# POWER BI LAB - Data Sources Data Shaping Modeling and Workflow II

# THEORY SECTION

**Multidimensional Data Modeling-Understand the star schema and its importance for Power BI**

## Star schema

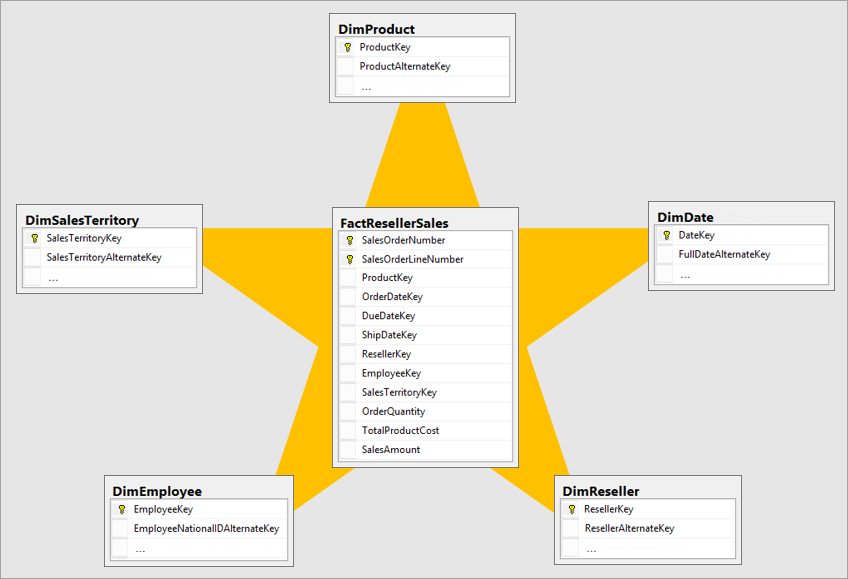
is a mature modeling approach widely adopted by data warehouses. It requires modelers to classify their tables as either *dimensions* or *facts*.

## Dimension tables

Describe business entities—the *things* you model. Entities can include products, people, places, and concepts including time itself. The most consistent table you'll find in a star schema is a date dimension table. A dimension table contains a key column (or columns) that acts as a unique identifier, and descriptive columns.

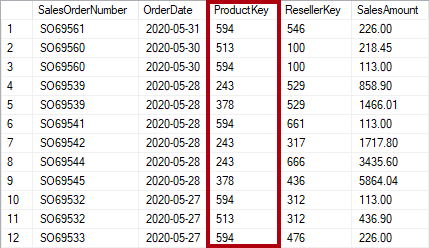
## **Fact tables**

Store observations or events, and can be sales orders, stock balances, exchange rates, temperatures, etc. A fact table contains key columns that relate to the dimension tables, and numeric measure columns. Fact tables contain a very large number of rows and continue to grow over time. 90% of the data in a data warehouse is in the fact table.



## Normalization vs. denormalization

***Normalization*** is the term used to describe data that's stored in a way that reduces repetitious data. Consider a table of products that has a unique key value column, like the product key, and additional columns describing product characteristics, including product name, category, color, and size. A sales table is considered normalized when it stores only keys, like the product key. In the following image, notice that only the **ProductKey** column records the product.

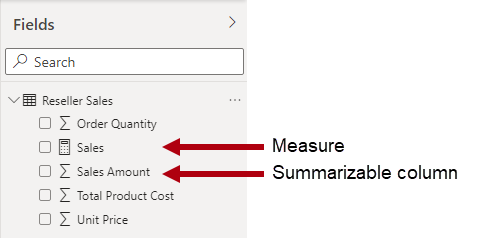
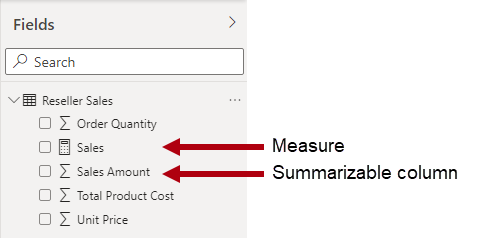


A **Denormalized** Data Store (like a database) contains a few tables with many fields in them.

## Measures

In star schema design, a **measure** is a fact table column. In a Power BI model, a **measure** has a different—but similar—definition. It's a formula written in [Data Analysis Expressions (DAX)](https://learn.microsoft.com/en-us/dax/data-analysis-expressions-dax-reference) that achieves summarization. Measure expressions often leverage DAX aggregation functions like SUM, MIN, MAX, AVERAGE, etc. to produce a scalar value result at query time (values are never stored in the model). Measure expression can range from simple column aggregations to more sophisticated formulas that override filter context and/or relationship propagation.

It's important to understand that Power BI models support a second method for achieving summarization. Any column—and typically numeric columns—can be summarized by a report visual or Q&A. These columns are referred to as *implicit measures*. They offer a convenience for you as a model developer, as in many instances you do not need to create measures. For example, the Adventure Works reseller sales **Sales Amount** column could be summarized in numerous ways (sum, count, average, median, min, max, etc.), without the need to create a measure for each possible aggregation type.



## Surrogate keys

A **surrogate key** is a unique identifier that you add to a data warehouse dimension tables to provide a unique identifier for each dimension table row. Power BI model relationships are based on a single unique column in one table, which propagates filters to a single column in a different table. When a dimension-type table in your model doesn't include a single unique column, you must add a unique identifier to become the "one" side of a relationship.

**Slowly changing dimensions**

A **slowly changing dimension** (SCD) is one that appropriately manages change of dimension members over time. It applies when business entity values change over time, and in an ad hoc manner. A good example of a *slowly* changing dimension is a customer dimension, specifically its contact detail columns like email address and phone number. In contrast, some dimensions are considered to be *rapidly* changing when a dimension attribute changes often, like a stock's market price. The common design approach in these instances is to store rapidly changing attribute values in a fact table measure.

Star schema design theory refers to two common SCD types: Type 1 and Type 2. A dimension-type table could be Type 1 or Type 2, or support both types simultaneously for different columns.

## Type 1 SCD

A Type 1 SCD always reflects the latest values, and when changes in source data are detected, the dimension table data is overwritten. This design approach is common for columns that store supplementary values, like the email address or phone number of a customer. When a customer email address or phone number changes, the dimension table updates the customer row with the new values. It's as if the customer always had this contact information.

## Type 2 SCD

A **Type 2** **SCD** supports versioning of dimension members. If the source system doesn't store versions, then it's usually the data warehouse load process that detects changes, and appropriately manages the change in a dimension table. In this case, the dimension table must use a surrogate key to provide a unique reference to a *version* of the dimension member. It also includes columns that define the date range validity of the version (for example, **StartDate** and **EndDate**) and possibly a flag column (for example, **IsCurrent**) to easily filter by current dimension members.

## The Role of Hierarchies in Star Schemas

A hierarchy is a logical structure that organizes dimensional attributes into levels of granularity, such as country, state, city, or product category, subcategory, item. Hierarchies can help you drill down and filter data. To define hierarchies in a star schema, you can either use a single dimension table with multiple columns for each level, or use multiple dimension tables with foreign keys to link them. The choice depends on factors such as the number and depth of hierarchies, the data volume and frequency, and the query performance and complexity. In this lab, we will create hierarchies with attributes in one table.

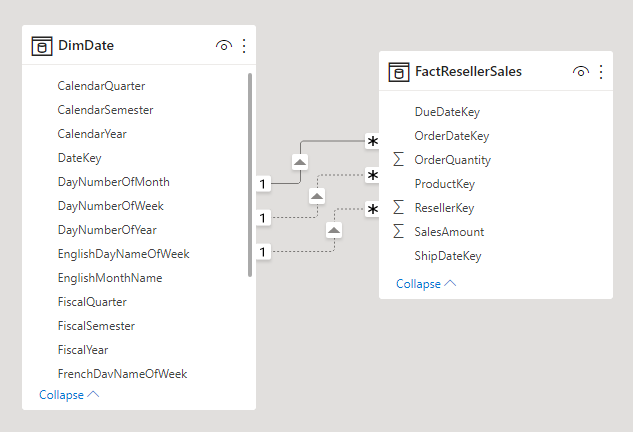
Source: Adapted end edited from LinkedIn

## Role-playing dimensions

A **role-playing dimension** is a dimension is the one that can have multiple relationships to the Fact Table. For example, a date dimension table can have three relationships to the sales fact table for dates such as order date, ship date, or delivery date.

In a data warehouse, the accepted design approach is to define a single date dimension table. At query time, the "role" of the date dimension is established by which fact column you use to join the tables. For example, when you analyze sales by order date, the table join relates to the reseller sales order date column.

In a Power BI model, this design can be imitated by creating multiple relationships between two tables. In the Adventure Works example, the date and reseller sales tables would have three relationships. While this design is possible, it's important to understand that there can only be one active relationship between two Power BI model tables. All remaining relationships must be set to inactive. Having a single active relationship means there is a default filter propagation from date to reseller sales. In this instance, the active relationship is set to the most common filter that is used by reports, which at Adventure Works is the order date relationship.



## Junk dimensions

(This section has been adapted from an article in Wikipedia. Credit: Wikipedia)

A junk dimension is a dimension table consisting of attributes that do not belong in the fact table or in any of the existing dimension tables. The nature of these attributes is usually text or various flags, e.g. non-generic comments or just simple yes/no or true/false indicators. These kinds of attributes are typically remaining when all the obvious dimensions in the business process have been identified and thus the designer is faced with the challenge of where to put these attributes that do not belong in the other dimensions.

One solution is to create a new dimension for each of the remaining attributes, but due to their nature, it could be necessary to create a vast number of new dimensions resulting in a fact table with a very large number of foreign keys. The designer could also decide to leave the remaining attributes in the fact table but this could make the row length of the table unnecessarily large if, for example, the attribute is a long text string.

The solution to this challenge is to identify all the attributes and then put them into one or several junk dimensions. One junk dimension can hold several true/false or yes/no indicators that have no correlation with each other, so it would be convenient to convert the indicators into a more describing attribute. An example would be an indicator about whether a package had arrived: instead of indicating this as “yes” or “no”, it would be converted into "arrived" or "pending" in the junk dimension. The designer can choose to build the dimension table so it ends up holding all the indicators occurring with every other indicator so that all combinations are covered. This sets up a fixed size for the table itself which would be 2x rows, where x is the number of indicators. This solution is appropriate in situations where the designer would expect to encounter a lot of different combinations and where the possible combinations are limited to an acceptable level. In a situation where the number of indicators are large, thus creating a very big table or where the designer only expects to encounter a few of the possible combinations, it would be more appropriate to build each row in the junk dimension as new combinations are encountered. To limit the size of the tables, multiple junk dimensions might be appropriate in other situations depending on the correlation between various indicators.

## Degenerate dimensions

*(This section has been adapted from an article in Wikipedia. Credit: Wikipedia)*

According to Ralph Kimball, in a data warehouse, a degenerate dimension is a dimension key in the fact table that does not have its own dimension table, because all the interesting attributes have been placed in other dimension tables.

Even though there is no corresponding dimension table of attributes, degenerate dimensions can be quite useful for grouping together related fact tables rows. For example, retail point-of-sale transaction numbers tie all the individual items purchased together into a single market basket. In health care, degenerate dimensions can group the claims items related to a single hospital stay or episode of care.

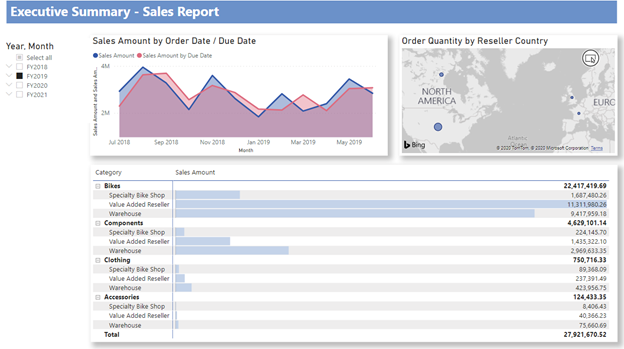
# LAB SECTION

## LAB: Multidimensional Data Modeling and Star Schemas in Power BI

**APPLIES TO:**  Power BI Desktop   Power BI service

In this tutorial, you start with a dimensional model and build a beautiful report from start to finish in 45 minutes. You work at AdventureWorks and your manager wants to see a report on your latest sales figures. They've requested an executive summary of:

* Which day had the most sales in February 2019?
* Which country/region is the company seeing the most success in?
* Which product category and reseller business types should the company continue to invest in?



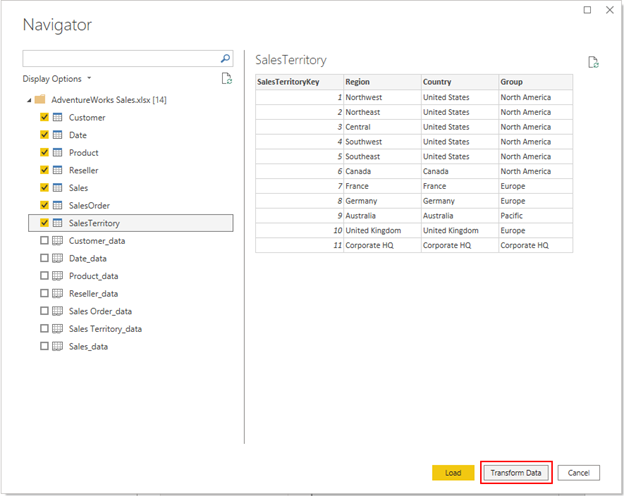
**Get data: Download the sample**

1. Download the **AdventureWorks Sales.xlsx** file from the datasets folder and save it in a folder of your preference on your own computer.
2. Start Power BI Desktop.
3. On the Ribbon at the top of the screen, click on "**Get Data**", select **Excel Workbook**, navigate to where you saved the "**AdventureWorks Sales**" Excel file, and select **Open**.

**Prepare your data**

In the Navigator pane, you have the option to ***transform* or *load*** the data. The Navigator provides a preview of your data so you can verify that you have the correct range of data. Numeric data types are italicized. In this tutorial, **we're going to transform** the data before loading.

Select all the tables in the left pane and choose **Transform Data**. Make sure not to select the sheets (labeled *\_data*) where \_data appears after the underscore.

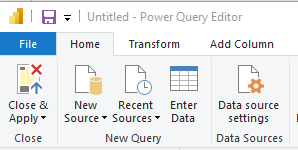


## Editing Data Types in Power BI Queries

Check that the data types of the columns proposed by PowerBI match the data types in the following table. To check the data type for a column in any of the queries in the left pane, select the query such as "**Customer**" which is the top one, then select each column referenced in the table below to check its data type. To check or change the data type for a query column, go to the Transform group on the **Home Ribbon** and click on **Data Type**.

| **Query** | **Column** | **Data type** |
| --- | --- | --- |
| **Customer** | CustomerKey | Whole Number |
| **Date** | DateKey | Whole Number |
|  | Date | Date |
|  | MonthKey | Whole Number |
| **Product** | ProductKey | Whole Number |
|  | Standard Cost | Decimal Number |
|  | List Price | Decimal Number |
| **Reseller** | ResellerKey | Whole Number |
| **Sales** | SalesOrderLineKey | Whole Number |
|  | ResellerKey | Whole Number |
|  | CustomerKey | Whole Number |
|  | ProductKey | Whole Number |
|  | OrderDateKey | Whole Number |
|  | DueDateKey | Whole Number |
|  | ShipDateKey | Whole Number |
|  | SalesTerritoryKey | Whole Number |
|  | Order Quantity | Whole Number |
|  | Unit Price | Decimal Number |
|  | Extended Amount | Decimal Number |
|  | Unit Price Discount Pct  **(Replace Current)** | Percentage |
|  | Product Standard Cost | Decimal Number |
|  | Total Product Cost | Decimal Number |
|  | Sales Amount | Decimal Number |
| SalesTerritory | SalesTerritoryKey | Whole Number |
| SalesOrder | SalesOrderLineKey | Whole Number |

Back on the **Home** tab, select **Close & Apply**.



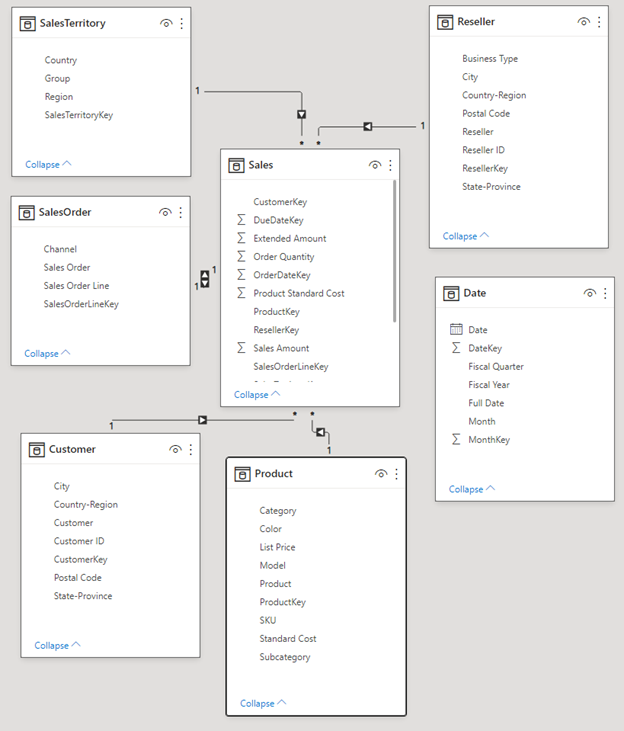
## Model your data

The data you loaded is almost ready for reporting. Let’s inspect the data model and make some changes.

Select the **Model View** in the left pane.



Your data model should look like the following image, with each table in a box.



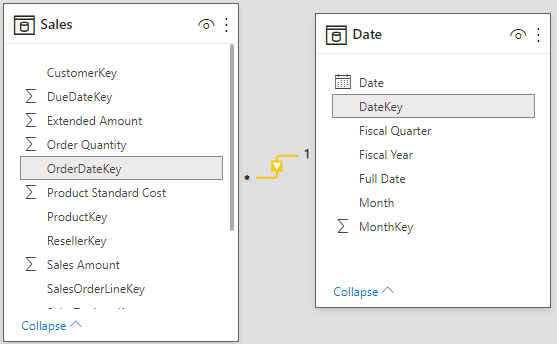
## Creating Relationships in Star Schemas

This model is a typical *star schema* that you might see in data warehouses and OLAP cubes: It resembles a star. The center of the star is the Fact table, in this case Sales, which contains numbers for the most part and the related tables are called Dimension tables which contain attributes and these dimensional tables are related to the Fact table with relationships. The Fact table contains numerical information about sales transactions, such as Sales Amount and Product Standard Cost. The Dimensions provide context (descriptors) so you can, among other things, analyze:

* What Product was sold...
* to which Customer...
* by which Reseller...
* in which Sales Territory.

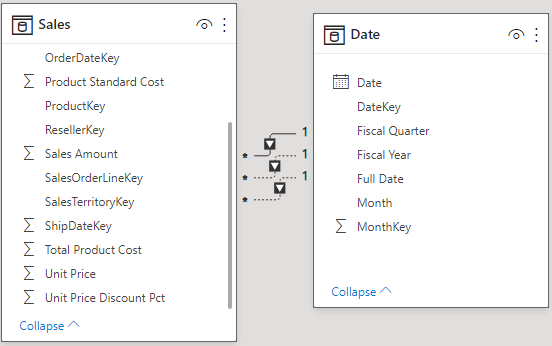
If you look closely, you notice that all Dimension tables are related to the Fact with a Relationship, except for the Date table. **Let’s add some relationships to Date now**.

1. **Drag the DateKey from the Date table to OrderDateKey on the Sales table**. You've created a so-called "one-to-many" relationship from Date to Sales, as indicated by the **1** and the asterisk \* (many) at the two ends of the line. The relationship is "one-to-many" because we have one or more Sales orders for a given Date. If each date had only one Sales order, the relationship would be "one-to-one". The little arrow in the middle of the line indicates the "cross-filtering direction." It indicates that you can use values from the Date table to filter the Sales table, so the relationship allows you to analyze when a Sales order was placed.



1. **Let’s add two more relationships to the Date table by dragging**:

* DateKey to DueDateKey
* DateKey to ShipDateKey



**You notice that the first relationship, on OrderDateKey, is active**, shown by the continuous line. The other two are inactive, shown by the dashed lines. Power BI uses the active relationship by default to relate the Sales Fact table with the Date dimension table. Hence a sum of SalesAmount is calculated by Order Date, not Due Date or Ship Date. You can alter this behavior.

## Working with Star Schema Key Columns

The typical star schema contains several keys that hold the relationships between Facts and Dimensions. Normally we don't want to use key columns in our reports. Let’s hide the key columns from view, so the Fields List shows fewer fields, and the data model is easier to use.

Go over all tables and hide any column whose name ends with *Key*:

Select the **Eye** icon next to the column and choose **Hide in report view**.

Screenshot of Visible column with Eye icon.

You can also select the **Eye** icon next to the column in the Properties pane.

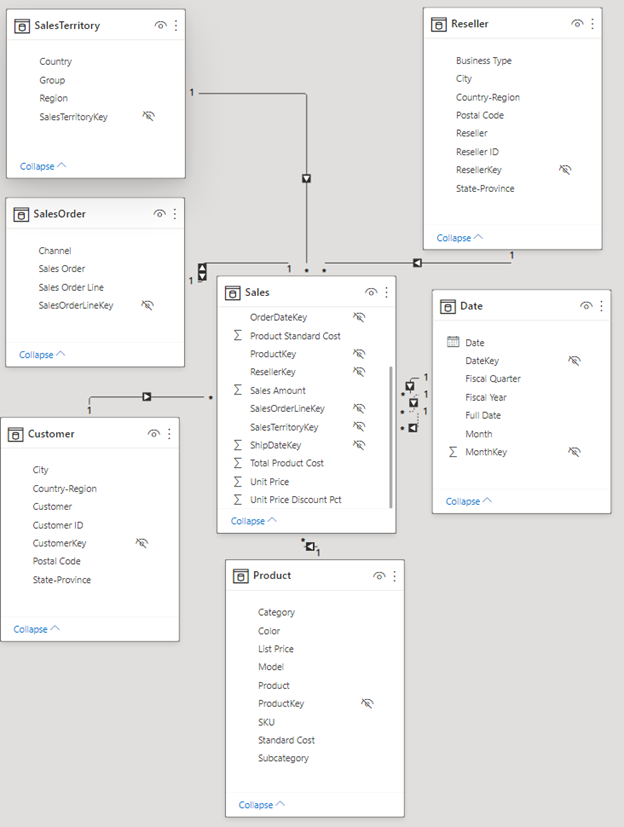
Hidden fields have this icon, an eye with a line through it.

Screenshot of Field with the hidden Eye icon.

Hide these fields.

| **Table** | **Column** |
| --- | --- |
| **Customer** | CustomerKey |
| **Date** | DateKey |
|  | MonthKey |
| **Product** | ProductKey |
| **Reseller** | ResellerKey |
| **Sales** | CustomerKey |
|  | DueDateKey |
|  | OrderDateKey |
|  | ProductKey |
|  | ResellerKey |
|  | SalesOrderLineKey |
|  | SalesTerritoryKey |
|  | ShipDateKey |
| **SalesOrder** | SalesOrderLineKey |
| **SalesTerritory** | SalesTerritoryKey |

Your data model should now look like this data model, with relationships between Sales and all the other tables, and all the key fields hidden:



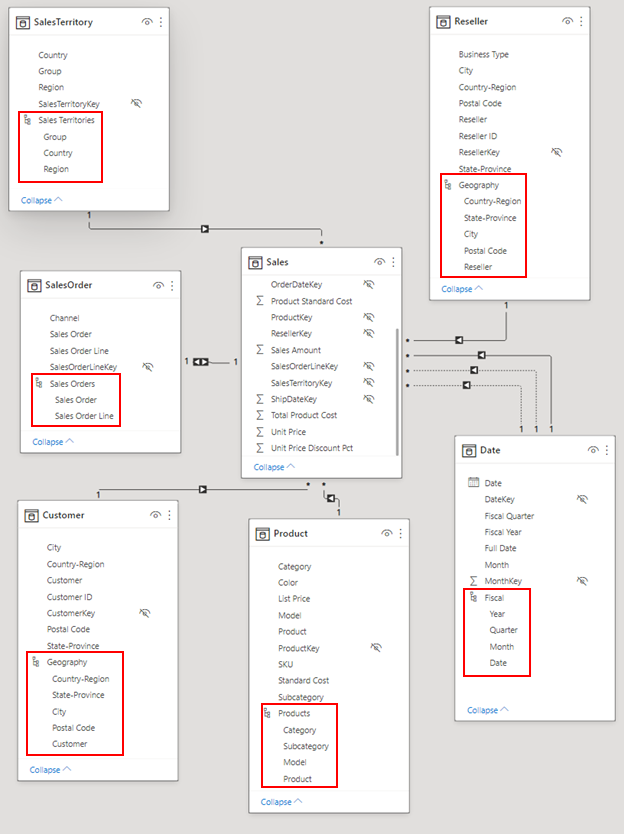
## Setup Hierarchies in Star Schemas

Hierarchies enable easier navigation of groupings. For example, cities are in a State which is in a Country. **Hierarchies enable Data Drill-Through**. For each hierarchy that you will create in this lab, follow the steps below:

1. In the Data Pane, expand the fields in the table. Right-click the highest-level field in the hierarchy and choose **Create hierarchy**. For example, in the Customer table, right click the "Country-Region" field and select "Create Hierarchy". PBI will create the hierarchy and you will see it in the table.
2. To add a field to a Hierarchy, right click on it, select "**Add to Hierarchy**" and choose the Hierarchy in which you want to include it. For example, right click on the "State-Province" field, select "Add to Hierarchy", and select the Country-Region Hierarchy.
3. You will need to **change the default names of the hierarchies** you have created. To rename a hierarchy, right click on its name and select “Rename”. Below are the hierarchies you need to create for this lab:

| **Table** | **Hierarchy name** | **Levels** | **Clarification** |
| --- | --- | --- | --- |
| **Customer** | **Geography** | Country-Region | Right click this field to create the hierarchy. |
|  |  | State-Province | Add this field to the hierarchy. |
|  |  | City | Add this field to the hierarchy. |
|  |  | Postal Code | Add this field to the hierarchy. |
|  |  | Customer | Add this field to the hierarchy. |
|  |  |  |  |
| **Date** | **Fiscal** | Fiscal Year | Right click this field to create the hierarchy. |
|  |  | Fiscal Quarter | Add this field to the hierarchy. |
|  |  | Month | Add this field to the hierarchy. |
|  |  | Date | Add this field to the hierarchy. |
|  |  |  |  |
| **Product** | **Products** | Category | Right click this field to create the hierarchy. |
|  |  | Subcategory | Add this field to the hierarchy. |
|  |  | Model | Add this field to the hierarchy. |
|  |  | Product | Add this field to the hierarchy. |
|  |  |  |  |
| **Reseller** | **Geography** | Country-Region | Right click this field to create the hierarchy. |
|  |  | State-Province | Add this field to the hierarchy. |
|  |  | City | Add this field to the hierarchy. |
|  |  | Postal Code | Add this field to the hierarchy. |
|  |  | Reseller | Add this field to the hierarchy. |
|  |  |  |  |
| **SalesOrder** | **Sales Orders** | Sales Order | Right click this field to create the hierarchy. |
|  |  | Sales Order Line | Add this field to the hierarchy. |
|  |  |  |  |
| **SalesTerritory** | **Sales Territories** | Group | Right click this field to create the hierarchy. |
|  |  | Country | Add this field to the hierarchy. |
|  |  | Region | Add this field to the hierarchy. |

**Do not forget to rename your hierarchies** after you have created them using the names in the table above. Now, your data model should look like the following. It has the same tables, but each dimension table contains a hierarchy:



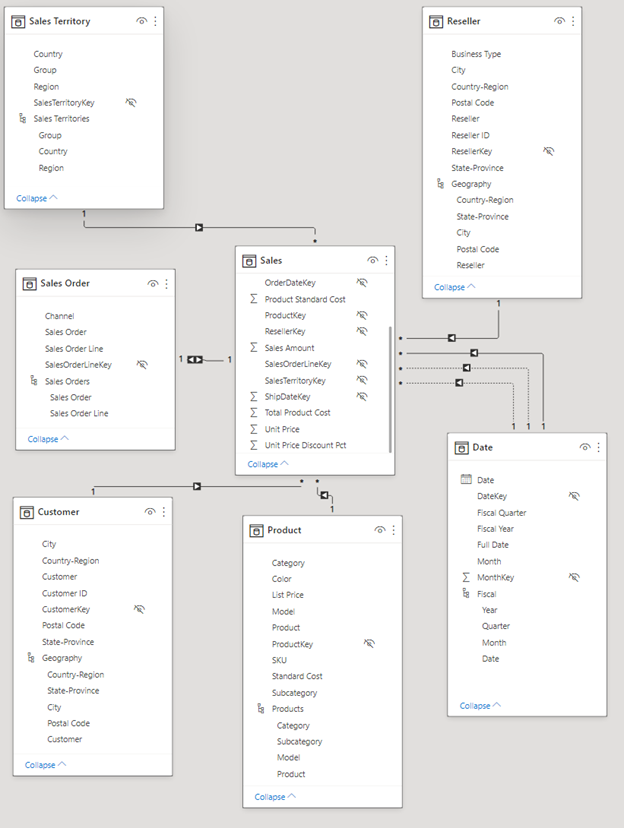
## Rename tables

To finish modeling, let's rename the following tables in the Properties pane:

| **Old table name** | **New table name** |
| --- | --- |
| SalesTerritory | Sales Territory |
| SalesOrder | Sales Order |

This step is necessary because Excel table names can't contain spaces.

Now your final data model is ready.



## Visual: Sum of Sales Amount by Date Line Chart

You will create a line chart to see which month and year had the highest sales amount.

1. From the data pane, in the **Sales** fact table (where the measures are), drag the **Sales Amount** field to a blank area on the report canvas. By default, Power BI displays a column chart with one column, Sales Amount.
2. Drag the **Fiscal** hierarchy in the **Date** table and drop it on the column chart.

A graph of sales

Description automatically generated

1. In the **Fields** section of the **Visualizations** pane, in the X-axis box, remove the **Fiscal Year**, **Fiscal Quarter,** and **Date** fields:

Screenshot of In the Fields section of the Visualizations pane, remove the Year and Quarter fields.

1. In the Visualizations pane, change the visualization type to **Area Chart**.
2. In the **Visualizations** pane, click on the "**Format Your Visual**" tab, expand the **Lines** section, and in **Colors**, change the **Sum of** **Sales Amount by Month** to a more contrasting color, such as red.

A graph showing a graph of sales

Description automatically generated with medium confidence

## Create a Map of the Sum of Order Quantity by Reseller Country-Region

Now we'll create a map to see in which Country or Region the Resellers have the highest Order Quantity Amount.

1. In the Data pane, from the **Reseller** table, drag the **Country-Region** field to a blank area on your report canvas. **Power BI creates a map**.
2. Drag the **Order Quantity** field from the **Sales** table and drop it on the map. Make sure **Country-Region** is in the **Location** well and the **Sum of** **Order Quantity** in the **Bubble size** well in the visualizations pane.

A screenshot of a computer

Description automatically generated

## Visual: Sum of Sales Amount by Product Category and Reseller Business type

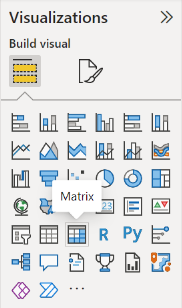
Next, we create a column chart to investigate which products are sold by what type of reseller business.

1. Drag the two charts you've created to be side by side in the top half of the canvas. Save some room on the left side of the canvas.
2. Click on a blank area in the lower half of your report canvas.
3. In the Fields pane, select **Sales Amount** from **Sales**, **Category** from **Product**, and **Business type from Reseller.**

A graph of different colored bars

Description automatically generated

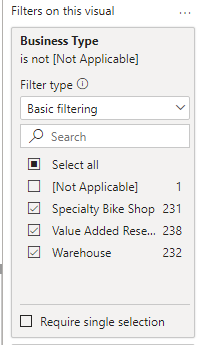
Power BI automatically creates a clustered column chart. Change the visualization to a **Matrix**:



A screenshot of a computer

Description automatically generated

1. Drag the **Business Type** field from the **Columns well**, and put it under **Category** in **Rows**.
2. With the matrix still selected, in the Filters pane, under **Business Type**, **Select all**, then clear the **[Not Applicable]** box.



A screenshot of a graph

Description automatically generated

1. Drag the matrix so it's wide enough to fill the space under the two upper charts.
2. In the visualizations pane, in the "**Format Your Visual**" tab for the matrix, in the search text box enter "*conditional"*. Turn on the **Data bars**. In data bars, click on the **fx** option, and set a lighter color for the positive bar. Select **OK**.
3. Increase the width of the Sum of Sales Amount column so it fills the whole area by dragging the matrix. Expand the Accessories, Bikes, and Clothing categories.

A screenshot of a video

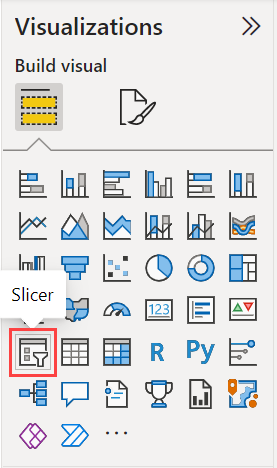
Description automatically generated

It looks like Bikes have a higher Sales Amount overall.

## Create a Fiscal calendar slicer

Slicers are a valuable tool for filtering the visuals on a report page to a specific selection. In this case, we can create a slicer to narrow in on performance for each month, quarter, and year.

1. Make sure to click on a blank spot of your report. Modify the size of any of the visuals if needed to create blank space in your report.
2. Now, in the Visualizations pane, in the “Build Your Visual” tab select **Slicer** from the available visuals.



1. Drag the Fiscal Hierarchy from the Date table on the Slicer. Your slicer will be similar to the one in the image below. **If your slicer is not exactly the same like the one below, it is absolutely fine as soon as you can see the fiscal years**.

A screenshot of a computer

Description automatically generated

1. We want the fiscal years to appear in rows and not next to each other. We also want to change the fiscal year rectangles to rounded rectangles. Make sure the slicer visual is selected or click on it to select it. Then:  
   1. In the Visualizations pane, click the “Format Visual” tab. Expand the “Shape” heading and change the “Shape” property from “Rectangle” to “Rounded Rectangle”. Collapse the “Shape” heading so that its properties do not show.
   2. Expand the “Layout” heading and change its properties to match the ones in the image below:

A screenshot of a computer

Description automatically generated

Now your slicer should be similar to the one in the image below:

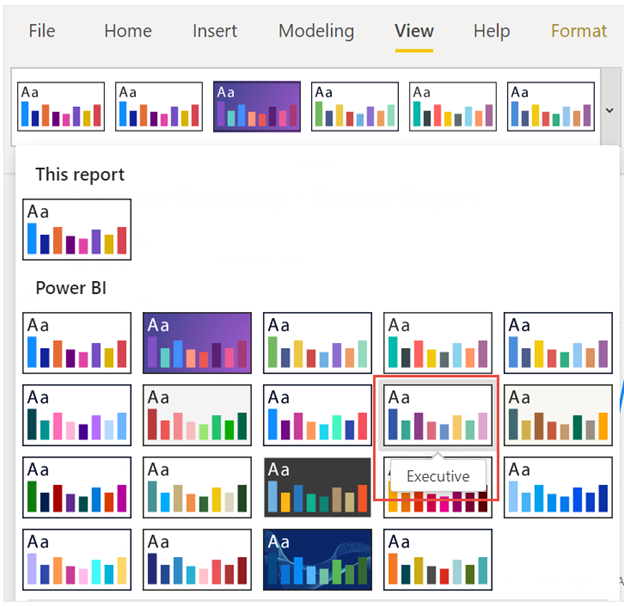
A screenshot of a phone

Description automatically generated

Save your Power BI report.

## Formatting Report Themes

* On the **View** ribbon, select **Themes**, and change the theme to **Executive**.



## Edit the "Sum of Order Quantity by Country-Region" visual

1. Select the visual by clicking on it.
2. In the Visualizations Pane, in the “Format Your Visual” tab, expand the **Map settings** section and change the **Style** to **Grayscale**.
3. On the **General** tab, in the **Title** section, change **Text** to "Order Quantity by Reseller Country-Region".
4. Set **Text size** to **16 pt**.
5. Also on the **General** tab, toggle **Shadow** to **On**.

**Edit the "Sales Amount by Product Category and Reseller Business Type" Visual**

1. Select the "Sales Amount by Product Category and Reseller Business Type" visual.
2. Click on the **General** tab of the “Format Your Visual” tab and in the **Title** section, change the **Text** to "Sales Amount by Product Category and Reseller Business Type". Enable the title, if it not enabled.
3. Set **Text size** to **16 pt**.
4. Also in the **General** tab, expand the **Effects**, and set the **Shadow** to **On**.

Select **FY2020** in the slicer. Your report should be similar, not identical, to the report image below:

A screenshot of a computer

Description automatically generated