POWER-BI PROGRAMMING LANGUANGES

Prof. Carlos J. Costa, PhD
Associate Professor of Information Systems and Operation Management

- M Language
- DAX
- SQL
- R
- Python

M Language

- Power Query M Formula Language.
- Language used in the context of PowerQuery
- It's a functional, case sensitive language like F#.

Microsoft Power Query

- Is a data import tool.
- Works with Analysis Services, Excel, and Power BI workbooks.
- A core capability of Power Query is to filter and combine, that is, to mash-up data from one or more of a rich collection of supported data sources.
- All those operations may be used through Power Query M Formula Language. It's a functional, case sensitive language like F#.

M Language

- a **function** is a mapping from a set of input values to a single output value
- A function is written by first naming the function parameters, and then providing an expression to compute the result of the function.

M Language

- The body of the function follows the goes-to (=>) symbol.
- Optionally, type information can be included on parameters and the function return value.
- A function is defined and invoked in the body of a **let** statement.
- Parameters and/or return value can be implicit or explicit.
- Implicit parameters and/or return value are of type **any**.
- Type **any** is similar to an object type in other languages.
- All types in M derive from type **any**.

- A **function** is a value just like a number or a text value.
- A function can be included in-line just like any other expression.
- The following example shows a function which is the value of an Add variable which is then invoked, or executed, from several other variables.
- When a function is invoked, a set of values are specified which are logically substituted for the required set of input values within the function body expression.

M Language

- To create an advanced query, you use the Query Editor. A mashup query is composed of variables, expressions, and values encapsulated by a let expression. A variable can contain spaces by using the # identifier with the name in quotes as in #"Variable name".

- A let expression follows this structure:
- powerquery-m

- let
- Variablename = expression,
- #"Variable name" = expression2
- in
- Variablename

- https://docs.microsoft.com/en-us/powerquery-m/

DAX

- **D**ata **A**nalysis E**x**pressions

DAX

- DAX is a collection of functions that can be used to perform a task and return one or more values.
- DAX is primarily a formula or a query language.

DAX

- was developed in 2009 by Microsoft to be used with Microsoft's PowerPivot, which at that time was available as an Excel (2010) add-in.
- It is now the language of choice for Power BI

New Column

- New Column for SalesValue:

SalesValue = Order[SalesPrice]*Order[quantity]

New Column

- New Column Unit Cost:

- UnitCosts = LOOKUPVALUE('Product'[unitCost];'
  Product'[idProduct];
  Sales[idProduct])

Create New Table

- Create a new table for Date

```
Date = CALENDAR (DATE(2018;1;1); DATE(2020;12;31))
```

- or:

```
Date =
    ADDCOLUMNS (
    CALENDAR (DATE(2000;1;1); DATE(2025;12;31));
    "Year"; YEAR ( [Date] )
     )
```

Create New Table

- Create new table called BackMagicStore Including all orders corresponding to Black Magic (p3).

BlackMagicStore=FILTER(Orders,Orders[ProductID] = "p3")

Create Column for Customer type

- create a column with total

```
Total =SUMX(FILTER(Orders;
        Orders[CustomerID]=Customer[CustomerID]);
[Quantity]*[SalesPrice])
```

- Create a column with total:

```
CustomerType = VAR TotalSales=Customer[Total]
        RETURN
        IF(TotalSales>100000;"Gold";"Silver")
```