Exercise 2

In this exercise, you will create calculated tables, calculated columns, and simple measures using Data Analysis Expressions (DAX).
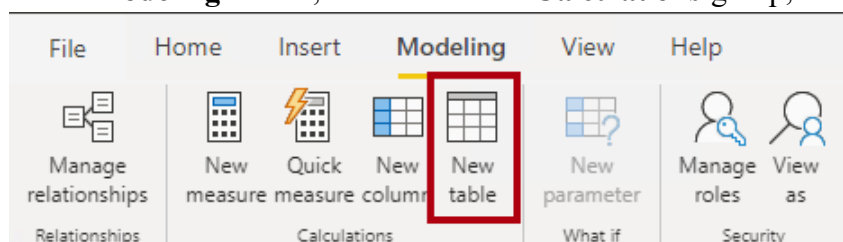
1.1 Create Calculated Tables

You will create two calculated tables. The first will be the *Salesperson* table, to allow a direct relationship between it and the *Sales* table. The second will be the *Date* table.

A calculated table is created by first entering the table name, followed by the equals symbol (=), followed by a DAX formula that returns a table. The table name cannot already exist in the data model. The formula bar supports entering a valid DAX formula. It includes features like auto-complete, Intellisense and color-coding, enabling you to quickly and accurately enter the formula.
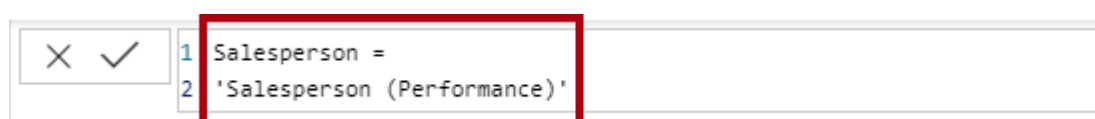
1.1.1 Create the Salesperson Table

In this task, you will create the *Salesperson* table (direct relationship to Sales). This table definition creates a copy of the *Salesperson (Performance)* table. It copies the data only, however properties like visibility, formatting, etc. are not copied.
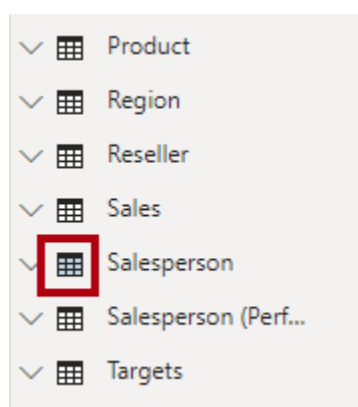
1. Select **File ➜ Open Report**, and open *Exercise 2.pbix*.

2. In the **Modeling** ribbon, from inside the **Calculations** group, click **New Table**.
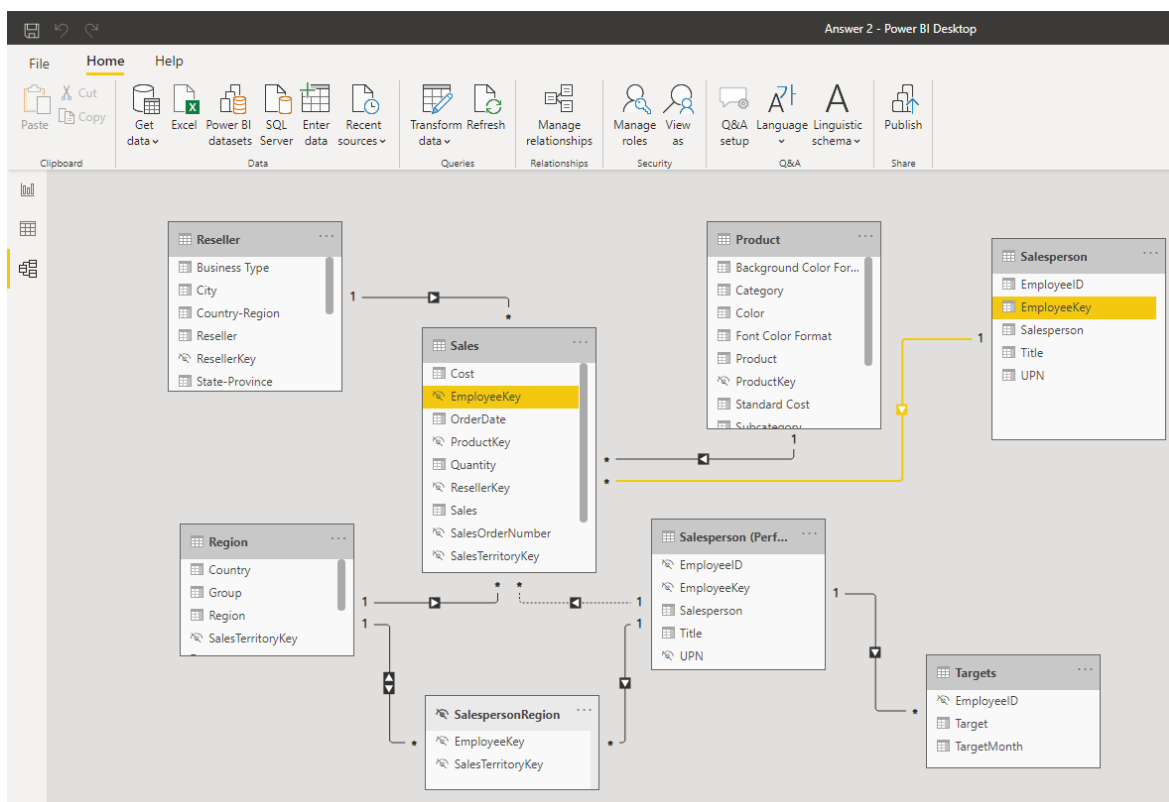

3. In the formula bar, type *Salesperson* =, press **[Shift]** + **[Enter]**, type *'Salesperson (Performance)'*, and then press **[Enter]**.
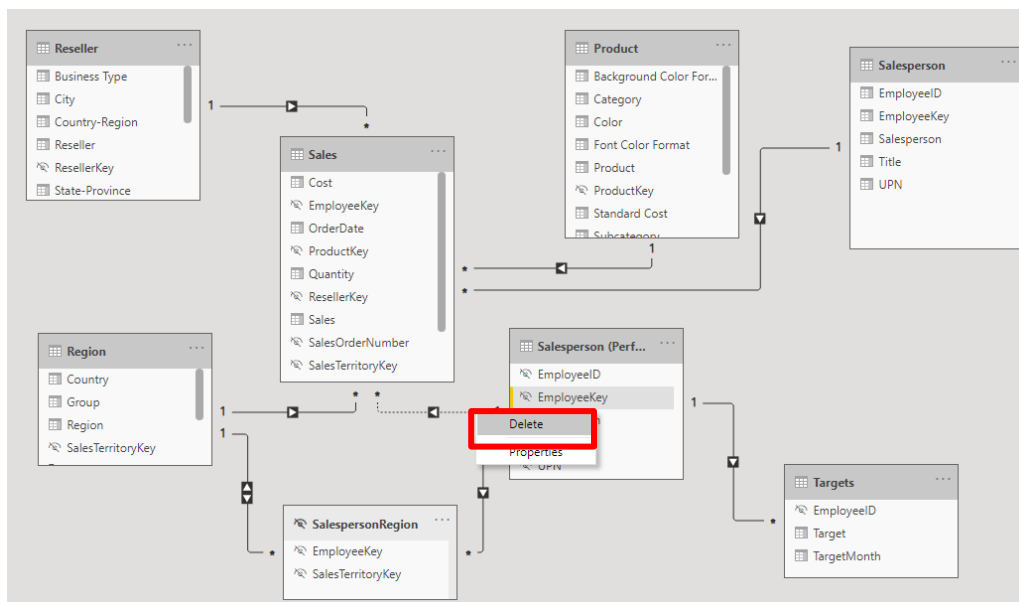

4. In the **Fields** pane, notice that the table icon is a shade of blue (denoting a calculated table).
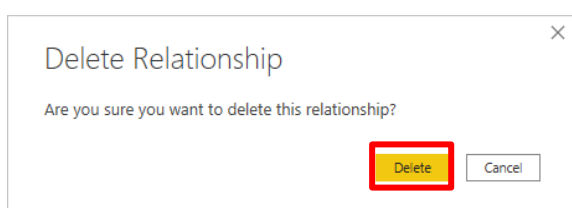
5. Switch to **Model** view. Create a relationship from the *Salesperson ➔ EmployeeKey* column to the *Sales ➔ EmployeeKey* column.

6. Right-click the inactive relationship between the *Salesperson (Performance)* and *Sales* tables, and then select **[Delete]**.
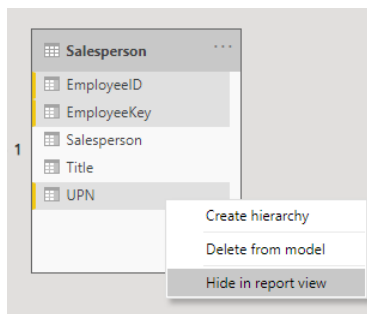

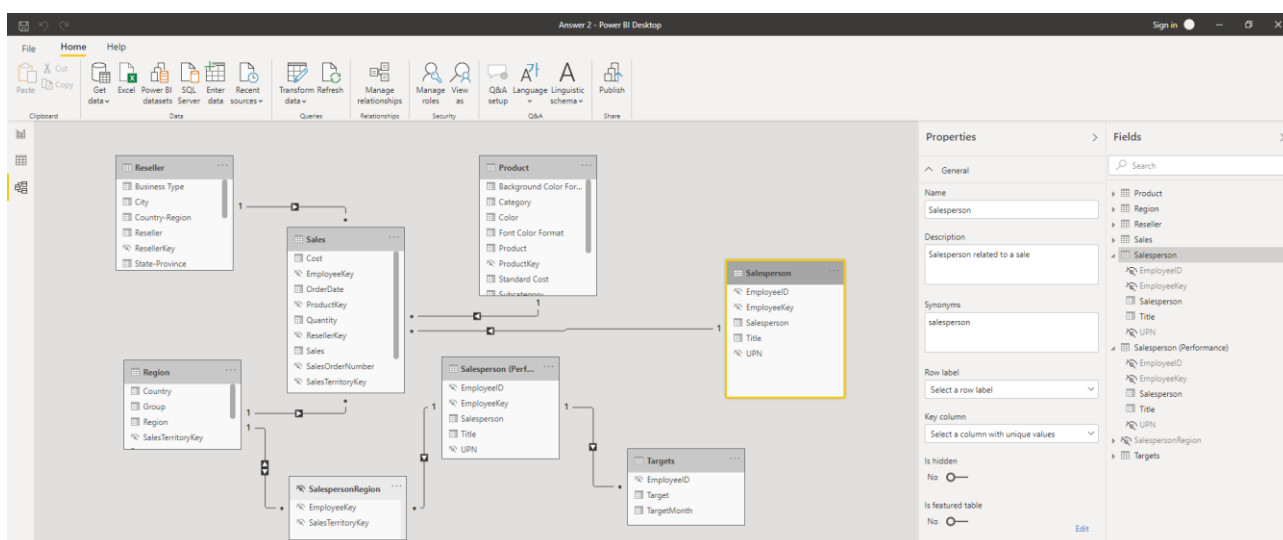7. When prompted to confirm the deletion, click **[Delete]**.

8. In the *Salesperson* table, multi-select the following columns, and then hide them by selecting **Hide in report view**:
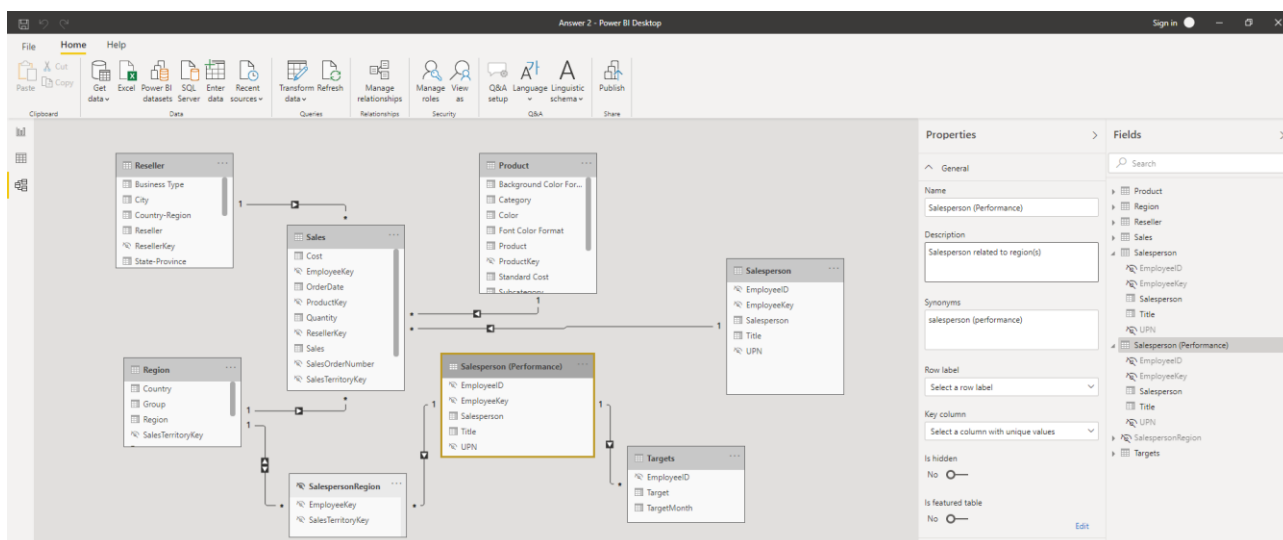
   - EmployeeID
   - EmployeeKey
   - UPN

9. Select the *Salesperson* table, in the **Properties** pane, in the **Description** box, enter: **Salesperson related to a sale**.

10. Select the *Salesperson (Performance)* table, set the **description** to: **Salesperson related to region(s)**.
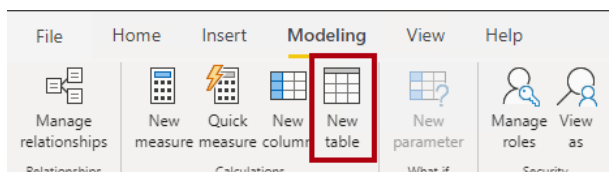
1.1.2 Create the Date table

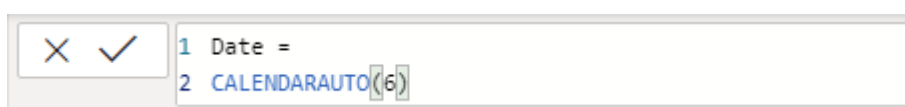In this task, you will create the *Date* table.

1. Switch to **Data** view.

2. In the **Modeling** ribbon, from inside the **Calculations** group, click **New Table**.

3. In the formula bar, enter the following, and then press Enter.

   ```
   1 Date =
   2 CALENDARAUTO(6)
   ```

   The **CALENDARAUTO** function returns a single-column table consisting of date values. The "auto" behavior scans all data model date columns to determine the earliest and latest date values stored in the data model. It then creates one row for each date within this range, extending the range in either direction to ensure full years of data is stored.

   This function can take a single optional argument which is the last month number of a year. In this case 6 is entered, meaning that June is the last month of the year.

4. In the **Fields** pane, select the **Date** table. At the bottom-left corner, in the status bar, notice the table statistics, confirming that 1826 rows of data have been generated, which represents five full years' data.
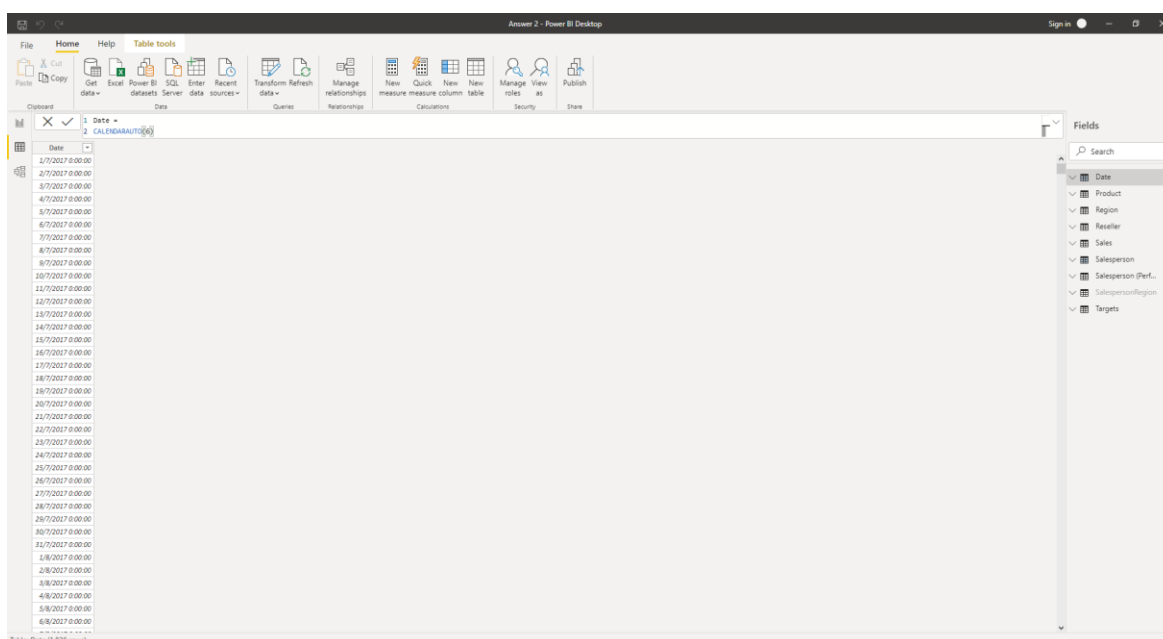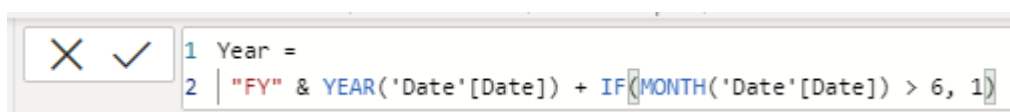
1.1.3 Create Calculated Columns

In this task, you will add additional columns to enable filtering and grouping by different time periods. You will also create a calculated column to control the sort order of other columns.
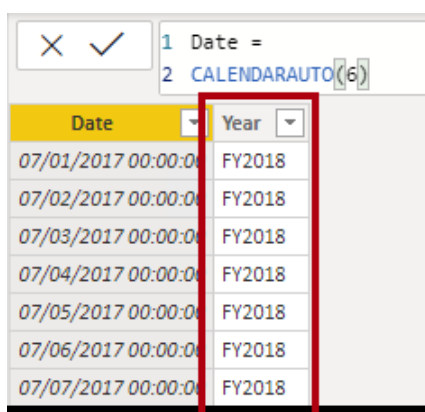
1. Select the *Date* table. On the **Table Tools** contextual ribbon, from inside the **Calculations** group, click **New Column**.
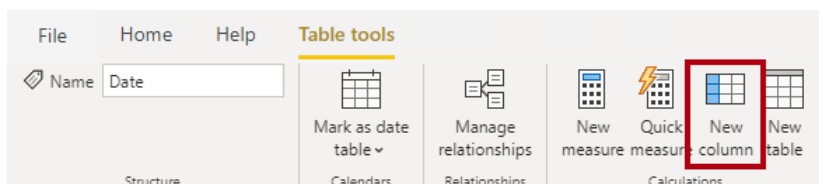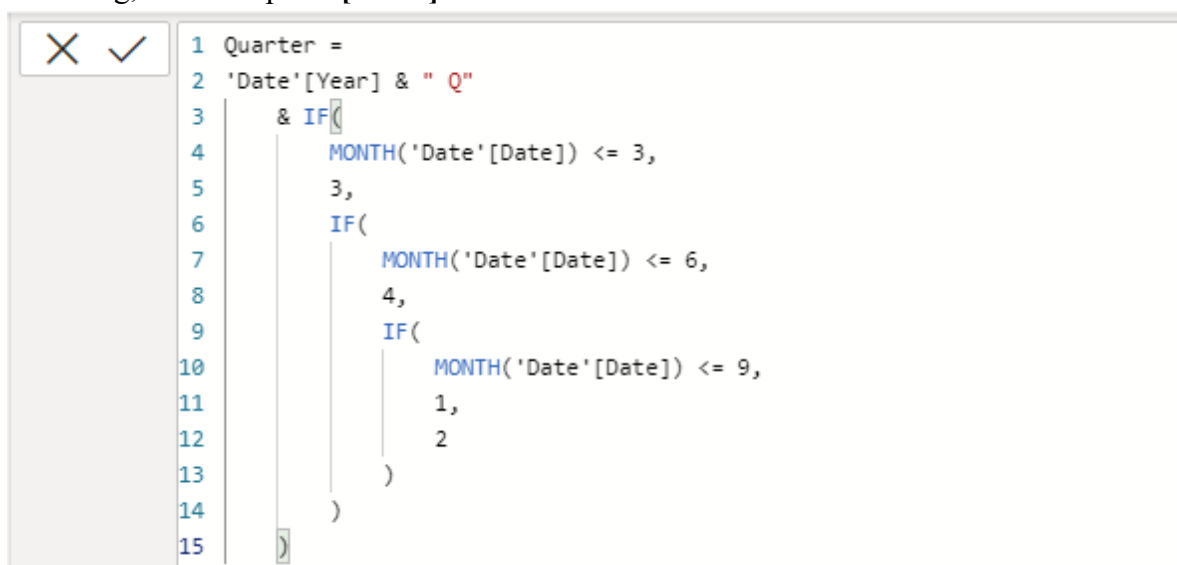

2. In the **formula** bar, type the following, and then press **[Enter]**. The formula uses the date's year value but adds one to the year value when the month is after June. This is how fiscal years at Adventure Works are calculated.



```
1 Year =
2 "FY" & YEAR('Date'[Date]) + IF(MONTH('Date'[Date]) > 6, 1)
```

3. Verify that the new column was added.


```
1 Date =
2 CALENDARAUTO(6)
```

Date	Year
07/01/2017 00:00:0(	FY2018
07/02/2017 00:00:0(	FY2018
07/03/2017 00:00:0(	FY2018
07/04/2017 00:00:0(	FY2018
07/05/2017 00:00:0(	FY2018
07/06/2017 00:00:0(	FY2018
07/07/2017 00:00:0(	FY2018

4. In order to create the *Quarter* column, select the *Date* table. On the **Table Tools** contextual ribbon, from inside the **Calculations** group, click **New Column**. In the **formula** bar, type the following, and then press **[Enter]**.

```
1  Quarter =
2  'Date'[Year] & " Q"
3      & IF(
4          MONTH('Date'[Date]) <= 3,
5          3,
6          IF(
7              MONTH('Date'[Date]) <= 6,
8              4,
9              IF(
10                 MONTH('Date'[Date]) <= 9,
11                 1,
12                 2
13             )
14         )
15     )
```

5. In order to create the *Month* column, select the *Date* table. On the **Table Tools** contextual ribbon, from inside the **Calculations** group, click **New Column**. In the **formula** bar, type the following, and then press **[Enter]**.

```
1 Month =
2 FORMAT('Date'[Date], "yyyy MMM")
```
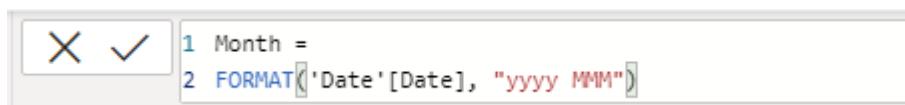
6. Verify that the new columns were added.

Date	Year	Quarter	Month
07/01/2017 00:00:00	FY2018	FY2018 Q1	2017 Jul
07/02/2017 00:00:00	FY2018	FY2018 Q1	2017 Jul
07/03/2017 00:00:00	FY2018	FY2018 Q1	2017 Jul
07/04/2017 00:00:00	FY2018	FY2018 Q1	2017 Jul
07/05/2017 00:00:00	FY2018	FY2018 Q1	2017 Jul
07/06/2017 00:00:00	FY2018	FY2018 Q1	2017 Jul
07/07/2017 00:00:00	FY2018	FY2018 Q1	2017 Jul

7. Switch to **Report** view and create a new page. Select the **matrix** visual type in the **Visualizations** pane.
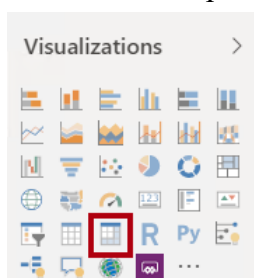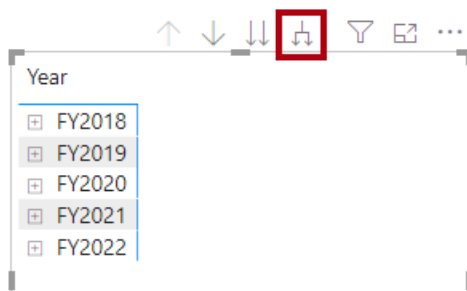
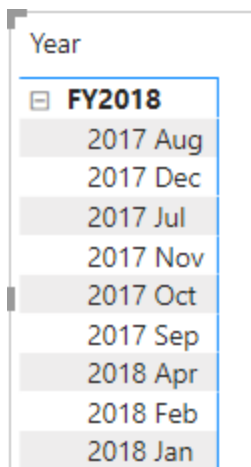8. In the **Fields** pane, from inside the *Date* table, drag the *Year* field into the **Rows** well.
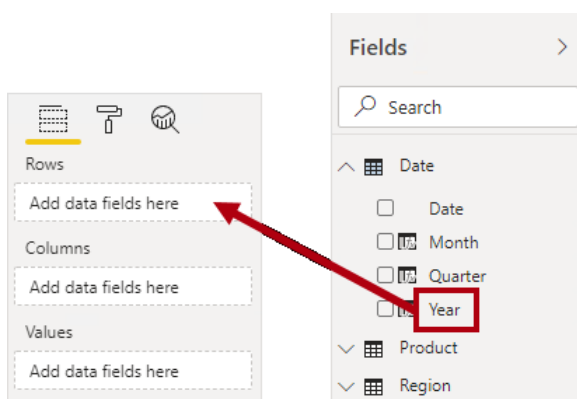
9. Drag the *Month* field into the **Rows** well, directly beneath the *Year* field.
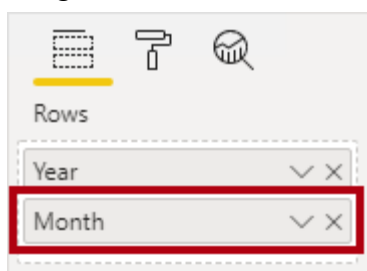
10. Next to the **Matrix** visual, click the forked-double arrow icon (which will expand all years down one level).

11. Notice that the years expand to months, and that the months are sorted alphabetically rather than chronologically.


12. To customize the *Month* field sort order, switch to **Data** view. Select the *Date* table. On the **Table Tools** contextual ribbon, from inside the **Calculations** group, click **New Column.** In the **formula** bar, add the *MonthKey* column to the *Date* table. This formula computes a numeric value for each year/month combination.

```
1 MonthKey =
2 (YEAR('Date'[Date]) * 100) + MONTH('Date'[Date])
```
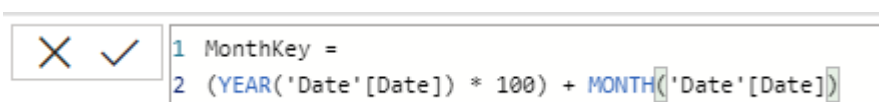
13. In Data view, verify that the new column contains numeric values (e.g. 201707 for July 2017, etc.).

```
1 MonthKey =
2 (YEAR('Date'[Date]) * 100) + MONTH('Date'[Date])
```

Date	Year	Quarter	Month	MonthKey
07/01/2017 00:00:00	FY2018	FY2018 Q1	2017 Jul	201707
07/02/2017 00:00:00	FY2018	FY2018 Q1	2017 Jul	201707
07/03/2017 00:00:00	FY2018	FY2018 Q1	2017 Jul	201707
07/04/2017 00:00:00	FY2018	FY2018 Q1	2017 Jul	201707
07/05/2017 00:00:00	FY2018	FY2018 Q1	2017 Jul	201707
07/06/2017 00:00:00	FY2018	FY2018 Q1	2017 Jul	201707
07/07/2017 00:00:00	FY2018	FY2018 Q1	2017 Jul	201707

14. In the **Fields** pane, ensure that the *Month* field is selected (when selected, it will have a dark gray background). On the **Column Tools** contextual ribbon, from inside the **Sort** group, click **Sort by Column**, and then select *MonthKey*.
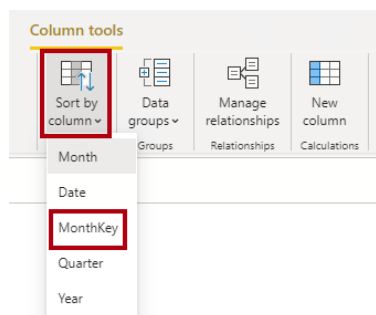

15. In the matrix visual, notice that the months are now chronologically sorted.
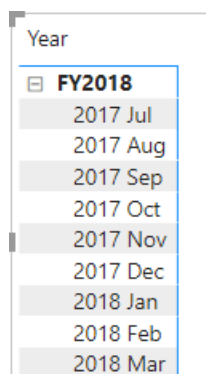
1.1.4 Complete the Date Table

In this task, you will complete the design of the *Date* table by hiding a column and creating a hierarchy. You will then create relationships to the *Sales* and *Targets* tables.

1. Switch to **Model** view, hide the *MonthKey* column in the *Date* table by set **Is hidden** to **Yes** in **Properties**.
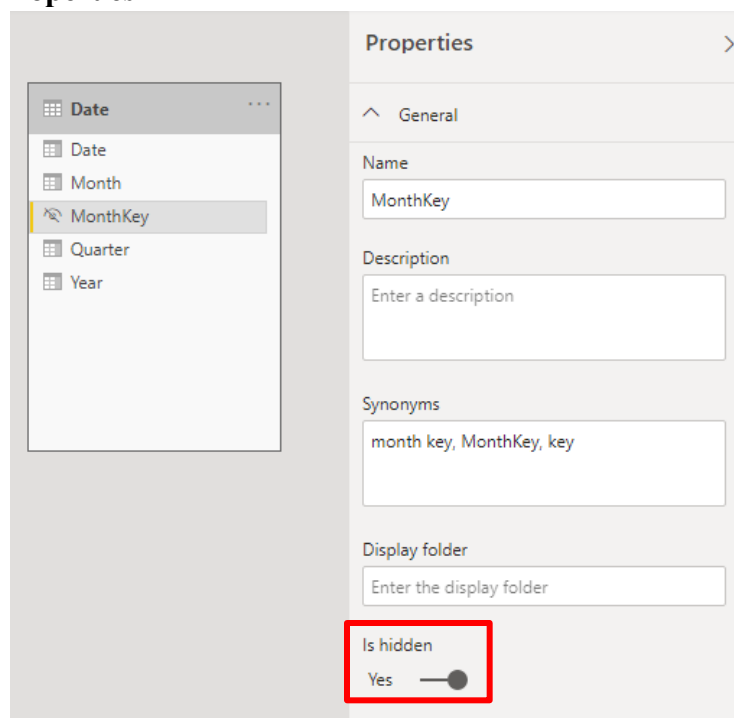
2. Now we would like to create a hierarchy *Fiscal* with the following three levels in the *Date* table:

- Year
- Quarter
- Month

First, right click the field *Date* ➔ *Year*, and select **Create hierarchy**.
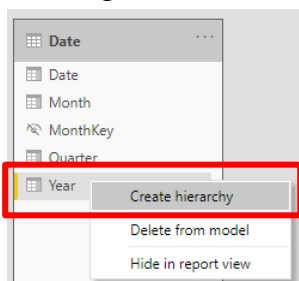
3. Change the **Name** to *Fiscal* in **Properties**.
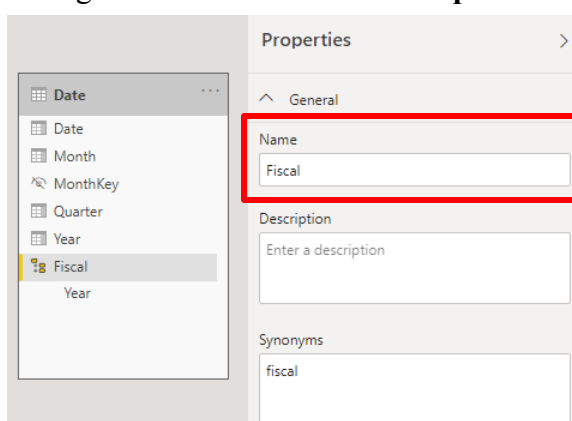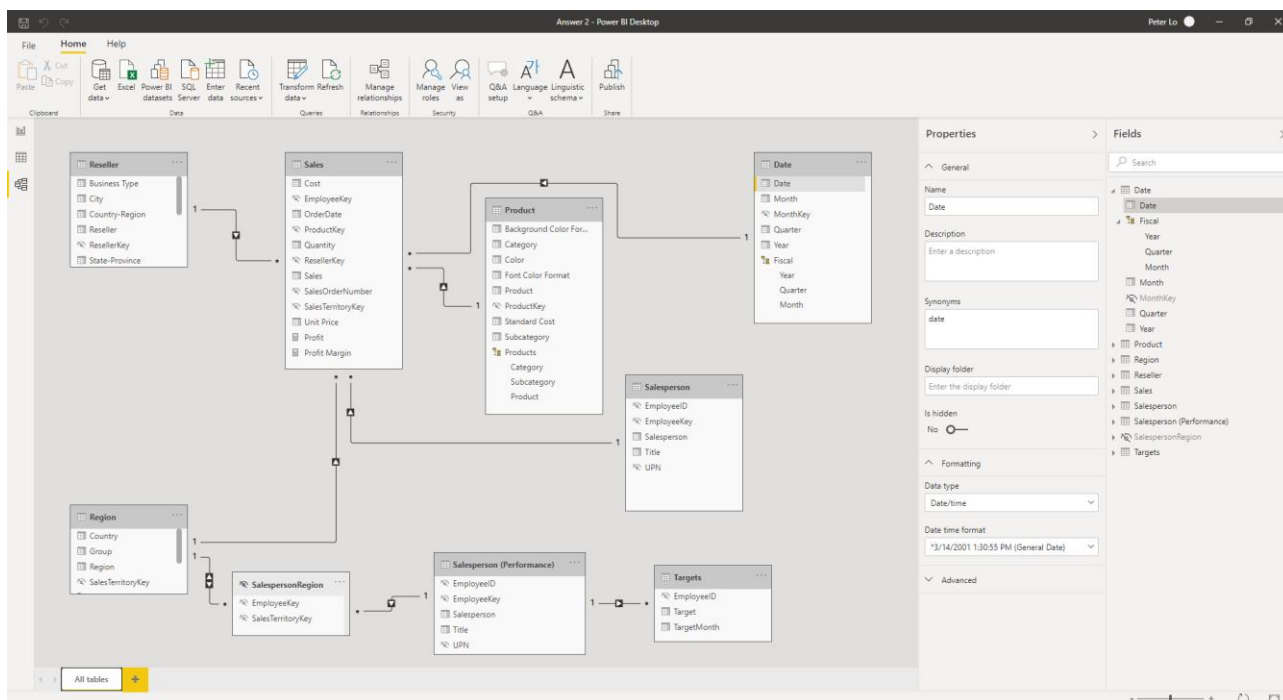

4. Select *Quarter* in **hierarchy** in **Properties**.
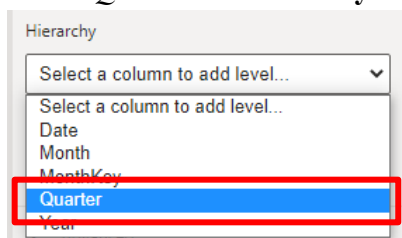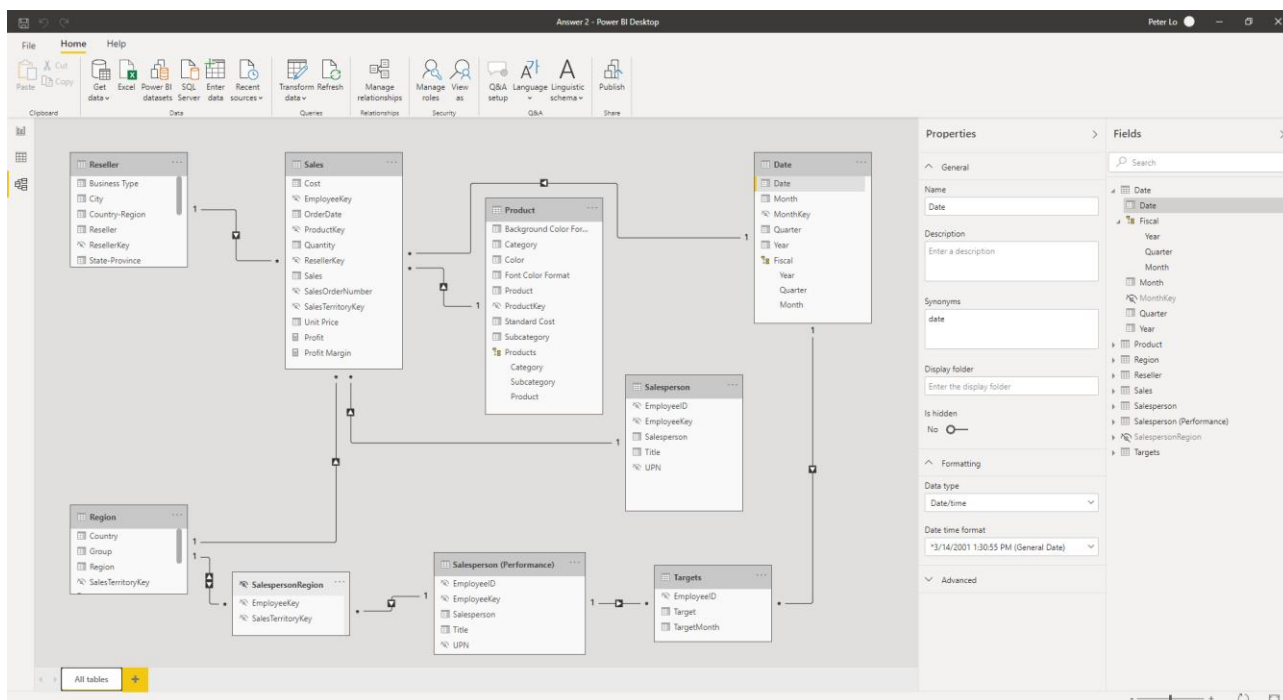

5. Select *Month* in **hierarchy** in **Properties**.
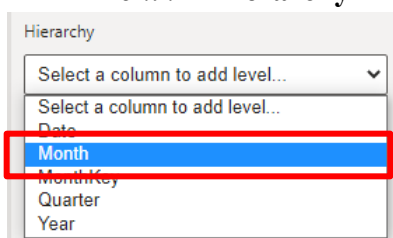
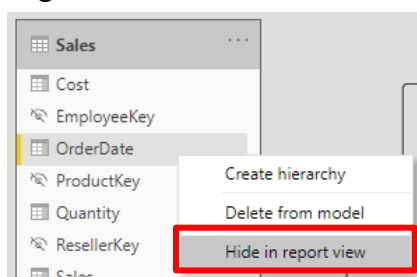6. Click **Apply Level Changes** to confirm.
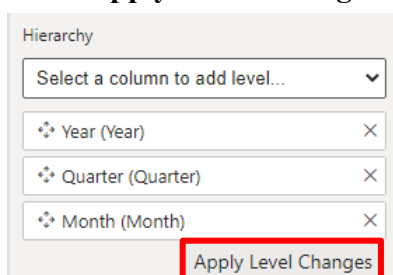
7. In order to create the model relationship, drag *Date* ➔ *Date* to *Sales* ➔ *OrderDate*.

8. Then create another model relationship, *Date* ➔ *Date* to *Targets* ➔ *TargetMonth*.

9. Right click the field *OrderDate* in *Sales* table, and select **Hide in report view**.

10. Right click the field *TargetMonth* in *Target* table, and select **Hide in report view**.
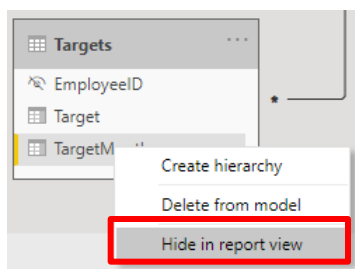

1.1.5   Mark the Date Table

In this task, you will mark the *Date* table as a date table.

1.   Switch to **Report** view. In the *Fields* pane, right click the *Date* table, and select **Mark as date table** ➔ **Mark as date table**.
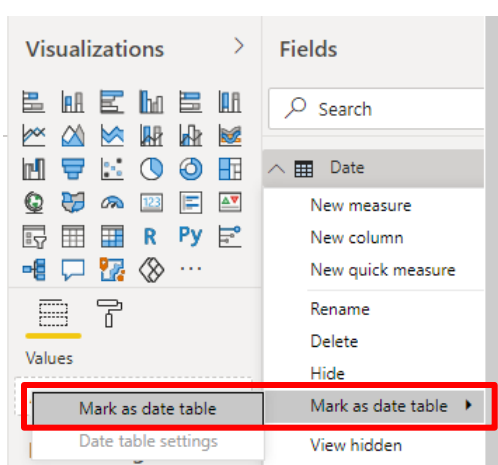
2.   In the **Mark as Date Table** window, in the **Date Column** dropdown list, select *Date* and then click **[OK]**. Power BI Desktop now understands that this table defines date (time). This is important when relying on time intelligence calculations.
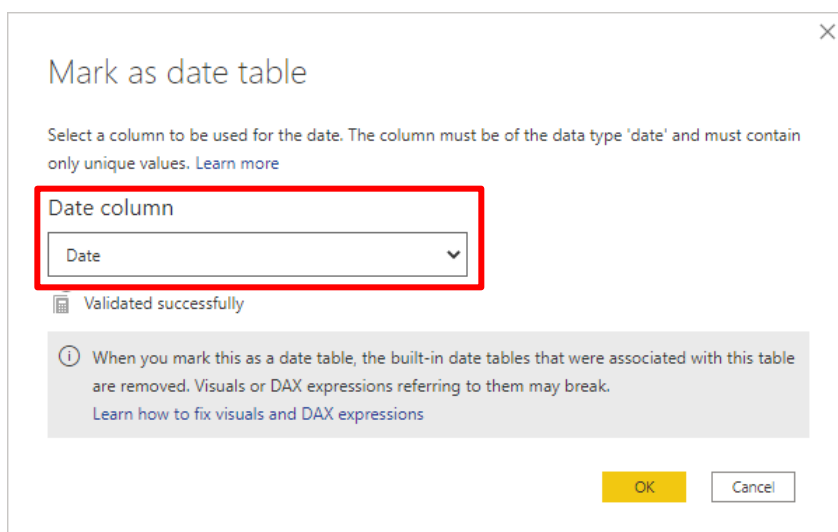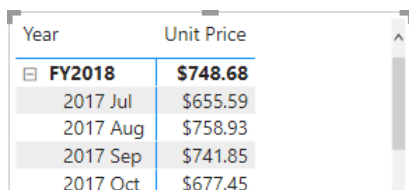
1.2 Create Measures

In this exercise, you will create and format several measures.

1.2.1   Create Simple Measures

In this task, you will create simple measures. Simple measures aggregate a single column or table.

1. Go to **Page 2** of the **Report** view. In the **Fields** pane, drag the *Sales* ➔ *Unit Price* field into the **Values** section in **Matrix** visual.
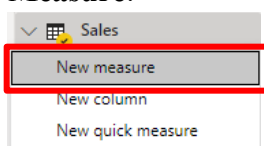
Year	Unit Price
☐ **FY2018**	**$748.68**
2017 Jul	$655.59
2017 Aug	$758.93
2017 Sep	$741.85
2017 Oct	$677.45

2. In the **Visual** fields pane, in the **Values** well, click the down-arrow for *Unit Price*, and then select **Average**.
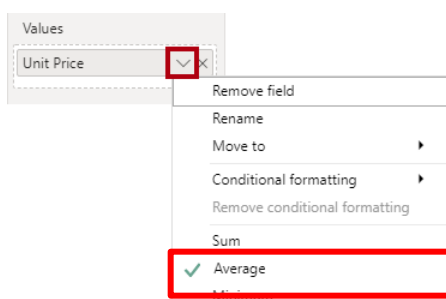
3. To create a measure, in the **Fields** pane, right-click the *Sales* table, and then select **New Measure**.
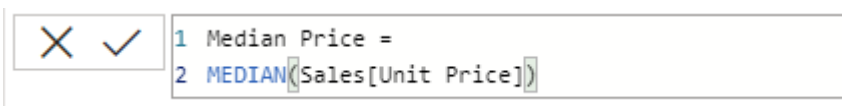
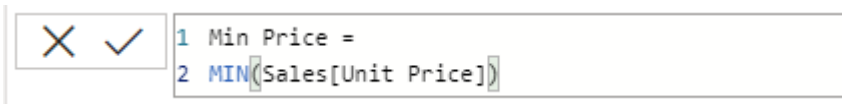4. In the formula bar, add the following measure definition:

```
1 Avg Price = AVERAGE(Sales[Unit Price])
```
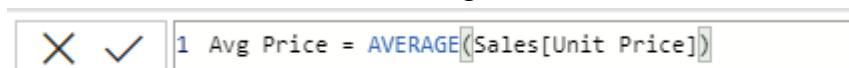
5. Add the *Avg Price* measure to the **Values** section in the **Matrix** visual. Notice that it produces the same result as the Unit Price column (but with different formatting)
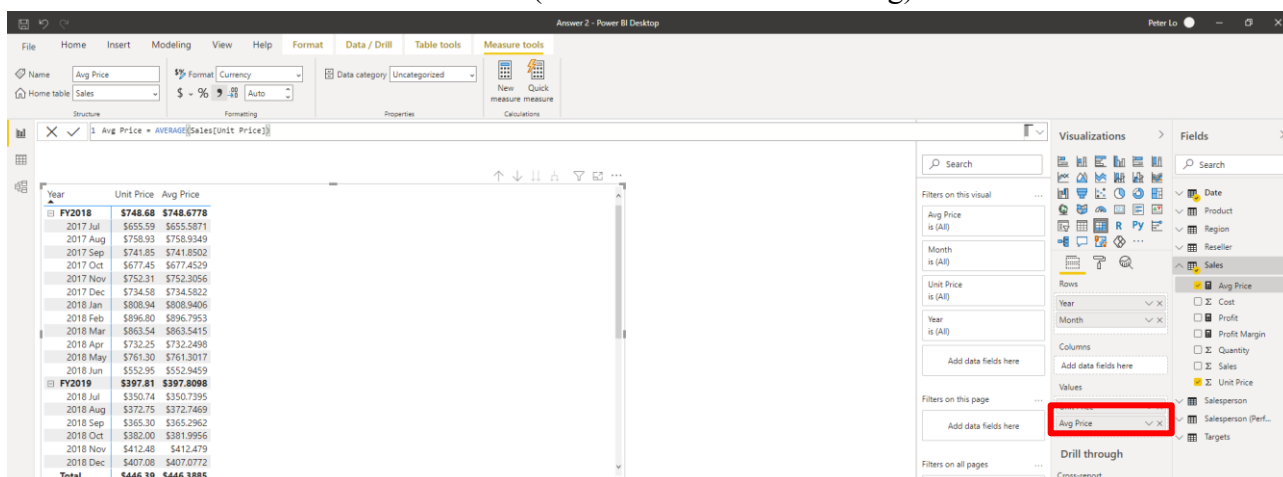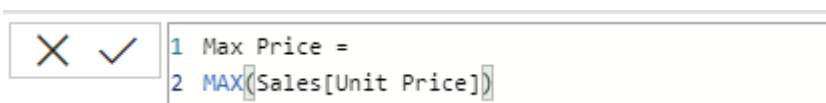
6. To create another measure *Median Price*, right click the *Sales* table and select **New measure**. In the formula bar, type the following, and then press **[Enter]**.
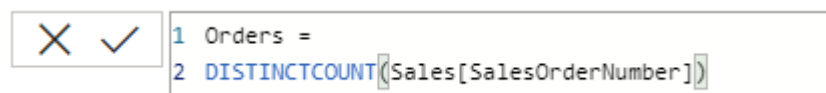
```
1 Median Price =
2 MEDIAN(Sales[Unit Price])
```

7. To create another measure *Min Price*, right click the *Sales* table and select **New measure**. In the formula bar, type the following, and then press **[Enter]**.

```
1 Min Price =
2 MIN(Sales[Unit Price])
```

8. To create another measure *Max Price*, right click the *Sales* table and select **New measure**. In the formula bar, type the following, and then press **[Enter]**.

```
1 Max Price =
2 MAX(Sales[Unit Price])
```

9. To create another measure *Orders*, right click the *Sales* table and select **New measure**. In the formula bar, type the following, and then press **[Enter]**. The **DISTINCTCOUNT** function used in the Orders measure will count orders only once (ignoring duplicates).

```
1 Orders =
2 DISTINCTCOUNT(Sales[SalesOrderNumber])
```

10. To create another measure *Order Lines*, right click the *Sales* table and select **New measure**. In the formula bar, type the following, and then press **[Enter]**. The COUNTROWS function used in the Order Lines measure operates over a table. In this case, the number of orders is calculated by counting the distinct *SalesOrderNumber* column values, while the number of order lines is simply the number of table rows (each row is a line of an order).
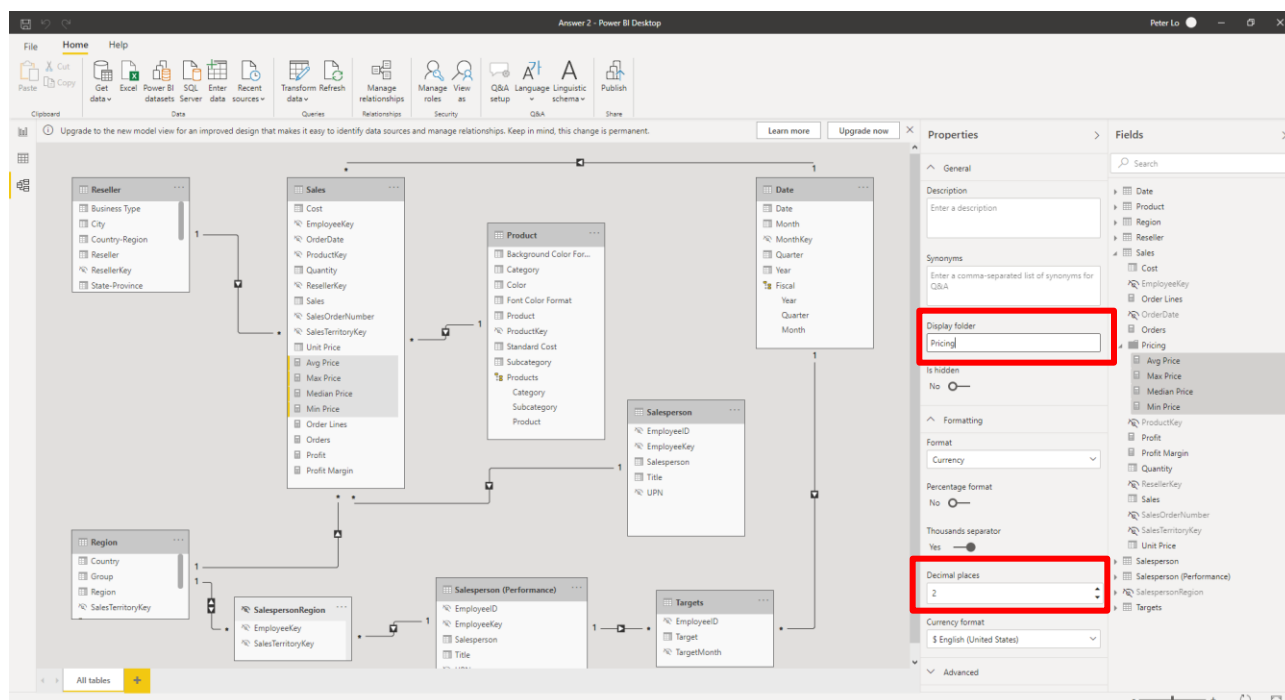
```
1 Order Lines =
2 COUNTROWS(Sales)
```
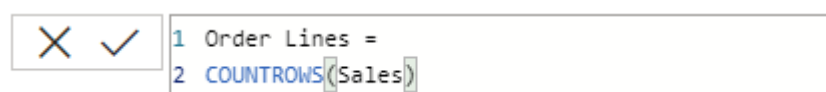
11. The new measure should be created successfully in the Sales table.
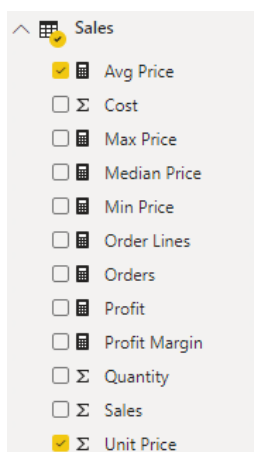
12. Switch to **Model** view, and multi-select the four price measures: *Avg Price, Max Price, Median Price*, and *Min Price*. Then set the **Properties** as follow:

   - Assign to a **Display folder** named **Pricing**
   - Set the **Decimal Places** to **2**

13. Hide the *Unit Price* column. The *Unit Price* column is now not available to report authors. They must use the measure you've added to the model. This design approach ensures that report authors won't inappropriately aggregate prices, for example, by summing them.
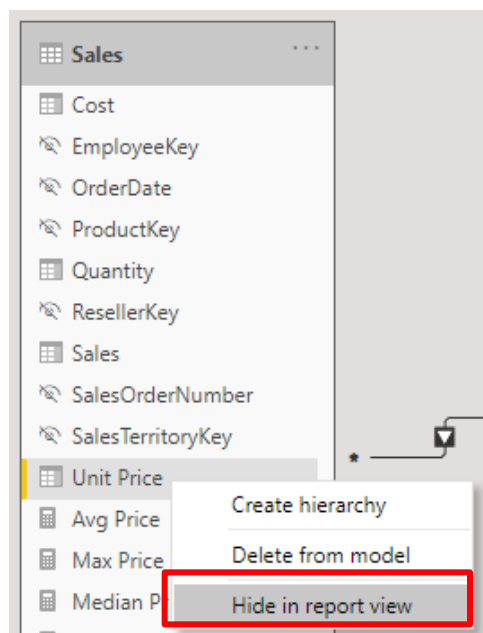
14. Multi-select the *Orders* and *Order Lines* measures, and configure the following requirements:

- Set the **Thousands separator** to **Yes**
- Assign to a **Display folder** named **Counts**
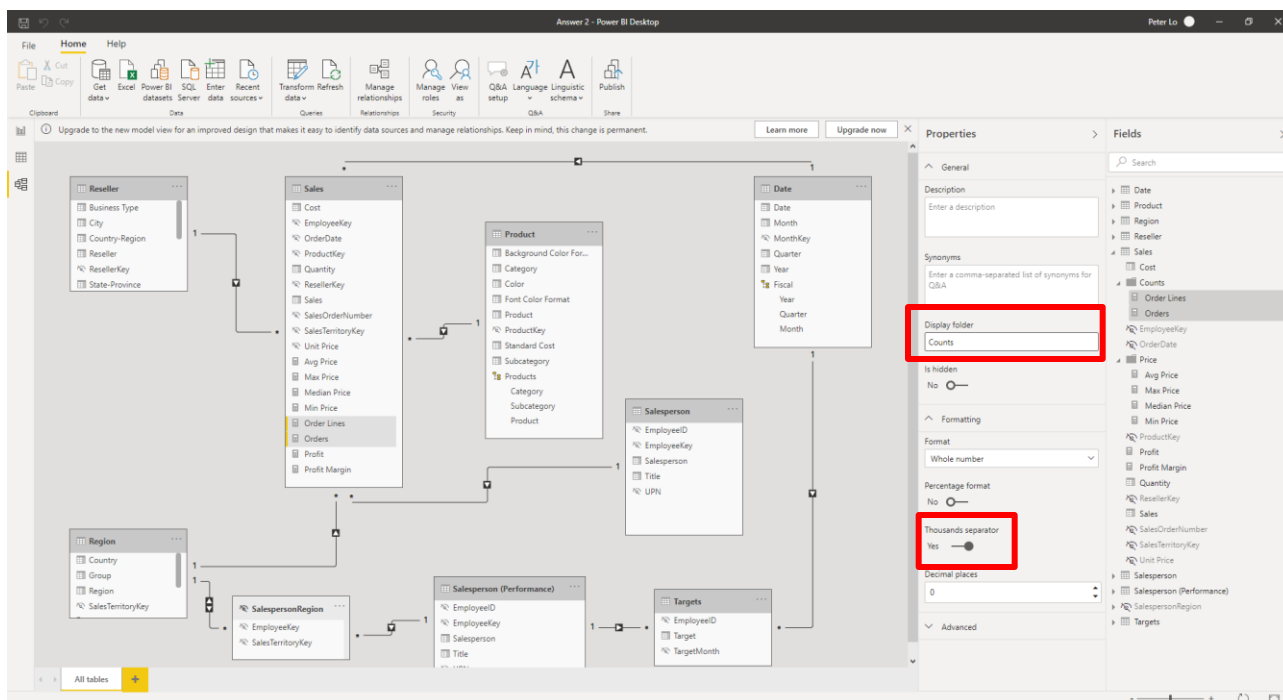
15. In **Report** view, in the **Values** well of the **Matrix** visual, for the *Unit Price* field, click [**X**] to remove it.
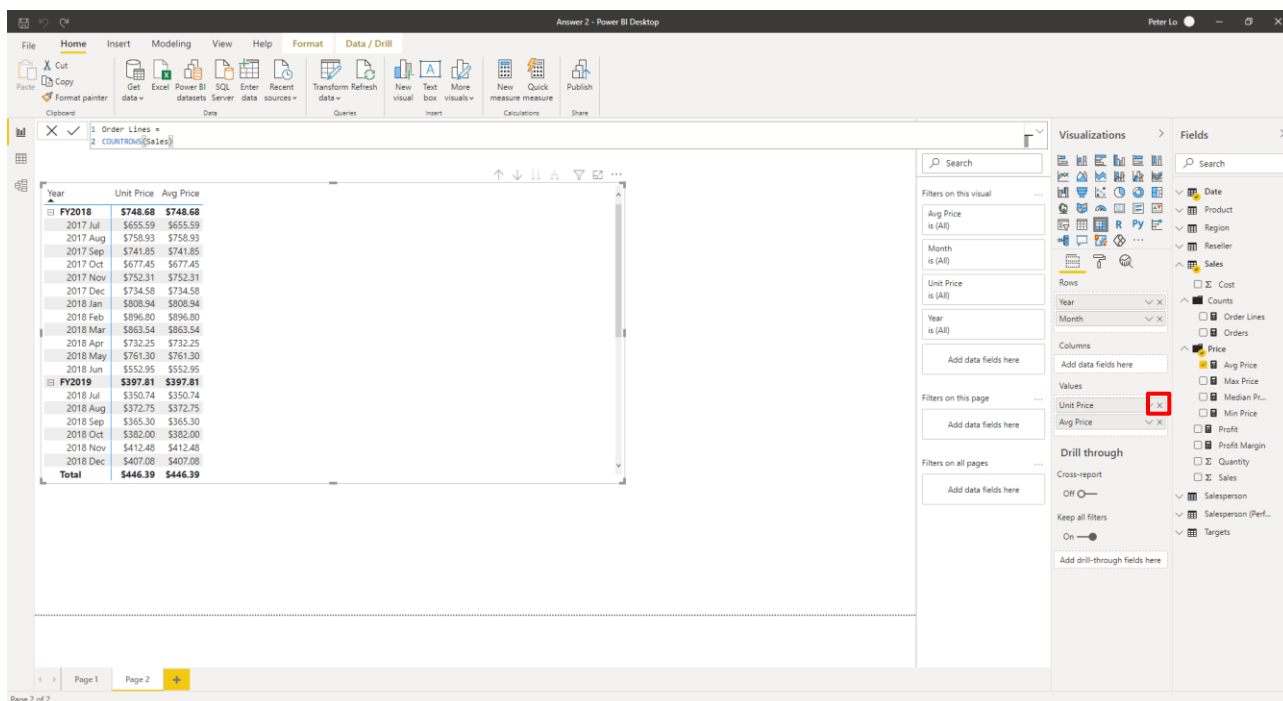
16. Increase the size of the matrix visual to fill the page width and height. Add the following five new measures to the matrix visual:

- Median Price
- Min Price
- Max Price
- Orders
- Order Lines

Verify that the results look sensible and are correctly formatted.
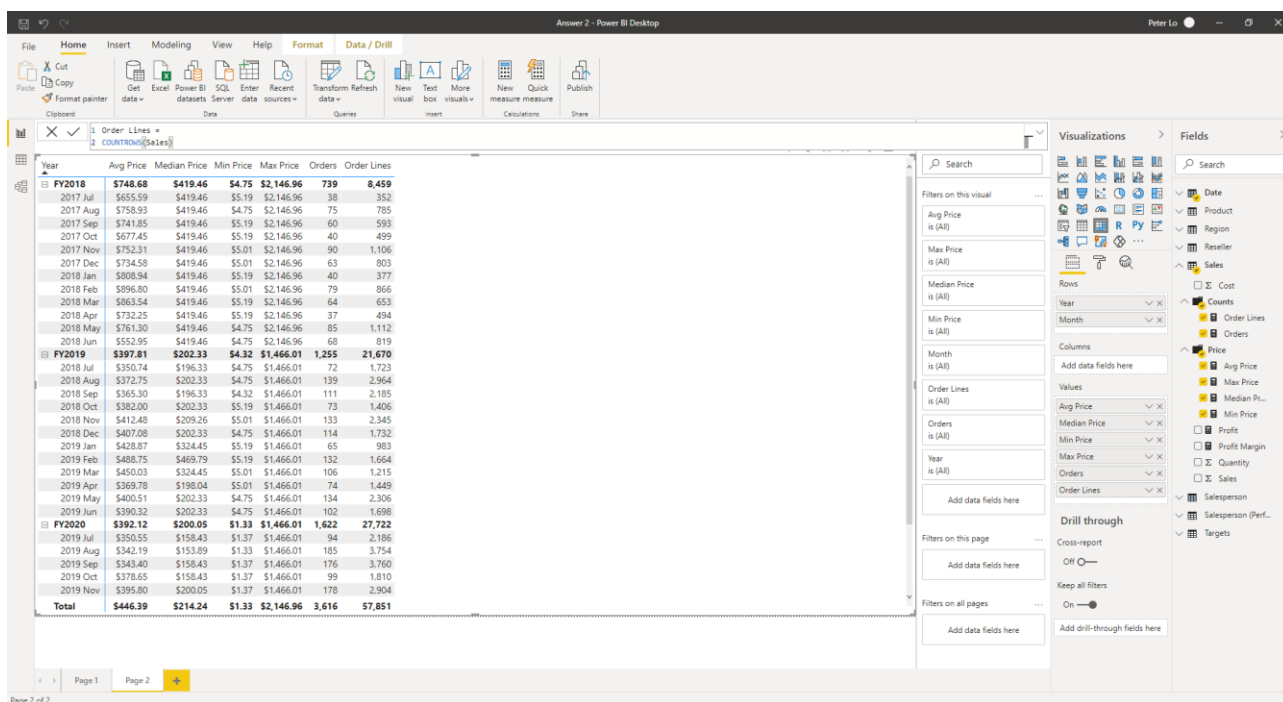
1.2.2   Create Additional Measures

In this task, you will create additional measures that use more complex expressions.

1. In **Report** view, select **Page 1**. Review the table visual, noticing the total for the **Target** column. Summing the target values together doesn't make sense because salespeople targets are set for each salesperson based on their sales region assignment. A target value should only be shown when a single salesperson is filtered. You will implement a measure now to do just that.
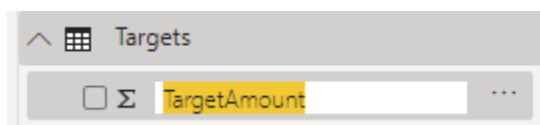
2. In the table visual, remove the **Target** field.
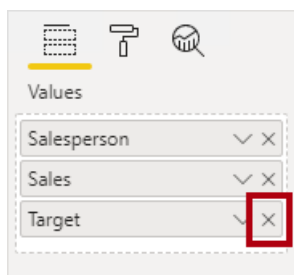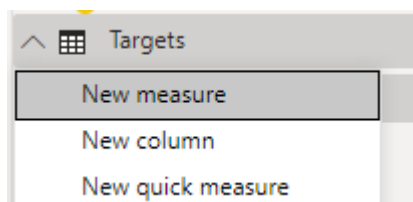

3. Right click the *Target* field in *Targets* table.
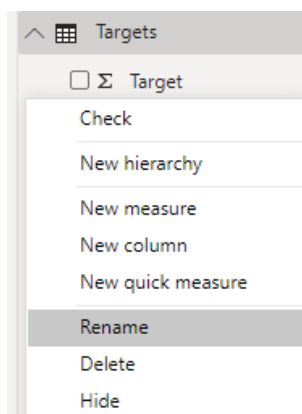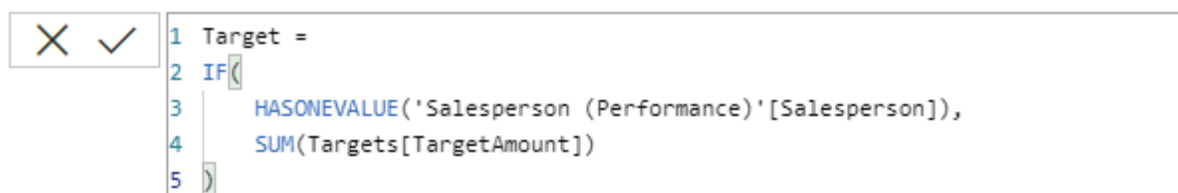

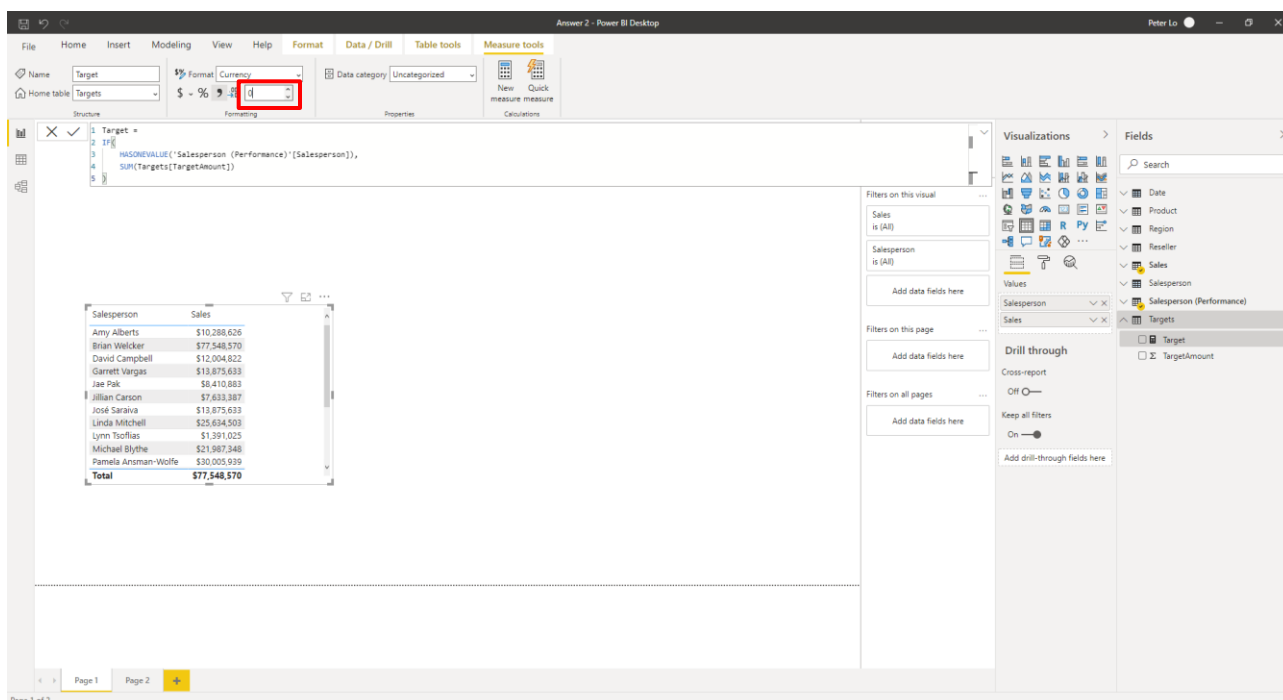4. Rename the *Target* column to *TargetAmount*.

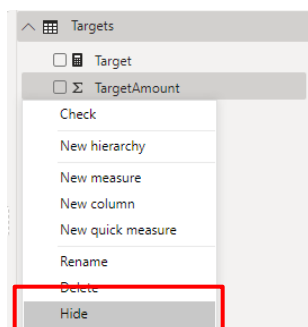5. Right click the *Targets* table, and select **New measure**.

6. In the formula bar, add the following measure definition. The **HASONEVALUE** function tests whether a single value in the Salesperson column is filtered. When true, the expression returns the sum of target amounts (for just that salesperson). When false, BLANK is returned.


    ```
    1  Target =
    2  IF(
    3      HASONEVALUE('Salesperson (Performance)'[Salesperson]),
    4      SUM(Targets[TargetAmount])
    5  )
    ```
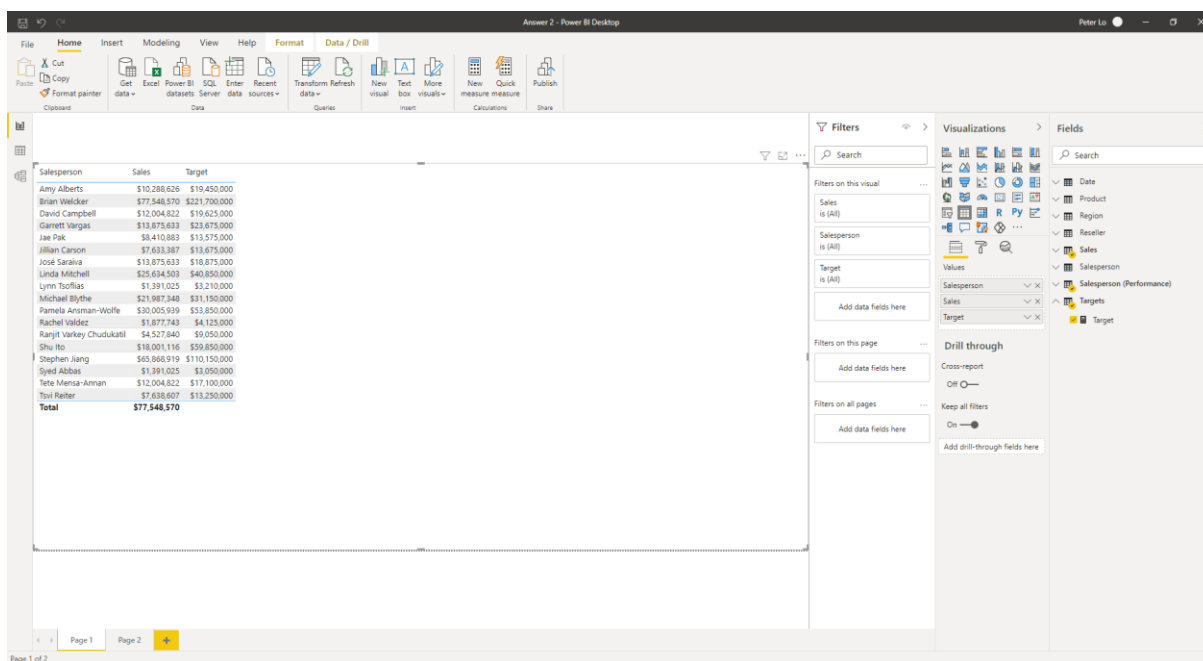
7. Format the *Target* measure for zero decimal places by input **0** in the **Formatting** group of **Measure tools** contextual ribbon.

8. Right click the *TargetAmount* column in *Targets* table, select **Hide**.

9. Add the *Target* measure to the table visual. Notice that the *Target* column total is now blank.

10. To create the *Variance* measure, right click the *Targets* table, and select **New measure**. In the formula bar, add the following measure definition.

```
1 Variance =
2 IF(
3     HASONEVALUE('Salesperson (Performance)'[Salesperson]),
4     SUM(Sales[Sales]) - [Target]
5 )
```

11. To create the *Variance Margin* measure, right click the *Targets* table, and select **New measure**. In the formula bar, add the following measure definition.

```
1 Variance Margin =
2 DIVIDE([Variance], [Target])
```

12. Format the Variance measure for zero decimal places.

13. Format the *Variance Margin* measure as percentage with two decimal places.

14. Add the *Variance* and *Variance Margin* measures to the table visual. Widen the table visual so all values are displayed. While it appears all salespeople are not meeting target, remember that the measures aren't yet filtered by a specific time period.
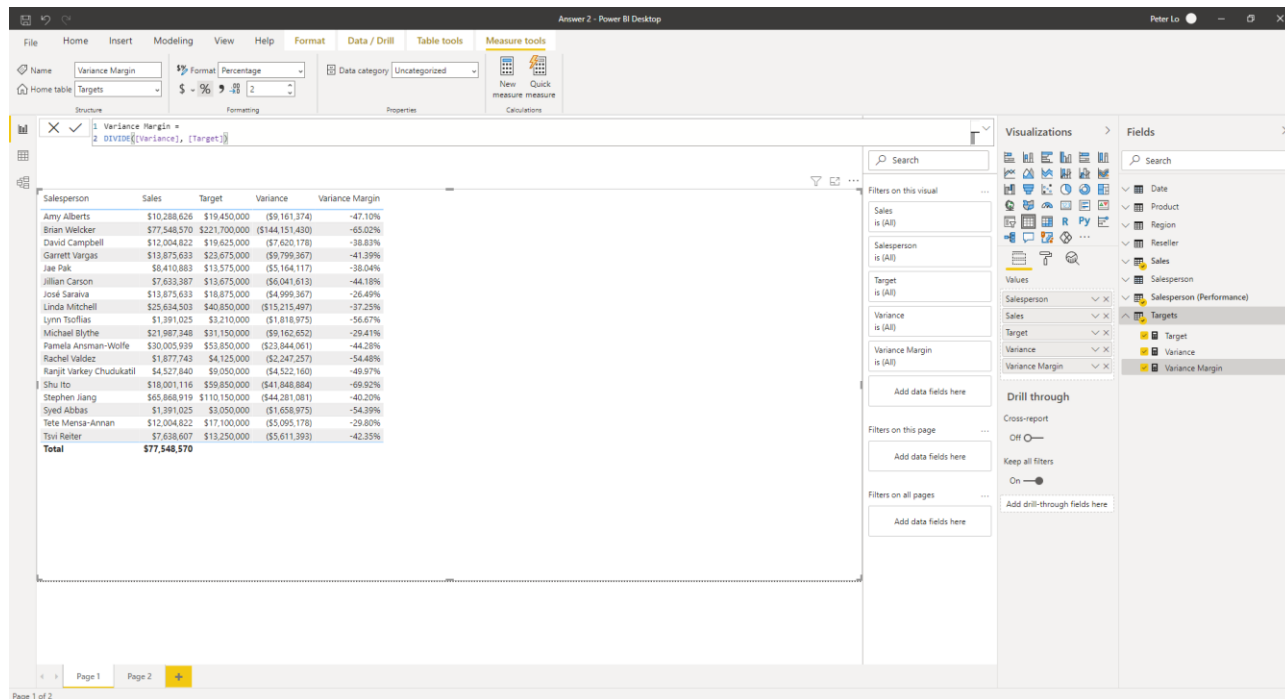
15. At the top-right corner of the **Fields** pane, collapse and then expand open the pane. Collapsing and re-opening the pane resets the content.
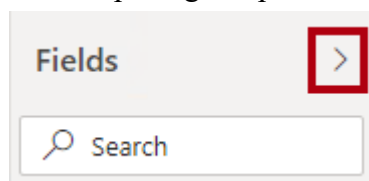
16. Notice that the Targets table now appears at the top of the list. Tables that comprise only visible measures are automatically listed at the top of the list.