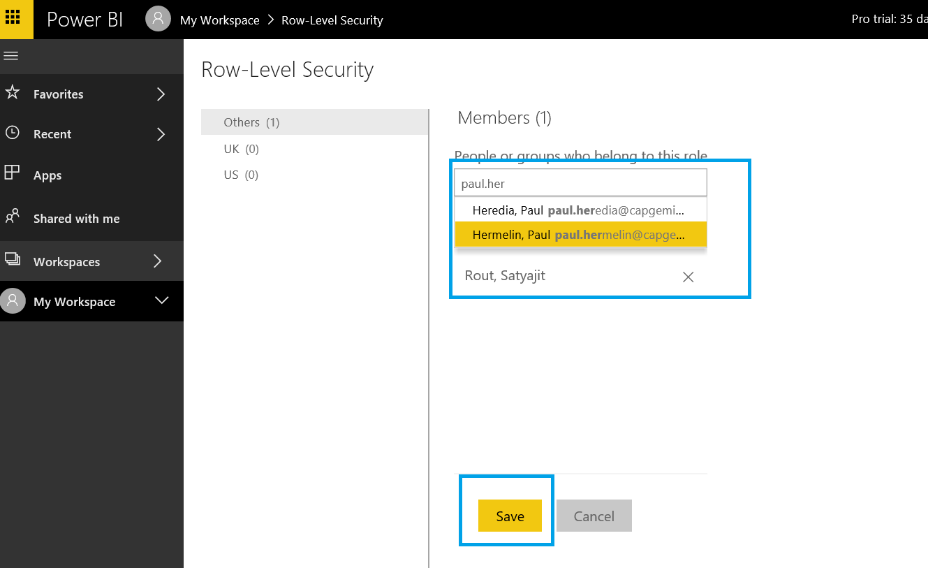
1. Log on to Power BI portal and navigate to My workspace. Click on the **Datasets** and click on the 3 ellipses under **Actions** and to the right of RLS Lab Demo to get **More Options** 🡪 **Security**.



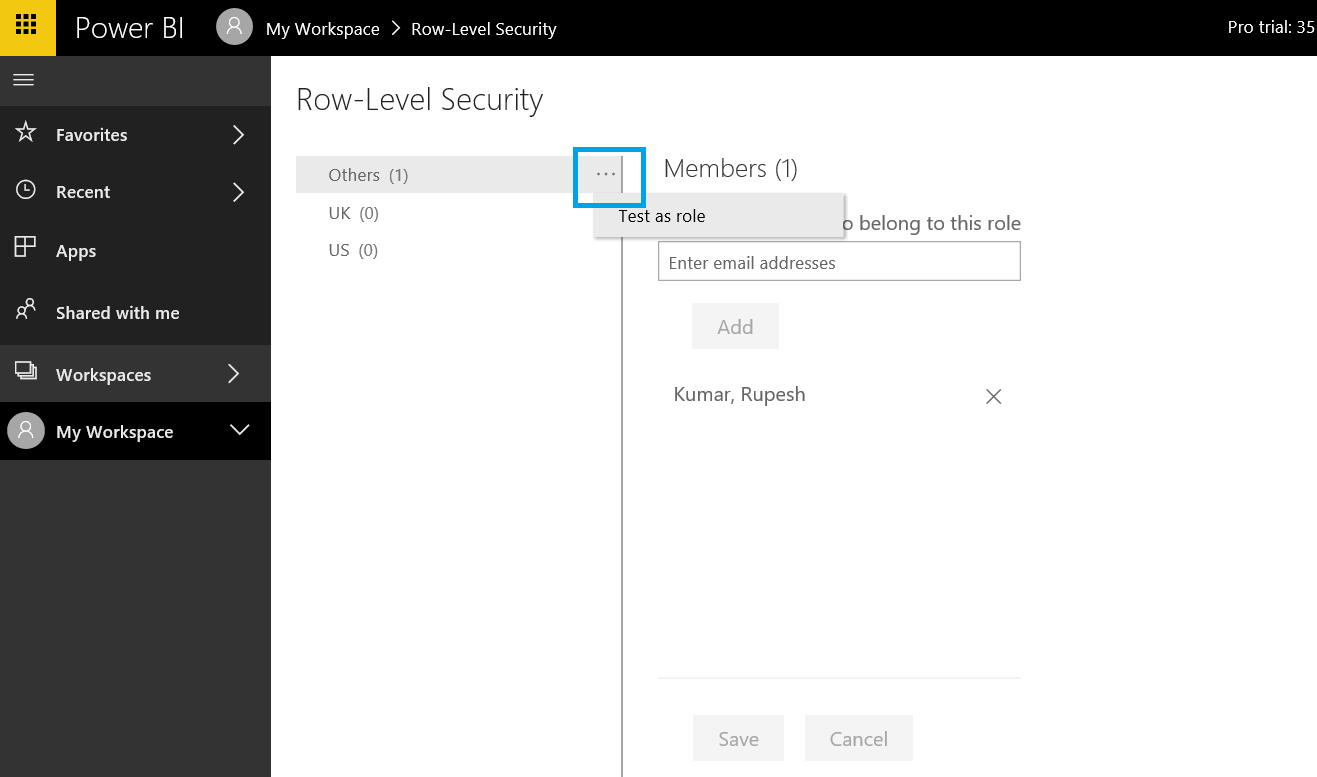
Row-Level Security should now appear like the one below:



Click on a particular role and under **Members**, type an e-mail address to whom the role should be associated. Add a few names to different roles and click **Save** to save the changes.



1. Let is test the roles just created. Click on the 3 ellipses on the right of any role and click the **Test as role**. Observe by clicking different roles and the data that appear in the chart.



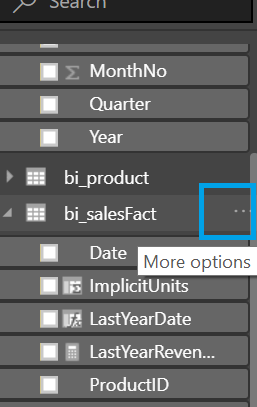
**Summary:**

This completes the lab session where you have observed how different users under different roles views data belonging to their own roles defined by the DAX filter expression. You can perform interactive testing by asking assigned Users actually log in to Power BI and verify that row level filters are working.

# Creating Measures Lab

This lab is meant to expose users to creating basic measures and observe how they work.

1. Open the SalesSample.pbix file that is preloaded with the data on PBI Desktop.
2. Create a **Table** visualization with the following attributes: -
3. **bi\_date** : Year and MonthNo
4. **bi\_salesFact**: Revenue
5. Click on the ellipse (3 dots) beside **bi\_salesFact** and create a New Measure



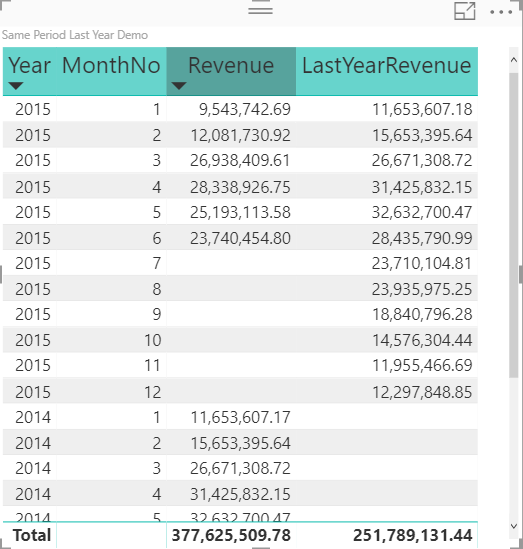
Paste the text of the new measure as:

**TotalRevenue = SUM(bi\_salesFact[Revenue])**

Next, Create another measure and use the below DAX expression to define it

**LastYearRevenue = CALCULATE(SUM(bi\_salesFact[Revenue]), SAMEPERIODLASTYEAR(bi\_date[Date]))**

1. Use this measure (TotalRevenue) in the **Table** visual just created and should look like:



Observe how the **Last Year Revenue** is calculated based on year and month number. Compare the values of 2014 and 2015 for **Revenue** and **LastYearRevenue**.

1. Now remove **MonthNo** and add **Quarter** instead. Observe how the values change both for **Revenue** and **LastYearRevenue**.

Create one more measure as follows:

**Variance Per = ([TotalRevenue]-[LastYearRevenue])/[TotalRevenue]**

Add this to the existing visual.

1. Create a **Card** visual and add **Variance Per** measure just created. Compare this with the values in Table visual just created to understand how variance percentage is dynamically calculated based on the evaluation context. Save and close the Power BI file.

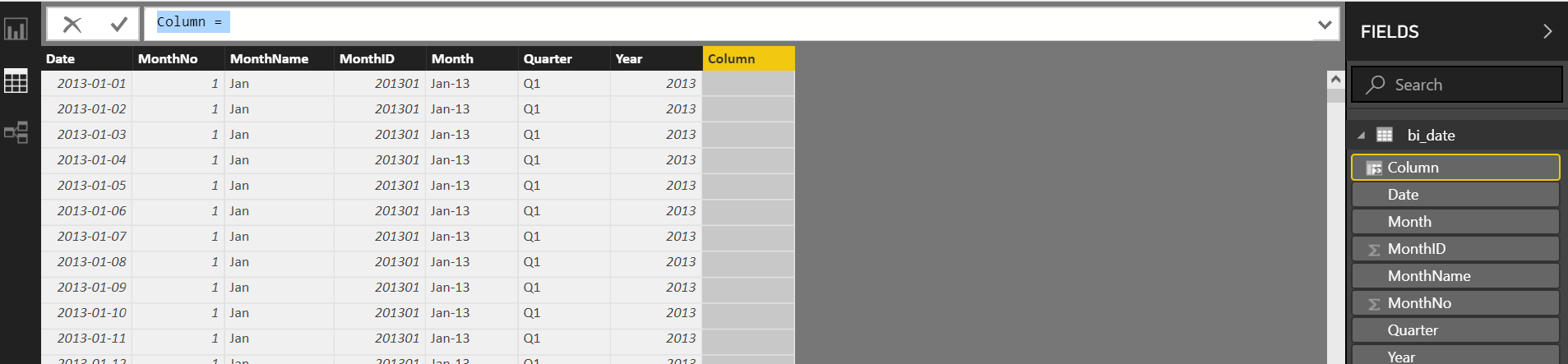
**Summary:**

This completes the Power BI lab session for basic understanding of measures and how it works.

# Time Intelligence & Calculate column vs Measures Lab

This lab is targeted in understanding Time intelligence functions and how it works in Power BI. Let us start this session by creating few time related functions.

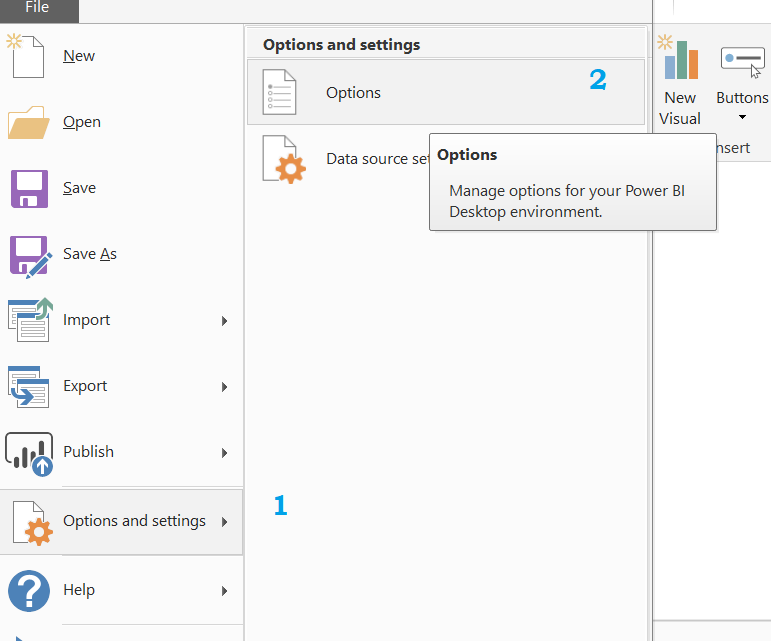
1. Click on the **Modeling** tab 🡪 Select the Data 🡪 **New Column** to add a new column.



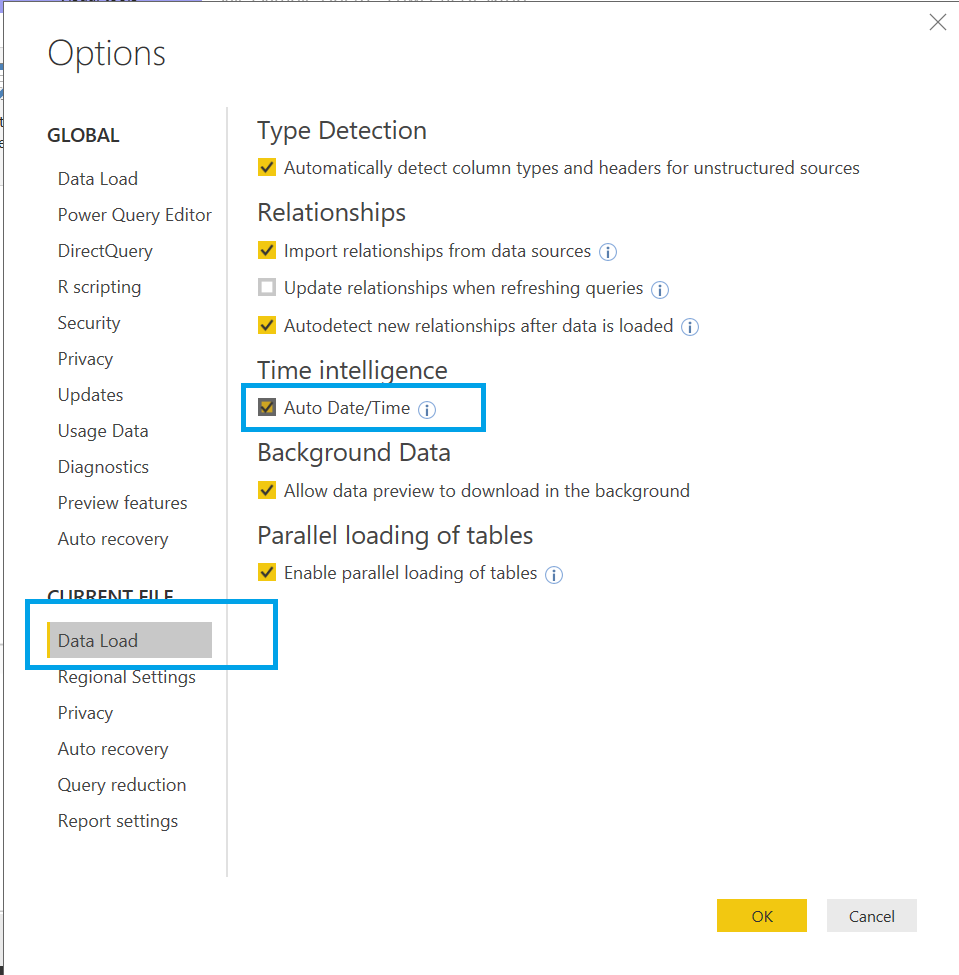
1. Edit the DAX query replace with **TimeNow = NOW()**

Observe the time that gets into the new column.

1. Create a new **Table** visual on a new Page and use **TimeNow** column just created. Observe how Year, Quarter, Month and Day gets loaded into different columns automatically!
2. Let us now play around with the time functions now. Go to menu **File** 🡪 **Options and settings** 🡪 **Options**



Then select the **Data Load** and under **Time intelligence** uncheck the **Auto Date/Time**.



Observe the Table visual just created changes and shows only the current date time as one single row. Also observe how the automatically created hierarchy of time got changed to have only oen field now under the fields selection for the Table visual.

1. Change this visual to a **Card** visual.
2. Next create a new measure on bi\_date as **MeasureNow = NOW()**
3. Create a new **Card** visual with MeasureNow as the field. Save the report now.
4. **Publish** the report and observe the values of the two visuals. Select on the **Refresh** button and observe what changes in the two visuals!!

Do both the visual values change? If not why? What is the reason behind this change?

Complete this lab by answering the questions above and then checking the **Auto Date/Time** checkbox, save and close the PBIX file.

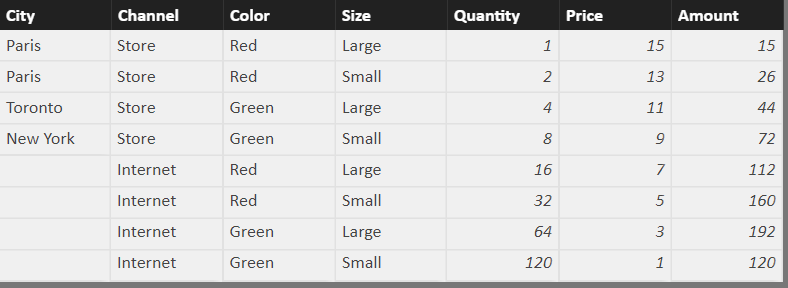
**Summary:**

With this you have basic understanding of DAX function related to time and how time intelligence makes a difference.

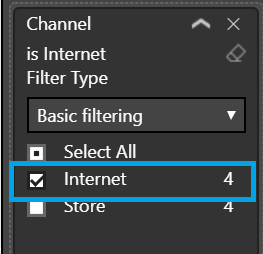
# Understanding DAX computation logic, Context Transition Lab

This lab session is aimed at creating and understanding DAX computation in more detail and understanding why things work the way it does. This understanding is if utmost importance with small set of data first and gradually increase the complexity in live projects.

1. Load the Orders.csv file into Power BI desktop. It should appear like the below:



1. Create a page level filter by checking the value of Internet for Channel. This will ensure that only Internet Channel are reported by normal visualization on Power BI report pages.



1. Now, create a new measure CalcAmount based on the below DAX:

**CalcAmount = SUMX(**

**FILTER(Orders, Orders[Price] > 1), Orders[Amount])**

1. Create a new **Table** visual and use the fields – **Color**, **Size**, and **CalcAmount**. Observe that the record line with color as green and size as small does not show any value. Why is that so?
2. Now, create one more measure as

**CalculateDemo = CALCULATE([CalcAmount], FILTER(ALL(Orders),Orders[Color]="Red"))**

Include this newly created measure as one of the fields in the above visual. Determine how is the summation happening by looking at the records in the Orders dataset.

1. Now, answer this question: **What is the highest amount for each color in this dataset?**

Think of how this can be achieved. Let us give it a try now.

Create a new measure based on the below DAX expression

**WithoutContextTransition = MAXX('Orders',**

**SUMX(Orders, Orders[Amount]+0**

**))**

1. Now create a **Table** visual to select the **Color** field and the newly created measure **WithoutContextTransition**. Does this give the right answer? Think a while.
2. Next create another measure:

**WithContextTransition = MAXX('Orders',**

**CALCULATE(SUMX(Orders, Orders[Amount])+0**

**))**

1. Create a new **Table** visual and select **Color** and the above newly created measure. Does this give the right answer? **What is the role of CALCULATE in this DAX expression?**
2. Now let us play around with one more DAX function **ALLSELECTED().** Clear the report level filters.
3. Create a new measure:

**AllSelectedSample = CALCULATE(SUM(Orders[Amount]), ALLSELECTED(Orders[Color]))**

1. Create a **Table** visual and select as the fields: Color, Channel, and AllSelectedSample

Irrespective of the color, **is the calculation correct for the given Channel?**

1. Next, change the above DAX measure to:

**AllSelectedSample = CALCULATE(SUM(Orders[Amount]), ALLSELECTED(Orders[Channel]))**

Is the figure correct for a given color irrespective of the Channel?

1. Observe the values coming because of the changes and the context under which DAX is getting evaluated.

**Summary:**

With this understanding of context and context transition, you have the understanding of DAX calculations and behavior of certain important functions.