



# Keep calm and C#

Scripting in Power BI



b.telligent

# Paulina Jędrzejewska

Born in Poland

Studied business mathematics in Germany

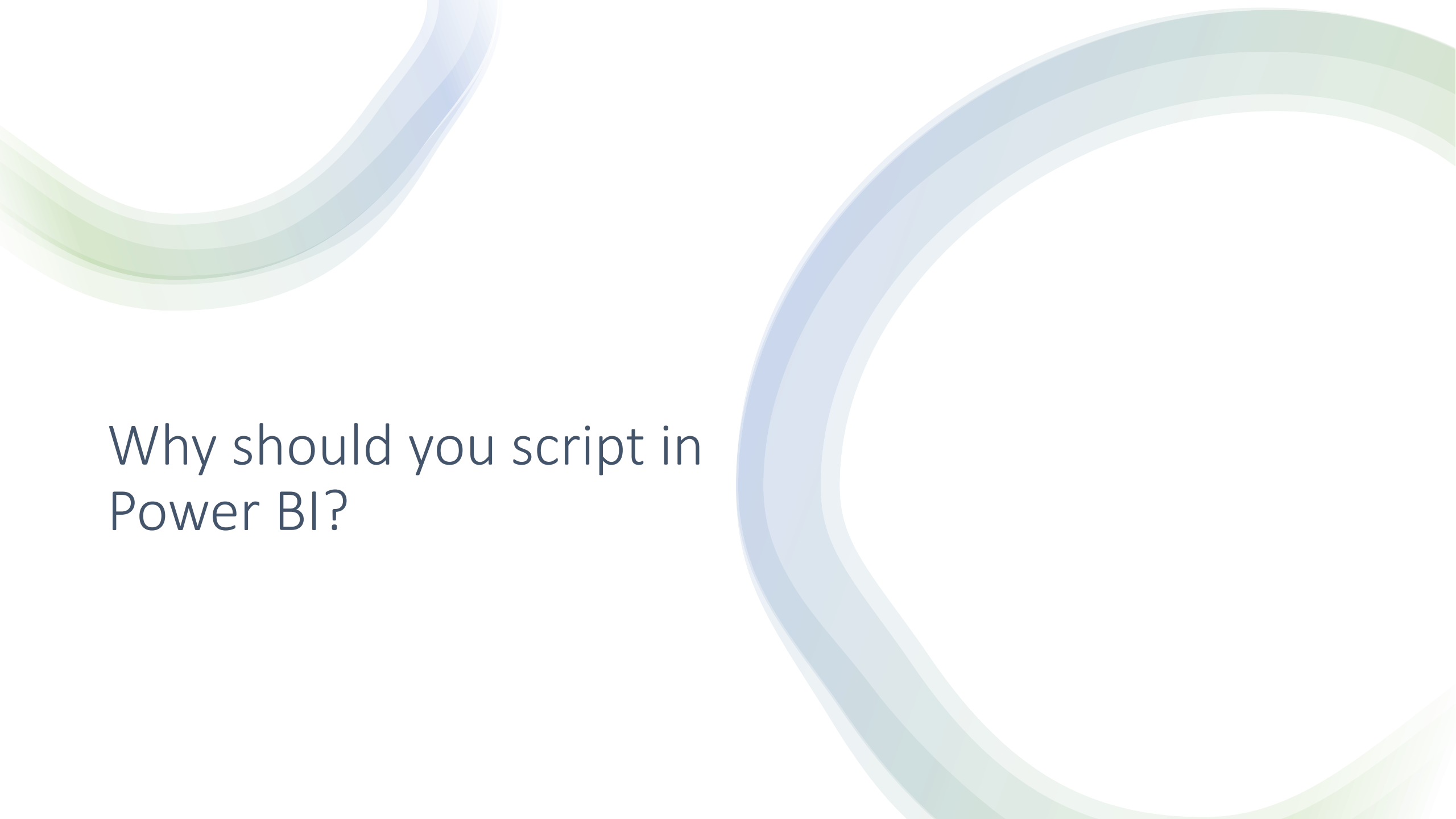
Living in Switzerland

Working with Power BI since 2018

Microsoft Certified Trainer

# Agenda

1. Why should you script in Power BI?
2. C# Basics
3. Objects in Tabular Model
4. Deep Dive into Code



Why should you script in  
Power BI?

# Tasks you can automate with C#

- Creating and deleting measures
- Renaming objects
- Moving objects into folders
- Formatting objects

# Advantages of scripting

- Works through a bunch of objects in seconds
- Reusable
- Homogeneous structure and naming
- No more tedious tasks for you!

# What do you need to start scripting?

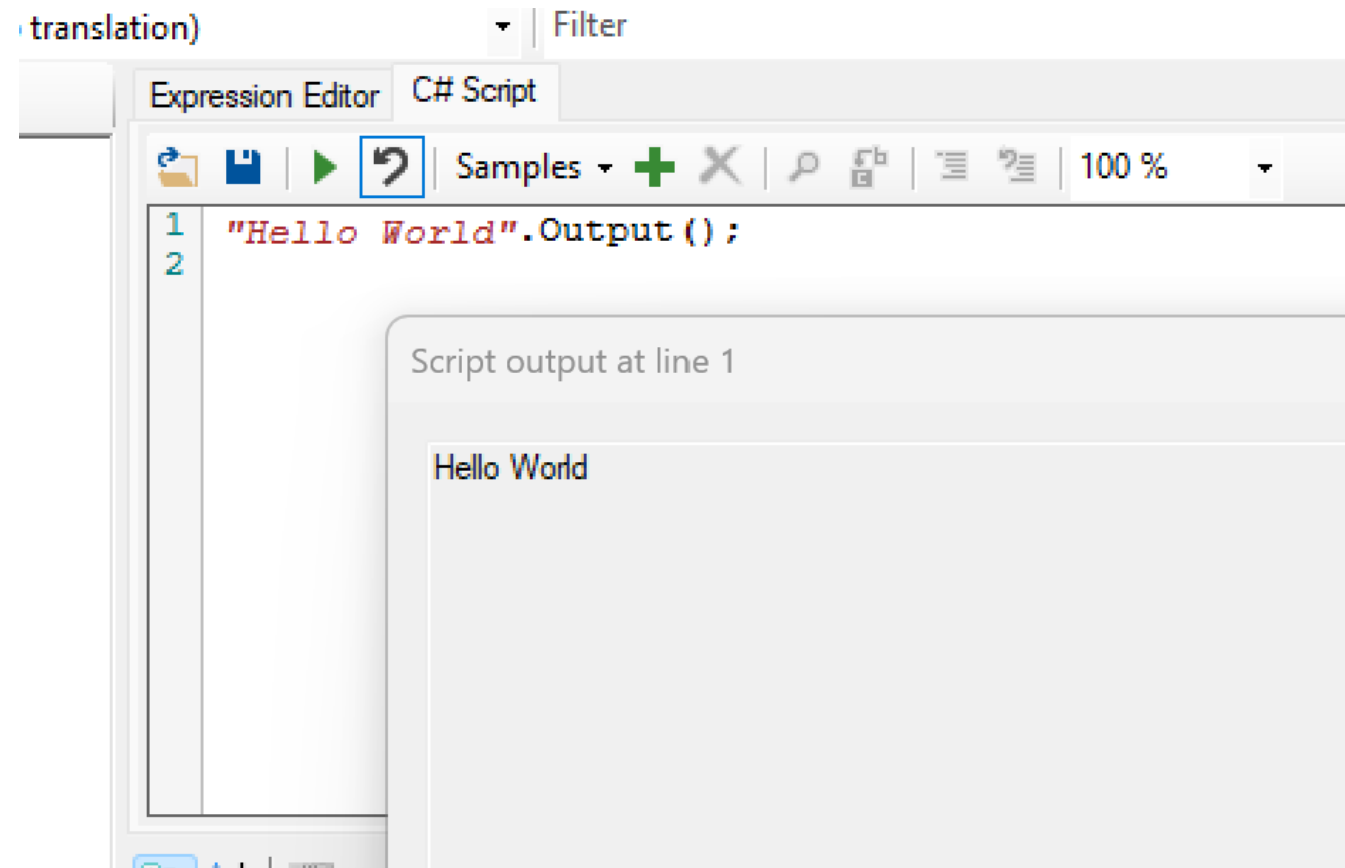
- Tabular Editor 2
- Power BI Desktop
- Code Editor



# C# Basics



*"Hello World".Output();*



```
// This is a comment
```

```
/* This is a  
   multi row Comment */
```

```
type variableName = value;  
var variableName = value;
```

Int - integers (whole numbers), 123 or -123

double - floating point numbers, 19.99 or -19.99

char - single characters, 'a' or 'B'

string - text, "Hello World"

bool - true or false

List:

```
var Name = new List<type>();
```

Dictionary:

```
var Name = new Dictionary<key type, value type>()  
{  
    {key1, value1},  
    {key2, value2},  
    {key3, value3}  
}
```

`"String".Contains("Substring");`

`"String".EndsWith ("Substring");`

`"String". Substring(Start, End);`

`string.Format("String with placeholder {0}", Text);`

`.ToList() or .ToArray();`

`List.Where(variable => condition);`

```
List<int> numbers = new List<int>()
```

```
{
```

```
    1, 2, 4, 8, 16, 32
```

```
};
```

```
var smallNumbers = numbers.Where(n => n < 10);
```

```
foreach (type variableName in arrayName)
{
    // code block to be executed
}
```

```
if (condition1)
```

```
{
```

```
    // block of code to be executed if condition1 is True
```

```
}
```

```
else if (condition2)
```

```
{
```

```
    // block of code to be executed if the condition1 is False and condition2 is True
```

```
}
```

```
else
```

```
{
```

```
    // block of code to be executed if the condition1 is False and condition2 is False
```

```
}
```



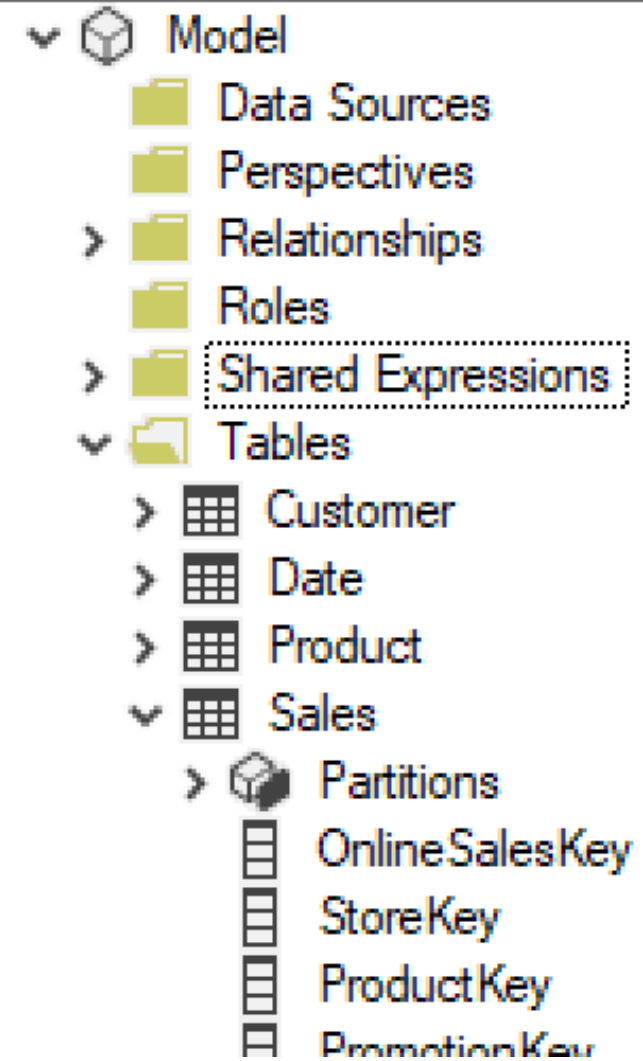


# Objects in Tabular Model

Model

Model.Tables

Model.Tables["table name"]



Model.Tables["table name"].Column

Model.Tables["table name"]  
                    .Column ["column name"]

Model.Tables["table name"].Measures

Model.Tables["table name"]  
    .Measures ["measure name"]

Model.Tables["table name"]

.Measures ["measure name"]

.Name

.DaxObjectFullName

.Expression

.DisplayFolder

.FormatString

▼	<b>Basic</b>
	Description
	Display Folder
>	Format String
	Hidden
	Name
▼	<b>Metadata</b>
	Data Type
	Error Message
	Object Type
▼	<b>Options</b>
	Data Category
	Detail Rows Expression
	Expression
	Format String Expression

```
Model.Tables["table name"]  
    .AddMeasure(Name,  
                Expression,  
                DisplayFolder)
```



Generate measures from columns

```
1  /* Generate measures from columns */
2
3  var displayTable = Model.Tables["_Measures"];
4  var kpiSourceTable = Model.Tables["Sales"];
5
6  foreach(var column in kpiSourceTable.Columns)
7  {
8      if (column.Name.Contains("KPI "))
9      {
10         var new_measure = displayTable.AddMeasure(
11             column.Name,                                     // Name
12             "SUM(" + column.DaxObjectFullName + ") ",      // DAX expression
13             "Generate from Columns"                          // Display Folder
14         );
15     }
16 }
```



- Model
  - Data Sources
  - Perspectives
  - Relationships
  - Roles
  - Shared Expressions
  - Tables
    - Customer
    - Date
    - Product
    - Sales
      - Partitions
        - OnlineSalesKey
        - StoreKey
        - ProductKey
        - PromotionKey
        - CurrencyKey
        - CustomerKey
        - OrderDateKey
        - DueDateKey
        - DeliveryDateKey
        - Order Date
        - Due Date
        - Delivery Date
        - Order Number
        - Order Line Number
        - KPI Quantity
        - Unit Price
        - Unit Discount
        - Unit Cost
        - Net Price
        - KPI Sales Amount
    - Store
    - Measures
      - Generate from BASE
      - Generate from calculation items
      - Change format
      - Generate measures from rows
        - Sales Amount
        - Value
    - Time Intelligence
    - Transposed Cities
  - Translations

```

1  /* Generate measures from columns */
2
3  var displayTable = Model.Tables["_Measures"];
4  var kpiSourceTable = Model.Tables["Sales"];
5
6  foreach(var column in kpiSourceTable.Columns)
7  {
8      if (column.Name.Contains("KPI "))
9      {
10         var new_measure = displayTable.AddMeasure(
11             column.Name,
12             "SUM(" + column.DaxObjectFullName + ")",
13             "Generate from Columns"
14         );
15     }
16 }
    
```

// Name  
// DAX expression  
// Display Folder

- Model
  - Data Sources
  - Perspectives
  - Relationships
  - Roles
  - Shared Expressions
  - Tables
    - Customer
    - Date
    - Product
    - Sales
      - Partitions
        - OnlineSalesKey
        - StoreKey
        - ProductKey
        - PromotionKey
        - CurrencyKey
        - CustomerKey
        - OrderDateKey
        - DueDateKey
        - DeliveryDateKey
        - Order Date
        - Due Date
        - Delivery Date
        - Order Number
        - Order Line Number
        - KPI Quantity
        - Unit Price
        - Unit Discount
        - Unit Cost
        - Net Price
        - KPI Sales Amount
      - Store
    - \_Measures
      - Generate from BASE
      - Generate from calculation items
      - Change format
      - Generate measures from rows
        - Sales Amount
        - Value
    - Time Intelligence
    - Transposed Cities
  - Translations

```

1  /* Generate measures from columns */
2
3  var displayTable = Model.Tables[" Measures"];
4  var kpiSourceTable = Model.Tables["Sales"];
5
6  foreach(var column in kpiSourceTable.Columns)
7  {
8      if (column.Name.Contains("KPI "))
9      {
10         var new_measure = displayTable.AddMeasure(
11             column.Name,
12             "SUM(" + column.DaxObjectFullName + ")",
13             "Generate from Columns"
14         );
15     }
16 }
    
```

// Name  
// DAX expression  
// Display Folder

File Edit View Model Tools

Perspective: (All objects)

Translation: (No translation)

Filter

Model

- Data Sources
- Perspectives
- Relationships
- Roles
- Shared Expressions
- Tables
  - Customer
  - Date
  - Product
  - Sales
    - Partitions
      - OnlineSalesKey
      - StoreKey
      - ProductKey
      - PromotionKey
      - CurrencyKey
      - CustomerKey
      - OrderDateKey
      - DueDateKey
      - DeliveryDateKey
      - Order Date
      - Due Date
      - Delivery Date
      - Order Number
      - Order Line Number
      - KPI Quantity
      - Unit Price
      - Unit Discount
      - Unit Cost
      - Net Price
      - KPI Sales Amount
- Store
- Measures
  - Generate from BASE
  - Generate from calculation items
  - Change format
  - Generate measures from rows
    - Sales Amount
      - Value
  - Time Intelligence
  - Transposed Cities
- Translations

Expression Editor C# Script

Samples + X

160 %

```

1  /* Generate measures from columns */
2
3  var displayTable = Model.Tables["_Measures"];
4  var kpiSourceTable = Model.Tables["Sales"];
5
6  foreach(var column in kpiSourceTable.Columns)
7  {
8      if (column.Name.Contains("KPI "))
9      {
10         var new_measure = displayTable.AddMeasure(
11             column.Name,
12             "SUM(" + column.DaxObjectFullName + ")",
13             "Generate from Columns"
14         );
15     }
16 }

```

// Name

// DAX expression

// Display Folder

Nothing selected.

0 BP issues

Script executed successfully. 6 model changes.

- Model
  - Data Sources
  - Perspectives
  - Relationships
  - Roles
  - Shared Expressions
  - Tables
    - Customer
    - Date
    - Product
    - Sales
      - Partitions
        - OnlineSalesKey
        - StoreKey
        - ProductKey
        - PromotionKey
        - CurrencyKey
        - CustomerKey
        - OrderDateKey
        - DueDateKey
        - DeliveryDateKey
        - Order Date
        - Due Date
        - Delivery Date
        - Order Number
        - Order Line Number
        - KPI Quantity
        - Unit Price
        - Unit Discount
        - Unit Cost
        - Net Price
        - KPI Sales Amount
    - Store
  - Measures
    - Generate from BASE
    - Generate from calculation items
    - Change format
    - Generate measures from rows
      - Sales Amount
      - Value
    - Time Intelligence
    - Transposed Cities
  - Translations

```

1  /* Generate measures from columns */
2
3  var displayTable = Model.Tables["_Measures"];
4  var kpiSourceTable = Model.Tables["Sales"];
5
6  foreach(var column in kpiSourceTable.Columns)
7  {
8      if (column.Name.Contains("KPI "))
9      {
10         var new_measure = displayTable.AddMeasure(
11             column.Name,
12             "SUM(" + column.DaxObjectFullName + ")",
13             "Generate from Columns"
14         );
15     }
16 }
    
```

// Name  
// DAX expression  
// Display Folder

FileEditViewModelTools

Perspective: (All objects)

Translation: (No translation)

Filter

Σ

📊

🔍

📄

📁

📌

📎

🔗

🔧

Model

- Data Sources
- Perspectives
- Relationships
- Roles
- Shared Expressions
- Tables
  - Customer
  - Date
  - Product
  - Sales
    - Partitions
      - OnlineSalesKey
      - StoreKey
      - ProductKey
      - PromotionKey
      - CurrencyKey
      - CustomerKey
      - OrderDateKey
      - DueDateKey
      - DeliveryDateKey
      - Order Date
      - Due Date
      - Delivery Date
      - Order Number
      - Order Line Number
      - KPI Quantity
      - Unit Price
      - Unit Discount
      - Unit Cost
      - Net Price
      - KPI Sales Amount
    - Store
  - \_Measures
    - Generate from BASE
    - Generate from calculation items
    - Change format
    - Generate measures from rows
      - Sales Amount
        - Value
    - Time Intelligence
    - Transposed Cities
  - Translations

Expression Editor C# Script

Samples + X

160 %

```

1  /* Generate measures from columns */
2
3  var displayTable = Model.Tables["_Measures"];
4  var kpiSourceTable = Model.Tables["Sales"];
5
6  foreach(var column in kpiSourceTable.Columns)
7  {
8      if (column.Name.Contains("KPI "))
9      {
10         var new_measure = displayTable.AddMeasure(
11             column.Name,
12             "SUM(" + column.DaxObjectFullName + ")",
13             "Generate from Columns"
14         );
15     }
16 }

```

// Name

// DAX expression

// Display Folder

Nothing selected.

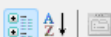
0 BP issues

Script executed successfully. 6 model changes.



- Model
  - Data Sources
  - Perspectives
  - Relationships
  - Roles
  - Shared Expressions
  - Tables
    - Customer
    - Date
    - Product
    - Sales
      - Partitions
        - OnlineSalesKey
        - StoreKey
        - ProductKey
        - PromotionKey
        - CurrencyKey
        - CustomerKey
        - OrderDateKey
        - DueDateKey
        - DeliveryDateKey
        - Order Date
        - Due Date
        - Delivery Date
        - Order Number
        - Order Line Number
        - KPI Quantity
        - Unit Price
        - Unit Discount
        - Unit Cost
        - Net Price
        - KPI Sales Amount
    - Store
- Measures
  - Generate from BASE
  - Generate from calculation items
  - Change format
  - Generate measures from rows
    - Generate from Columns
      - KPI Quantity
      - KPI Sales Amount
  - Sales Amount
  - Value
- Time Intelligence
- Transposed Cities
- Translations

```
1  /* Generate measures from columns */
2
3  var displayTable = Model.Tables["_Measures"];
4  var kpiSourceTable = Model.Tables["Sales"];
5
6  foreach(var column in kpiSourceTable.Columns)
7  {
8      if (column.Name.Contains("KPI "))
9      {
10         var new_measure = displayTable.AddMeasure(
11             column.Name,                                // Name
12             "SUM(" + column.DaxObjectFullName + ")",    // DAX expression
13             "Generate from Columns"                     // Display Folder
14         );
15     }
16 }
```

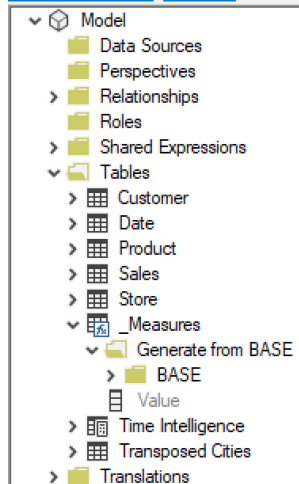


The background features two large, decorative, curved lines. One line, in a light green color, starts at the top left and curves towards the bottom right. The other line, in a light blue color, starts at the top right and curves towards the bottom left. These lines are layered, with the blue line appearing slightly behind the green one in some areas.

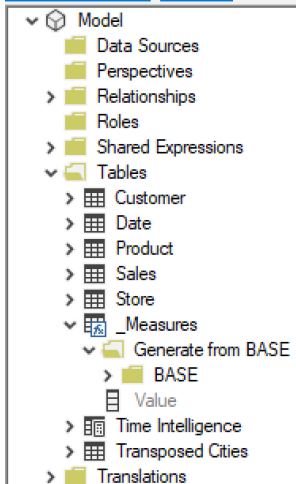
Generate measures from another  
measures

```
1  /* Generate measures from another measures */
2
3  var MeasuresTable = Model.Tables["_Measures"];
4  var baseMeasures = MeasuresTable.Measures.Where(m => (m.DisplayFolder=="Generate from BASE\\BASE")).ToList();
5
6  foreach(var measure in baseMeasures)
7  {
8      // get correct measure name
9      var indexOfBase = measure.Name.IndexOf("BASE");
10     var correctMeasureName = measure.Name.Substring(0, indexOfBase);
11
12     // add AC measure
13     var new_AC_measure = MeasuresTable.AddMeasure(
14         correctMeasureName + " AC", // Name
15         measure.DaxObjectFullName, // DAX expression
16         "Generate from BASE\\AC" // Display Folder
17     );
18
19     // add PY measure
20     var new_PY_measure = MeasuresTable.AddMeasure(
21         correctMeasureName + " PY", // Name
22         "CALCULATE(" + measure.DaxObjectFullName + ", SAMEPERIODLASTYEAR('Date'[Date]))", // DAX expression
23         "Generate from BASE\\PY" // Display Folder
24     );
25 }
26
```

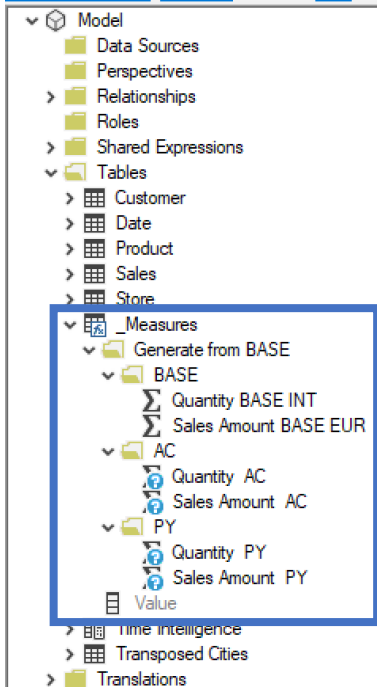




```
1  /* Generate measures from another measures */
2
3  var MeasuresTable = Model.Tables["_Measures"];
4  var baseMeasures = MeasuresTable.Measures.Where(m => (m.DisplayFolder=="Generate from BASE\\BASE")).ToList();
5
6  foreach(var measure in baseMeasures)
7  {
8      // get correct measure name
9      var indexOfBase = measure.Name.IndexOf("BASE");
10     var correctMeasureName = measure.Name.Substring(0, indexOfBase);
11
12     // add AC measure
13     var new_AC_measure = MeasuresTable.AddMeasure(
14         correctMeasureName + " AC", // Name
15         measure.DaxObjectFullName, // DAX expression
16         "Generate from BASE\\AC" // Display Folder
17     );
18
19     // add PY measure
20     var new_PY_measure = MeasuresTable.AddMeasure(
21         correctMeasureName + " PY", // Name
22         "CALCULATE(" + measure.DaxObjectFullName + ", SAMEPERIODLASTYEAR('Date'[Date]))", // DAX expression
23         "Generate from BASE\\PY" // Display Folder
24     );
25 }
26
```



```
1  /* Generate measures from another measures */
2
3  var MeasuresTable = Model.Tables["_Measures"];
4  var baseMeasures = MeasuresTable.Measures.Where(m => (m.DisplayFolder=="Generate from BASE\\BASE")).ToList();
5
6  foreach(var measure in baseMeasures)
7  {
8      // get correct measure name
9      var indexOfBase = measure.Name.IndexOf("BASE");
10     var correctMeasureName = measure.Name.Substring(0, indexOfBase);
11
12     // add AC measure
13     var new_AC_measure = MeasuresTable.AddMeasure(
14         correctMeasureName + " AC", // Name
15         measure.DaxObjectFullName, // DAX expression
16         "Generate from BASE\\AC" // Display Folder
17     );
18
19     // add PY measure
20     var new_PY_measure = MeasuresTable.AddMeasure(
21         correctMeasureName + " PY", // Name
22         "CALCULATE(" + measure.DaxObjectFullName + ", SAMEPERIODLASTYEAR('Date'[Date]))", // DAX expression
23         "Generate from BASE\\PY" // Display Folder
24     );
25 }
26
```



```
1  /* Generate measures from another measures */
2
3  var MeasuresTable = Model.Tables["_Measures"];
4  var baseMeasures = MeasuresTable.Measures.Where(m => (m.DisplayFolder=="Generate from BASE\\BASE")).ToList();
5
6  foreach(var measure in baseMeasures)
7  {
8      // get correct measure name
9      var indexOfBase = measure.Name.IndexOf("BASE");
10     var correctMeasureName = measure.Name.Substring(0, indexOfBase);
11
12     // add AC measure
13     var new_AC_measure = MeasuresTable.AddMeasure(
14         correctMeasureName + " AC", // Name
15         measure.DaxObjectFullName, // DAX expression
16         "Generate from BASE\\AC" // Display Folder
17     );
18
19     // add PY measure
20     var new_PY_measure = MeasuresTable.AddMeasure(
21         correctMeasureName + " PY", // Name
22         "CALCULATE(" + measure.DaxObjectFullName + ", SAMEPERIODLASTYEAR('Date'[Date]))", // DAX expression
23         "Generate from BASE\\PY" // Display Folder
24     );
25 }
26
```

The image features decorative curved lines in the top-left and top-right corners. These lines are composed of multiple overlapping layers in shades of light green and light blue, creating a sense of depth and movement.

Delete measures



- Model
  - Data Sources
  - Perspectives
  - Relationships
  - Roles
  - Shared Expressions
  - Tables
    - Customer
    - Date
    - Product
    - Sales
    - Store
  - Measures
    - Generate from BASE
    - TO DELETE
      - Measure to delete
      - Value
  - Time Intelligence
  - Transposed Cities
  - Translations

```
1 /* Delete measures */
2
3 var MeasuresTable = Model.Tables["_Measures"];
4 foreach(var m in MeasuresTable.Measures.Where(m => (m.DisplayFolder.Contains("TO DELETE"))).ToList())
5 {
6     m.Delete();
7 }
```

# Thank You!



**Paulina Jedrzejewska**  
Senior Consultant b.telligent

