DAX in Power BI

- Introduction to DAX Language
- When to use M vs. DAX
- Examples

DAX in Power BI

Introduction to DAX Language

DAX: Data Analysis eXpressionLanguage

Unlike M, DAX is an expression language and mostly uses formulas similar to **Excel**.

DAX in Power BI

When to use M vs. DAX

- You will use M mostly for data transformation –joining/union tables, adding columns, pivot/unpivot
- You will use DAX mostly for writing analytical queries –like sum, average, running total etc.
- With M you can fetch data from source and add it to PBI data model. DAX works with data that has been already added to PBI data model.
- Because DAX and M have been built independently (M is part of power query suite and DAX is part of SSAS/power pivot suite), identical things can be achieved using both –join/union, custom column etc. My approach is to use DAX for items which cannot be easily achieved using M –example average, sum, calculating %.

There is a DAX studio which can be used to write and test DAX expressions before using in PBI. Download it here.

DAX in Power BI

Examples

External data | Resources | Insert | Custom visuals | Relationships | Calculations | Share

```
Timesheet_Perc = ADDCOLUMNS( CALCULATETABLE(SUMMARIZE(TimeSheet , TimeSheet[RESOURCE_KEY], TimeSheet[RESOURCE_NAME], "Perc", (IF(ISBLANK(COUNT(TimeSheet[TIMESHEET_KEY])),0,COUNT(TimeSheet[TIMESHEET_KEY]))/4)
*100   ), FILTER(TimeSheet, OR(TimeSheet[TIMESHEET_STATUS_KEY]=4, TimeSheet[TIMESHEET_STATUS_KEY]=1)), FILTER(TimeSheet, TimeSheet[PERIOD_FINISH_DATE]>= TimeSheet[CLARITY_UPDATED_DATE]), filter(TimeSheet,
AND(TimeSheet[PERIOD_START_DATE] <= TODAY(), TimeSheet[PERIOD_START_DATE] >= TODAY()-28))), "Band1", 33, "Band2", 66, "End Value", 100)
```

```
NumberOfIssues = countrows(RELATEDTABLE(Issue))+0
```

DAX in Power BI

Example: Calculate % of timesheets submitted on time for last 4 weeks

TIMESHEET data is added to model.
Created another table using DAX which contains the percentage data:

```
ADDCOLUMNS (
CALCULATETABLE (
SUMMARIZE (
TimeSheet,
TimeSheet[RESOURCE_KEY],
TimeSheet[RESOURCE_NAME],
"Perc", ( COUNT ( TimeSheet[TIMESHEET_KEY] ) / 4 )
* 100
),
FILTER (
TimeSheet,
        OR ( TimeSheet[TIMESHEET_STATUS_KEY] = 4, TimeSheet[TIMESHEET_STATUS_KEY] = 1 )
),
FILTER (
TimeSheet,
TimeSheet[PERIOD_FINISH_DATE] >= TimeSheet[CLARITY_UPDATED_DATE]
),
FILTER (
TimeSheet,
AND (
TimeSheet[PERIOD_START_DATE] <= TODAY (),
TimeSheet[PERIOD_START_DATE]
>= TODAY () -28
)
)
),
"Band1", 33,
"Band2", 66,
"End Value", 100
)
```

You will mostly use **SUMMARIZE**, **CALCULATETABLE**and **FILTER**to pull information from datamodel

DAX in Power BI

Example: Convert SQL to DAX

EVALUATEis not used in PBI. Instead you use =
Filters in **CALCULATETABLE**are always in logical AND
Filters in FILTER can be put in AND/OR.

SQL

```
select *
From dwh_tme_sheet
```

is equivalent to:

DAX

```
EVALUATE
TimeShee
t
```

```
select *
from dwh_tme_sheet
where RESOURCE_KEY = 5003126
and is_adjustment= 0
```

can be written as:

```
EVALUATE
CALCULATETABLE (
TimeSheet,
TimeSheet[RESOURCE_KEY] = 5003126,
TimeSheet[IS_ADJUSTMENT] = 0
)
OR
EVALUATE
FILTER (
TimeSheet,
   AND ( TimeSheet[RESOURCE_KEY] = 5003126, TimeSheet[IS_ADJUSTMENT] = 0 )
)
```

```
Select                    ADJUSTMENT_KEY,
APPROVED_BY From dwh_tme_sheet
```

is equivalent to:

```
EVALUATE
SUMMARIZECOLUMNS ( TimeSheet[ADJUSTMENT_KEY], TimeSheet[APPROVED_BY],
TimeSheet )
```

You can combine CALCULATETABLE and FILTER:

```
EVALUATE
CALCULATETABLE (
TimeSheet,
FILTER ( TimeSheet, TimeSheet[RESOURCE_KEY] = 5003126 ),
FILTER ( TimeSheet, TimeSheet[IS_ADJUSTMENT] = 0 )
)
```

DAX in Power BI

Example: Summarize/Group By

SQL

```
select resource_key, resource_name,
count(timesheet_key) from dwh_tme_sheet
group by resource_key, resource_name
```

is written as:

DAX

```
EVALUATE
SUMMARIZE (
TimeSheet,
TimeSheet[RESOURCE_KEY],
TimeSheet[RESOURCE_NAME],
   "Count", COUNT ( TimeSheet[TIMESHEET_KEY] )
)
```

Now, add a few filter conditions:

```
select resource_key, resource_name, count(timesheet_key)
from dwh_tme_sheet
where TIMESHEET_STATUS_KEY in (4,1)
and PERIOD_FINISH_DATE >= CLARITY_UPDATED_DATE
and PERIOD_START_DATE between (sysdate-28) and sysdate
group by resource_key, resource_name
```

is written as:

```
EVALUATE
CALCULATETABLE (
SUMMARIZE (
TimeSheet,
TimeSheet[RESOURCE_KEY],
TimeSheet[RESOURCE_NAME],
"Count", COUNT ( TimeSheet[TIMESHEET_KEY] )
),
FILTER (
TimeSheet,
    OR ( TimeSheet[TIMESHEET_STATUS_KEY] = 4, TimeSheet[TIMESHEET_STATUS_KEY] = 1 )
),
FILTER (
TimeSheet,
TimeSheet[PERIOD_FINISH_DATE] >= TimeSheet[CLARITY_UPDATED_DATE]
),
FILTER (
TimeSheet,
AND (
TimeSheet[PERIOD_START_DATE]
>= TODAY () -28,
TimeSheet[PERIOD_START_DATE] <= TODAY ()
)
)

)
```

DAX in Power BI

Example: Use ADDCOLUMNS to add new columns

```
EVALUATE
ADDCOLUMNS (
CALCULATETABLE (
SUMMARIZE (
TimeSheet,
TimeSheet[RESOURCE_KEY],
TimeSheet[RESOURCE_NAME],
"Count", COUNT ( TimeSheet[TIMESHEET_KEY] )
),
FILTER (
TimeSheet,
        OR ( TimeSheet[TIMESHEET_STATUS_KEY] = 4, TimeSheet[TIMESHEET_STATUS_KEY] = 1 )
),
FILTER (
TimeSheet,
TimeSheet[PERIOD_FINISH_DATE] >= TimeSheet[CLARITY_UPDATED_DATE]
),
FILTER (
TimeSheet,
AND (
TimeSheet[PERIOD_START_DATE]
>= TODAY () -28,
TimeSheet[PERIOD_START_DATE] <= TODAY ()
)
)
),
"COL1", 100,
"COL2", 200
)
```

Number of Tasks for Each Projects:

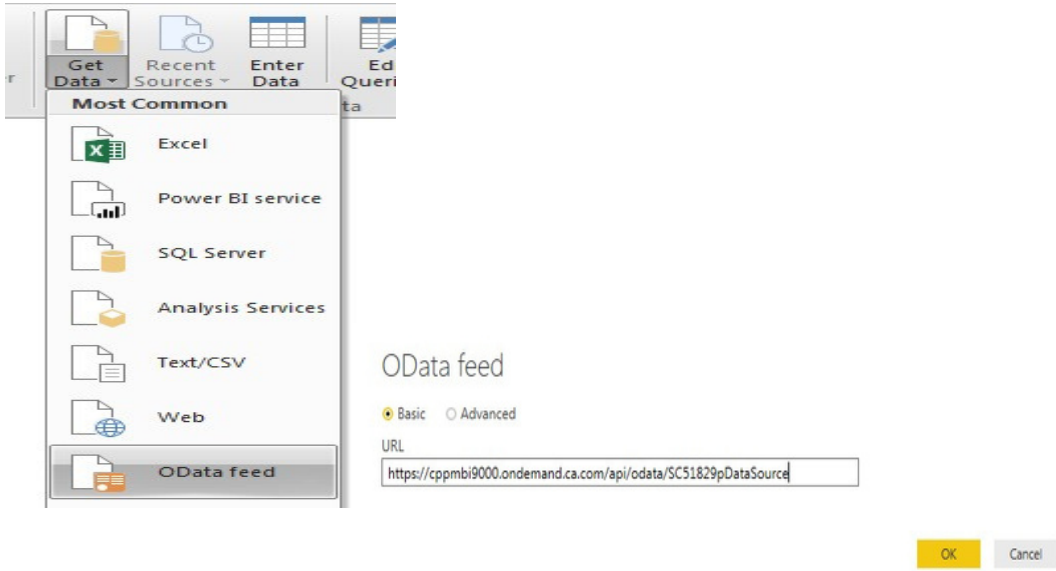
```
EVALUATE
ADDCOLUMNS ( Project, "Number of Task", COUNTROWS ( RELATEDTABLE ( Task ) ) )
```

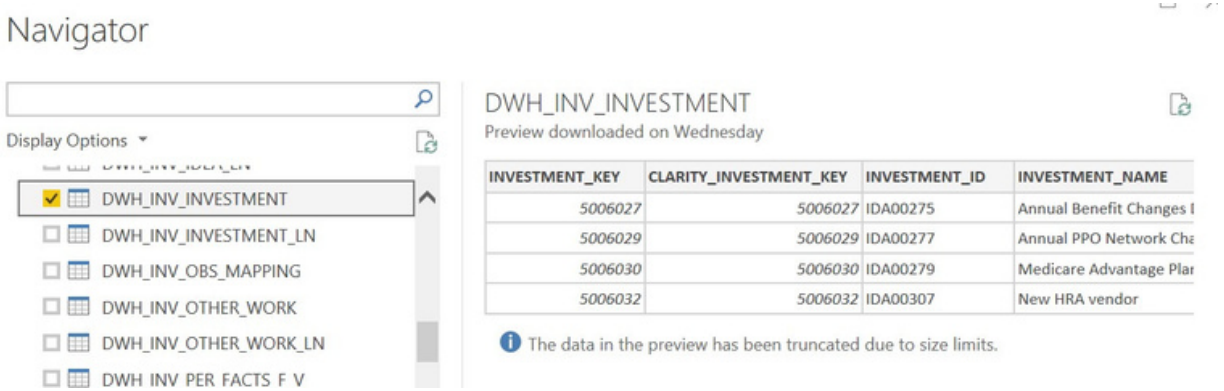
****RELATEDTABLE** uses the data model relationship to find the table joins. Avoid using this and use explicit join conditions.

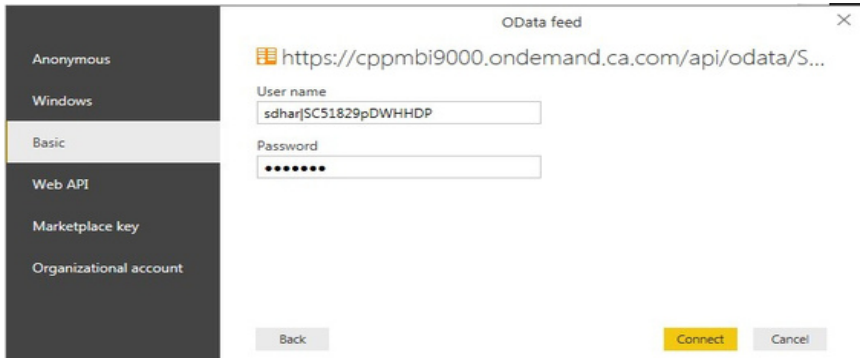
Create a Dashboard by connecting to OdataFeed

Here are the steps to create a dashboard:

- Open your Power BI Desktop instance
- Select ODatafeed option under GetData
- Enter the OdataURL provided by your admin and click on OK.
- Enter the username & password and click on Connect to authenticate.
- OdataTable Navigator opens up. Select the tables required for your dashboard and click on Load



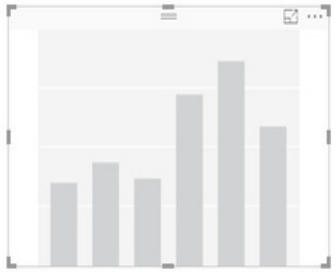


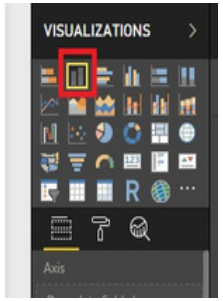


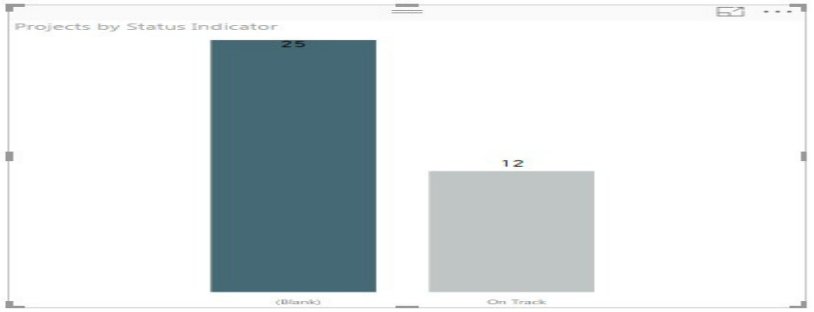
- Selected Tables gets loaded and power BI canvas opens up.

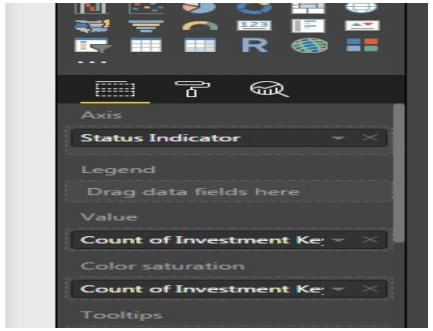
Create a Dashboard by connecting to OdataFeed

- Select the required visualization and drag the fields from the tables on to the settings to generate a meaningful visualization.









- You can group related visualizations under a single dashboard. Save the dashboard after completion.