


Exercise 3.1: DAX Calculated columns

 We received a new version of the Excel file used for the reporting. In the next steps you will replace the source file and add all available tables within the new source file. The new file adds data to different scenarios as well as taxation.

1. Open the new Excel file “Exercise 3.xlsx” in Excel to familiarize yourself with the data. Did you find any changes compared to the file “Exercise 2.xlsx”?

Hint


There is another sheet named “Scenario” in the workbook as well as a new column in the fact sheet to link the transactional data to the different scenarios. In the next two steps, we will replace the existing data from Exercise 2 with the new one and add our new table “Scenario”.


2. Replace the source of “Exercise 2.xlsx” with “Exercise 3.xlsx” (same steps as in Exercise 2) and click on “Apply changes”
3. Import the Sheet “Scenario” from “Exercise 3.xlsx”


Hint

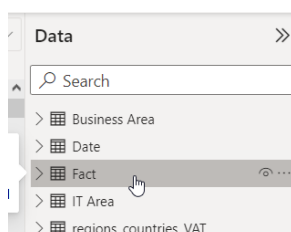
Even though we already imported other sheets from “Exercise 3.xlsx”, we will need to follow the same initial steps to load a new sheet. This means, adding more tables/sheets to an Excel doesn’t automatically make the data model bigger. Only when data is added/changed in existing sheets will the data be updated. The steps are thus the same as in Exercise 1:

Click on “get data” and select “Excel Workbook”. If required, confirm with “Connect” button. In Explorer that opens, navigate to folder where training data is locally stored and select “Exercise 1.xlsx”. In Navigator that opens, activate selection for “Scenario” and click on “Transform Data”.

4.  In model view: connect ‘Scenario’[Scenario ID] to ‘Fact’[Scenario ID]:
‘Scenario’[Scenario ID] ↔ ‘Fact’[Scenario ID]:
5. Import the data from the sheet “regions_countries_VAT” from the Excel file “Exercise 3 regions_countries_VAT.xlsx”.

 Now we will use DAX and new visuals to improve our reporting to fit the new data we added to the model.

6.  In Data view, select the ‘Fact’ table so we can add new columns to the ‘Fact’ table using the **calculated columns** feature



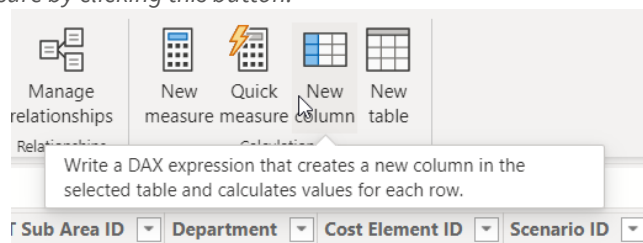
Hint

We select Fact since we want to create new columns in the Fact table. If we have another table selected, the new column is created in the other column.

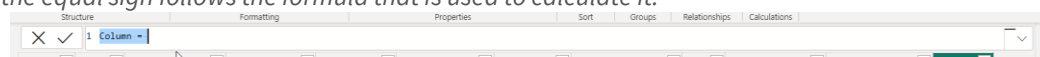
- Let's create a **VAT** column: It should display the "VAT" rate from the table "regions_countries_VAT" next to our transactions in the Fact table.

Hint

You create a measure by clicking this button:



Calculated columns always start with the name of the new column before the equal sign. After the equal sign follows the formula that is used to calculate it.

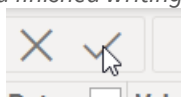


The measure makes use of the `RELATED()` function since it is retrieving a value that is in a related table:

Measure name = `RELATED('Related table name'[Related column name])`

⚠ Please do not copy-paste formulas from this sheet. When you type formulas, the auto-complete function will help you avoid mistakes. Additionally, the hints given may be incomplete or have different names for tables and columns. Always use your own data model as the starting point for your formula. Only write a formula once you are certain you know what you want to achieve with it. Otherwise, feel free to ask for help.

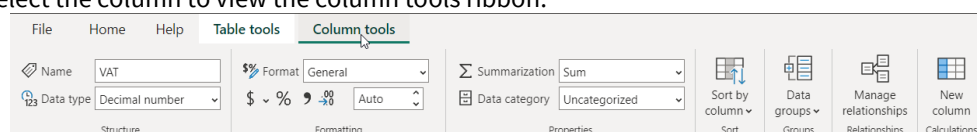
Once you finished writing your formula, you can press the tick icon to save it:



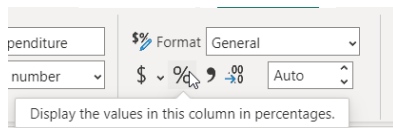
The new column should look like this:

VAT
0.1
0.1
0.1
0.1

- Before moving on to the next column, let's format the VAT column so all values are shown as percentages. Select the column to view the column tools ribbon:



- Click the percentage icon in the Formatting tab. All visuals using the VAT column will now use the percentage formatting when displaying the data.



- Create a **Value_times_VAT** column: It displays the VAT paid on any transaction. It is calculated by multiplying the value without tax by the tax rate.

Solution

```
2 Value_times_VAT = [VAT]*[value]
```

- Create a **Total_expenditure** column: It displays the sum of the previously created columns Value and Value_times_VAT.

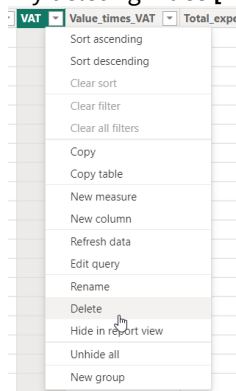
Solution

```
1 Total_expenditure = [value]+(Value_times_VAT)
```

Bonus Exercise

We now have our value before tax **'Fact'[Value]** and our value after tax **'Fact'[Total_expenditure]**. In the process of getting to this, we had to create two intermediary columns: **'Fact'[VAT]** and **'Fact'[Value_times_VAT]**. Because of this, our table now looks quite cluttered, especially because most analyses will not require them. Our task is to remove the intermediary columns to improve our model for usability and performance.

- Try deleting **'Fact'[VAT]** to declutter the model. What happens?



- All other columns that built on it now show an error:

Value_times_VAT	Total_expenditure
#ERROR	#ERROR
#ERROR	#ERROR
#ERROR	#ERROR
#ERROR	#ERROR
#ERROR	#ERROR

3. Add '**Fact**'[VAT] back and see if the error disappears:

```
VAT = RELATED(regions_countries_VAT[VAT])
```

So since that didn't work, how can we remove our intermediary columns? The solution is to use **variables** within one formula. Variables are declared with **var**. To demark the final formula that is returned we use **return**.

4. Try writing a function using variables by following this example. Make sure to adjust the formula in case your columns have different names.


```
Total_expenditure_var =
var VAT = RELATED(regions_countries_VAT[VAT])
var Value_times_VAT = [VAT]*[Value]
return [Value]+[Value_times_VAT]
```

5. Do the values match with the previous Total_expenditure column?

Total_expenditure	VAT	Total_expenditure_var
11208.263	0.1	11208.263
1369.5	0.1	1369.5
1369.5	0.1	1369.5
1369.5	0.1	1369.5
11208.274	0.1	11208.274


Exercise 3.2: DAX measures


Thanks to the Scenario table added in Exercise 3.1, we can now distinguish between our plan values and our actual expenses.

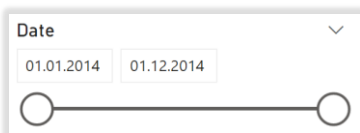
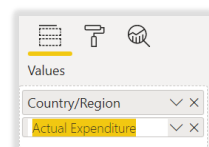
1.  In data view: add following **measures** to the fact table:
 - a. Actual = CALCULATE(SUM([Total_expenditure]), Scenario[ScenarioDescription]= "Actual")

Hint

You will want to sum up all entries in calculated column "Total_expenditures" in Fact table where ScenarioID equals 1, which corresponds to scenario description = "Actual" in Scenario table.

It is a good idea to check your work on measures with a table in the  report view, since the results are not immediately visible. For this measure, try creating a table that shows Date, Total_expenditures and Actual as columns.

- b. Plan: Sum over total_expenditure with filter on Scenario[ScenarioDescription]= "Plan" (analogous to measure "Actual")
 - c. perc_deviation_plan_actual: values from column "Actual" minus values from column "Plan" divided by "Plan"
2.  In report view: add a new sheet and call it "Exercise 3". Add following visualizations:
 - a. Visualize the percentage difference between planned and actual expenditures(perc_deviation_plan_actual) per country in a clustered column chart. Give the chart a meaningful title.
 - b. There are many countries with a deviation of -1. That means there were expenditures in 2014 even though no expenditures at all had been planned. To account for these cases, do the following steps.
 - c. Add a filter on the clustered column chart on perc_deviation_plan_actual so it does not show values of -1
 - d. Add a table with countries and their actual expenditures. Add a filter for those countries that do not have planned expenditures. Rename the column "Actual Expenditure" (Double click on "Actual" in Values section of visual)
 - e. Add a title for the table: "Countries with no planned expenditures"
 - f. Add a slicer with date values from Fact table:



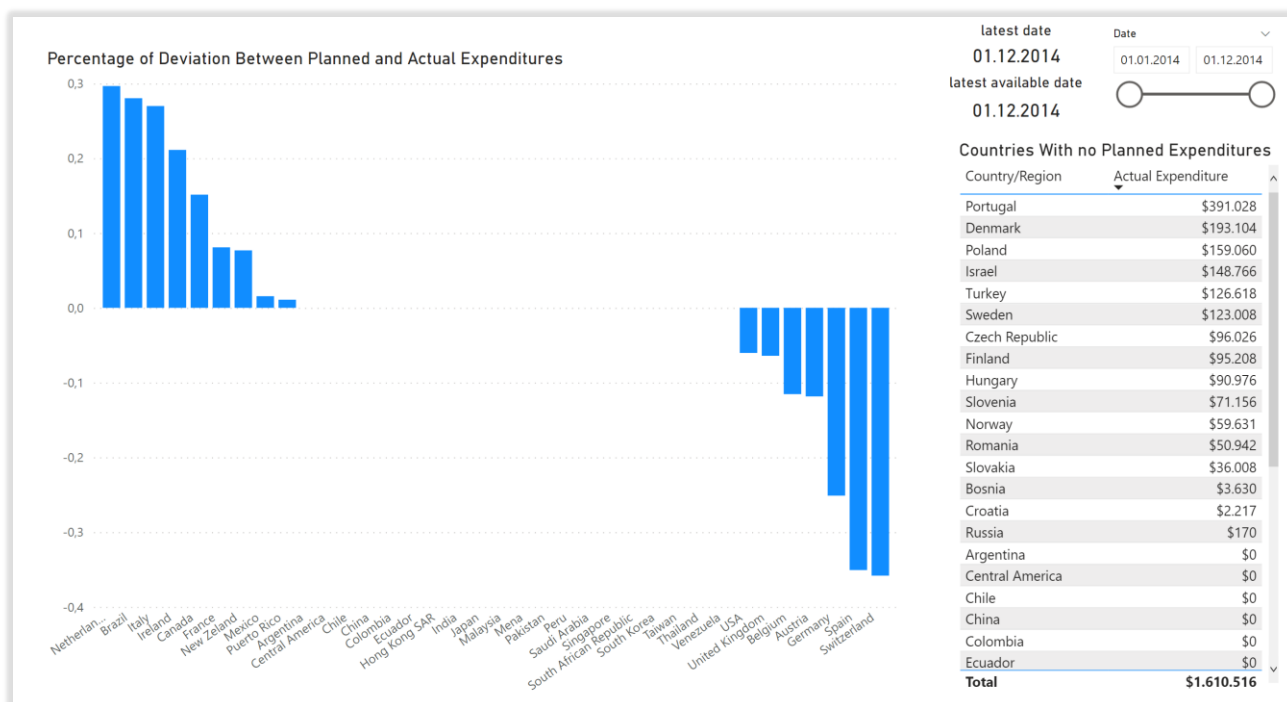
- g. Play with the slicer. Why do the entries in the table "Countries with no planned expenditures" vanish, if we select a period that ends before August 2014?
 - h. Click on measure "Actual" in Fact table. Add "+0" at the end of the DAX formula.
 - i. Play again with the slicer. Do the countries still vanish if we select a period before August 2014? Why?

Bonus Exercise

In Fact table, create a measure for the maximum available date:

max_date = max('Fact'[Date])

1. Add a card visual that displays the latest date. Format the visual, so it looks like in the screenshot on the next page.
2. Play with the date slicer. Why does the latest available date change?
3. From "Exercise 3.xlsx", import sheet "Date".
4. In date table: add a measure with the maximum available date:
universal_max_date = max('Date'[Date])
5. Add another card visual containing universal_max_date.
6. Try formatting the report, so it looks like in the below screenshot.



(The values in the left visual may differ from the screenshot)

Reminder of important DAX-Syntax

- 'Table'[Column]
- "Text"
- Mathematical operators: +-* /
- Concatenation of strings: &
- Reference to a column from the same table via square brackets []
- Reference to a column from another table via "RELATED()"