The Godfather of Talent | Futurense

Power BI Cheat Sheet

Data Visualization

VISUALIZE RELATIONSHIP

BAR CHART

Use Case

- Volume of google searches by region
- Market share in revenue by product

COLUMN CHART

Use Case

- Brand market share
- Profit Analysis by region

WORLD CLOUD CHART

Use Case

- Top 100 used words by customers in customer service tickets

CONNECTED SCATTERPLOT

Use Case

- Cryptocurrency price index
- Visualizing timelines and events when analyzing two variables

BUBBLE CHARTS

Use Case

- Adwords analysis: CPC vs Conversions vs Share of total conversions
- Relationship between life expectancy, GDP per capita, & population size

SCATTER PLOT

Use Case

- Display the relationship between time-on-platform and churn
- Display the relationship between salary and years spent at company

CAPTURE A TREND

LINE CHART

Use Case

- Revenue in $ over time
- Google searches over time

MULTI-LINE CHART

Use Case

- Apple vs Amazon stocks over time
- Bitcoin vs Ethereum price over time

AREA CHART

Use Case

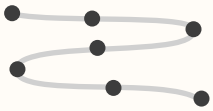
- Total sales over time
- Active users over time

STACKED AREA CHART

Use Case

- Active users over time by segment
- Total revenue over time by country

SPLINE CHART



Use Case

- Electricity consumption over time
- CO2 emissions over time

The Godfather of Talent | Futurense

PART-TO-WHOLE CHARTS

PIE CHART

Use Case

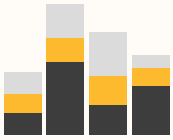
- Voting preference by age group
- Market share of cloud providers

DONUT PIE CHART

Use Case

- Monthly sales by channel
- Android OS market share

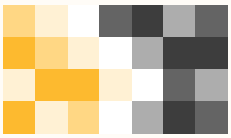
STACKED COLUMN CHART



Use Case

- Total car sales by producer
- Quarterly sales per region

HEAT MAPS



Use Case

- Monthly averages for the entire year
- Departments with the highest amount of attrition over time

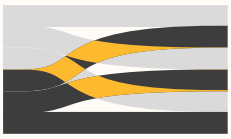
TREEMAP CHARTS

Use Case

- Grocery sales count with categories
- Stock price comparison by industry and company

VISUALIZE A FLOW

SANKEY CHART



Use Case

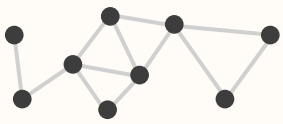
- Energy flow between countries
- Supply chain volumes between warehouses

CHORD CHART

Use Case

- Export between countries to showcase biggest export partners
- Supply chain volumes between the largest warehouses

NETWORK CHART

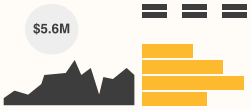


Use Case

- How different airports are connected worldwide
- Social media friend group analysis

VISUALIZE A SINGLE VALUE

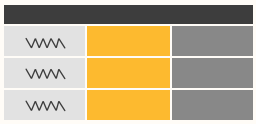
CARD



Use Case

- Revenue to date on a sales dashboard
- Total sign-ups after a promotion

TABLE CHART



Use Case

- Account executive leaderboards
- Registrations per webinar

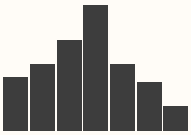
GAUGE CHART

Use Case

- NPS scores
- Revenue to target

CAPTURE DISTRIBUTIONS

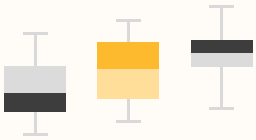
HISTOGRAMS



Use Case

- Distribution of salaries in an organization
- Distribution of height in one cohort

BOX PLOT



Use Case

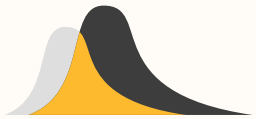
- Gas efficiency of vehicles
- Time spent reading across readers

VIOLIN PLOT

Use Case

- Time spent in restaurants across age groups
- Length of pill effects by dose

DENSITY PLOT



Use Case

- Distribution of price of hotel listings
- Comparing NPS scores by customer segment

Power Query M Formula Language

READING DATA FROM VARIOUS SOURCES

- **SQL SERVER**

```
let
    Source = Sql.Database("localhost",
             "DatabaseName")
in
    Source
```

- **EXCEL FILE**

```
let
    Source = Excel.Workbook(File.Contents
            ("C:\path\file.xlsx"), null, true)
in
    Source
```

- **ORACLE**

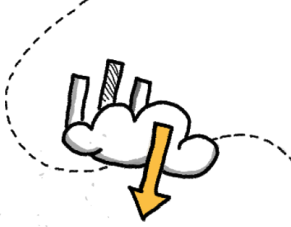
```
let
    Source = Oracle.Database("localhost:1521/orclpdb1.localdomain")
in
    Source
```

- **API (JSON RESPONSE)**

```
let
    Source = Json.Document(Web.Contents
        ("https://api.example.com/resource"))
in
    Source
```

- **WEB PAGE**

```
let
    Source = Web.Page(Web.Contents
            ("https://www.example.com"))
in
    Source
```



- **CSV FILE**

```
let
    Source = Csv.Document(File.Contents("C:\path\file.csv"),[Delimiter=",", Columns=3,
             Encoding=1252, QuoteStyle=QuoteStyle.None])
in
    Source
```

FILTERING AND SORTING DATA

- **FILTERING DATA**

```
let
    Source = Excel.Workbook(File.Contents("C:\path\to\your\file.xlsx"), null, true),
    //Filter rows where 'Column1' is greater than 100
    FilteredRows = Table.SelectRows(Source, each [Column1] > 100)
in
    FilteredRows
```

- **REMOVING ROWS WITH NULL VALUES**

```
let
    Source = Excel.Workbook(File.Contents("C:\path\to\your\file.xlsx"), null, true),
    //Remove rows where 'Column1' is null
    NoNulls = Table.SelectRows(Source, each [Column1] <> null)
in
    NoNulls
```

- **FILTERING TEXT**

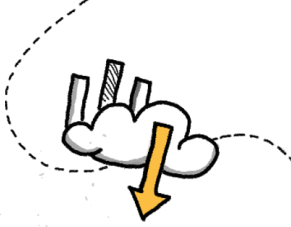
```
let
    Source = Excel.Workbook(File.Contents("C:\path\to\your\file.xlsx"), null, true),
    //Keep rows where 'Column1' contains "example"
    FilteredRows = Table.SelectRows(Source, each Text.Contains([Column1], "example"))
in
    FilteredRows
```

- **SORTING DATA**

```
let
    Source = Excel.Workbook(File.Contents("C:\path\to\your\file.xlsx"), null, true),
    //Sort by 'Column1' in ascending order
    SortedRows = Table.Sort(Source,{{"Column1", Order.Ascending}})
in
    SortedRows
```

- **SORTING DATA** (Multiple Columns)

```
let
    Source = Excel.Workbook(File.Contents("C:\path\to\your\file.xlsx"), null, true),
    //Sort by 'Column1' in ascending order and 'Column2' in descending order
    SortedRows = Table.Sort(Source,{{"Column1", Order.Ascending}, {"Column2", Order.Descending}})
in
    SortedRows
```



MERGING AND APPENDING QUERIES

- **MERGING QUERIES**

```
let
    //Create table 1
    Table1 = #table({"ID", "Name"}, {{1, "John"}, {2, "Bob"}, {3, "Alice"}}),
    //Create table 2
    Table2 = #table({"ID", "Age"}, {{1, 25}, {2, 30}, {3, 35}}),
    //Merge the tables on the 'ID' column
    MergedTables = Table.NestedJoin(Table1,{"ID"},Table2,{"ID"}, "NewColumn", JoinKind.LeftOuter)
in
    MergedTables
```

Note: *There are different types of joins you can perform:* **JoinKind.Inner**, **JoinKind.LeftOuter**, **JoinKind.RightOuter**, **JoinKind.FullOuter**, **JoinKind.LeftAnti**, **JoinKind.RightAnti**.

- **EXPANDING MERGED QUERIES**

After a merge, the columns from the second table are nested into a single column. You can expand these columns using the **Table.ExpandTableColumn** function.

```
ExpandedTables = Table.ExpandTableColumn(MergedTables, "NewColumn", {"Age"}, {"Age"})
```

- **APPENDING QUERIES**

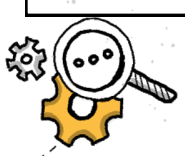
```
let
    //Create table 1
    Table1 = #table({"ID", "Name"}, {{1, "John"}, {2, "Bob"}, {3, "Alice"}}),
    //Create table 2
    Table2 = #table({"ID", "Name"}, {{4, "Jane"}, {5, "Steve"}, {6, "Emma"}}),
    //Append the tables
    AppendedTables = Table.Combine({Table1, Table2})
in
    AppendedTables
```

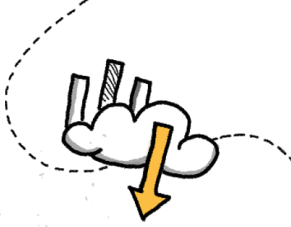
GROUPING ROWS AND AGGREGATING DATA

- **GROUPING ROWS**

```
let
    //Create a table
    Source = #table({"ID", "Score"}, {{1, 90}, {1, 85}, {2, 70}, {2, 75}, {3, 80}, {3, 90}}),
    //Group by 'ID' and get the count for each group
    GroupedRows = Table.Group(Source, {"ID"}, {{"Count", each Table.RowCount(_), type number}})
in
    GroupedRows
```

- **AGGREGATING DATA**
  - `each Table.RowCount(_)` : Count
  - `each List.Sum([Score])` : Sum
  - `each List.Average([Score])` : Average
  - `each List.Min([Score])` : Minimum
  - `each List.Max([Score])` : Maximum
  - `each List.StandardDeviation([Score])` : Standard deviation





- **MULTIPLE AGGREGATIONS**

```
let
    //Create a table
    Source = #table({"ID", "Score"}, {{1, 90}, {1, 85}, {2, 70}, {2, 75}, {3, 80}, {3, 90}}),
    //Group by 'ID' and get the count, sum, and average of 'Score' for each group
    GroupedRows = Table.Group(Source, {"ID"}, {{"Count", each Table.RowCount(_), type number},
                  {"Total Score", each List.Sum([Score]), type number}, {"Average Score",
                  each List.Average([Score]), type number}})
in
    GroupedRows
```

CONDITIONAL LOGIC

- **CREATING A CONDITIONAL COLUMN**

```
let
    //Create a table
    Source = #table({"ID", "Score"}, {{1, 90}, {2, 70}, {3, 80}}),
    //Add a conditional column
    AddedConditionalColumn = Table.AddColumn(Source,"Grade", each if [Score]>=80 then "A" else "B")
in
    AddedConditionalColumn
```

- **FILTERING ROWS CONDITIONALLY**

```
//Filter rows where 'Score' is greater than 75
FilteredRows = Table.SelectRows(Source, each [Score] > 75)
```

- **USING MULTIPLE CONDITIONS**

```
//Add a conditional column
AddedConditionalColumn = Table.AddColumn(Source, "Grade", each if [Score] >= 80 then "A"
                         else if [Score] >= 70 then "B" else "C")
```

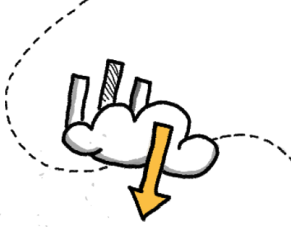
ERROR HANDLING

- **USING TRY EXPRESSION**

```
let
    //Create a table
    Source = #table({"ID", "Score"}, {{1, 90}, {2, "Not a number"}, {3, 80}}),
    //Add a column with calculation that can fail
    AddedColumn = Table.AddColumn(Source, "Score Plus Ten", each try [Score] + 10)
in
    AddedColumn
```

- **USING OTHERWISE EXPRESSION**

```
let
    //Create a table
    Source = #table({"ID", "Score"}, {{1, 90}, {2, "Not a number"}, {3, 80}}),
    //Add a column with calculation that can fail
    AddedColumn = Table.AddColumn(Source, "Score Plus Ten", each try [Score] + 10 otherwise null)
in
    AddedColumn
```



- **REPLACING ERRORS**

```
let
    //Create a table
    Source = #table({"ID", "Score"}, {{1, 90}, {2, "Not a number"}, {3, 80}}),
    //Add a column with calculation that can fail
    AddedColumn = Table.AddColumn(Source, "Score Plus Ten", each try [Score] + 10),
    //Replace errors with null
    ReplacedErrors = Table.ReplaceErrorValues(AddedColumn, {{"Score Plus Ten", null}})
in
    ReplacedErrors
```

DAX (Data Analysis Expressions)

SYNTAX AND OPERATORS

- **SYNTAX**

```
<expression> ::=
    <function> ( [<expression> [, <expression> [, … ] ] ] )
  | <value>
  | <column reference>
  | <table>
  | <constant>
```

Example:

```
SUM('Sales'[Amount])
```

- **OPERATORS**
  - **Arithmetic:** `+` , `-` , `*` , `/` , `^`

    Example:

    ```
    [Column1] + [Column2]
    ```

  - **Comparison:** `=` , `>` , `<` , `>=` , `<=` , `<>`

    Example:

    ```
    IF([Col1] > [Col2], "Greater","Smaller")
    ```

  - **Text Concatenation:** `&`

    Example:

    ```
    [Column1] & [Column2]
    ```

  - **Logical:** `&&` (AND) , `||` (OR), `!` (NOT)

    Example:

    ```
    IF([Col1] > 100 && [Col2] < 50, "Yes", "No")
    ```

- **VARIABLES**

  Syntax:

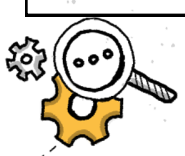
  ```
  VAR VariableName = Expression
  RETURN
  ExpressionUsingVariable
  ```

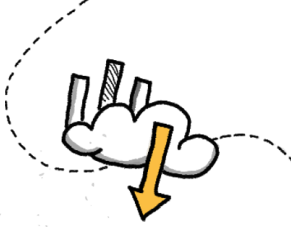
  Example:

  ```
  VAR TotalSales = SUM('Sales'[Sales Amount])
  VAR TotalCost = SUM('Sales'[Cost])
  RETURN
  TotalSales - TotalCost
  ```

FUNCTIONS

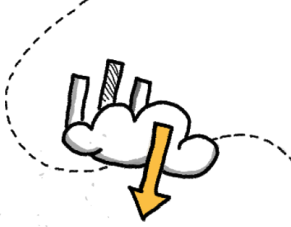
- **AGGREGATION FUNCTIONS**
  - `SUM(column)` : Returns the sum of all the numbers in the column.
  - `AVERAGE(column)` : Returns the average of the numbers in the column.
  - `COUNT(column)` : Counts the number of rows in the column.





- `MIN(column)` : Returns smallest number in a column.
- `MAX(column)` : Returns largest number in a column.

- **TEXT FUNCTIONS**
  - `LEFT(text, num_chars)` : Returns the first (left-most) num_chars characters from a text string.
    Example:
    ```
    ProductShortName = LEFT('Product'[ProductName], 5)
    ```
  - `RIGHT(text, num_chars)` : Returns the last (right-most) num_chars characters from a text string.
  - `LEN(text)` : Returns the number of characters in a text string.
  - `REPLACE(old_text, start_num, num_chars, new_text)` : Replaces part of a text string with a different text string.
    Example:
    ```
    UpdatedProduct = REPLACE('Product'[ProductName], 1, 5, "NewProd")
    ```
  - `SUBSTITUTE(text, old_text, new_text)` : Replaces existing text with new text in a text string.
- **DATE AND TIME FUNCTIONS**
  - `YEAR(date)` : Returns year of a date as a four-digit number.
  - `MONTH(date)` : Returns month of a date as a number.
  - `DAY(date)` : Returns the day of a date as a number.
  - `TODAY()` : Returns the current date.
  - `NOW()` : Returns the current date and time.
- **MATH AND TRIGONOMETRY FUNCTIONS**
  - `ROUND(number, num_digits)` : Rounds a number to a specified number of digits.
  - `SQRT(number)` : Returns year of a date as a four-digit number.
  - `POWER(number, power)` : Raises the number to the power.
- **STATISTICAL FUNCTIONS**
  - `STDEV.P(column)` : Calculates the standard deviation based on the entire population.
  - `VAR.P(column)` : Calculates the variance of an entire population.
  - `RANKX(table, expression)` : Returns the rank of an expression evaluated in the current context in the list of values for the expression evaluated for each row in the specified table.
- **FILTER FUNCTIONS**
  - `ALL(column)` : Removes all context filters in the table or columns provided.
    Example:
    ```
    TotalSales = CALCULATE(SUM('Sales'[SalesAmount]), ALL('Sales'))
    ```
  - `FILTER(table, expression)` : Returns a table that has been filtered.
    Example:
    ```
    HighValueSales = CALCULATE(SUM('Sales'[SalesAmount]), FILTER('Sales', 'Sales'[SalesAmount]>100))
    ```
  - `CALCULATE(expression, filter)` : Evaluates an expression in a context modified by filters.



CONTEXTS

- **ROW CONTEXT** : Each line of the M code operates in a row context.

```
let
    Source = #"Sales Table",
    NewColumn = Table.AddColumn(Source, "Total", each [Quantity] * [Price])
in
    NewColumn
```

Each [Quantity] * [Price] *is operating in a row context*

- **QUERY CONTEXT** : The query context here is the order of transformations in our Power Query script.

```
let
    Source = #"Sales Table",
    RemovedColumns = Table.RemoveColumns(Source,{"Region"}),
    RenamedColumns = Table.RenameColumns(RemovedColumns,{{"Sales Amount", "Revenue"}})
in
    RenamedColumns
```

- **FILTER CONTEXT** : This can be related to any operation that filters or limits the rows in a table. Here's an example of filtering a table to include only rows where "Region" is "West"

```
let
    Source = #"Sales Table",
    FilteredRows = Table.SelectRows(Source, each [Region] = "West")
in
    FilteredRows
```

CALCULATED COLUMNS AND TABLES

- **CALCULATED COLUMNS**

If we have a sales table and we want to add a new calculated column for the total cost of a transaction:

```
Sales[TotalCost] = Sales[Quantity] * Sales[Price]
```

Here Sales[Quantity] and Sales[Price] are existing columns in the Sales table. Sales[TotalCost] is the new calculated column that we are creating using a DAX formula.

- **CALCULATED TABLES**

We can create a new calculated table that includes only the 2022 sales data.

```
Sales2022 = FILTER(Sales, YEAR(Sales[OrderDate]) = 2022)
```

Advanced DAX Techniques

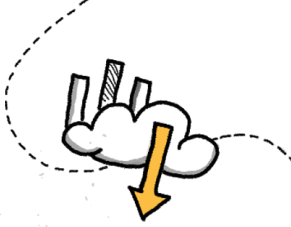
TIME INTELLIGENCE

- **CALENDAR FUNCTION**

```
CALENDAR(<start_date>, <end_date>)
```

Example:

```
DateTable = CALENDAR(DATE(2020, 1, 1), DATE(2020, 12, 31))
```



- **SAMEPERIODLASTYEAR FUNCTION**

```
SAMEPERIODLASTYEAR(<dates>)
```

Example:

```
Sales LY = CALCULATE(SUM('Sales'[SalesAmount]), SAMEPERIODLASTYEAR('Date'[Date]))
```

In this example, SAMEPERIODLASTYEAR function is used inside the CALCULATE function. It modifies the filter context to include the same period last year. The SUM function then adds up the 'SalesAmount' for only these rows.

- **TOTALYTD FUNCTION**

```
TOTALYTD(<expression>, <dates>[, <year_end_date>])
```

Example:

```
Sales YTD = TOTALYTD(SUM('Sales'[SalesAmount]), 'Date'[Date])
```

In this example, TOTALYTD function calculates the year-to-date sum of 'SalesAmount'. The year-to-date is calculated for the dates specified in the 'Date'[Date].

- **DATESYTD FUNCTION**

```
DATESYTD(<dates>[, <year_end_date>])
```

- **DATEADD FUNCTION**

```
DATEADD(<dates>, <number_of_intervals>, <interval>)
```

Example:

```
Sales Last Month = CALCULATE(SUM('Sales'[SalesAmount]), DATEADD('Date'[Date], -1, MONTH))
```

This measure Sales Last Month calculates the sales amount for the previous month

- **EDATE FUNCTION**

```
EDATE(<start_date>, <months>)
```

This function returns a date a specified number of months before or after a start date.

ITERATOR FUNCTIONS

- **GENERAL SYNTAX**

```
<function>X(<table>, <expression>)
```

- **SUMX FUNCTION**

```
SUMX(<table>, <expression>)
```

Example:

```
Total Sales = SUMX('Sales', 'Sales'[Quantity] * 'Sales'[Unit Price])
```

In this example, SUMX iterates over each row in the 'Sales' table and multiplies the 'Quantity' by the 'Unit Price' for each row. Then, it sums up these results.

- **AVERAGEX FUNCTION**

```
AVERAGEX(<table>, <expression>)
```

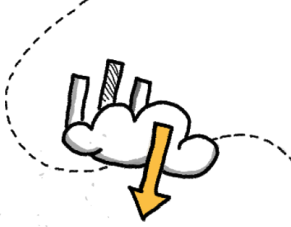
- **COUNTX FUNCTION**

```
COUNTX(<table>, <expression>)
```

Example:

```
Count Sales = COUNTX('Sales', 'Sales'[SalesID])
```

In this example, COUNTX iterates over each row in the 'Sales' table and counts the number of rows where 'SalesID' is not blank.



RANKX FUNCTION

```
RANKX(<table>, <expression>[, <value>[, <order>[, <ties>]]])
```

Example:

```
Rank by Sales = RANKX(ALL('Sales'[Product]), CALCULATE(SUM('Sales'[SalesAmount])), , DESC, Dense)
```

In this example, RANKX function is used to calculate the rank of each product by its total sales amount. ALL function is used to remove any filters on the 'Product' column. CALCULATE is used to sum the 'SalesAmount' for each product. The DESC argument is used to sort the ranks in descending order . Dense argument is used to handle ties.

VARIABLES IN DAX

- **GENERAL SYNTAX**

```
VAR <variable_name> = <expression>
```

Now we can use the variable in your calculations using the RETURN statement:

```
RETURN <variable_name>
```

- **SIMPLE VARIABLE EXAMPLE**

```
Total Sales =
VAR TotalSalesAmount = SUM('Sales'[SalesAmount])
RETURN TotalSalesAmount
```

- **USING MULTIPLE VARIABLES**

```
Sales Analysis =
VAR TotalSalesAmount = SUM('Sales'[SalesAmount])
VAR TotalCostAmount = SUM('Sales'[CostAmount])
VAR ProfitAmount = TotalSalesAmount - TotalCostAmount
RETURN ProfitAmount
```

- **USING VARIABLES WITH TABLES**

```
Top Sales Product =
VAR SalesTable = ALL('Sales')
VAR TopProduct =
    TOPN( 1, SalesTable, CALCULATE(SUM('Sales'[SalesAmount])), DESC )
RETURN SELECTCOLUMNS(TopProduct, "Product", 'Sales'[Product])
```

Data Modeling

RELATIONSHIPS AND RELATIONSHIP FUNCTIONS

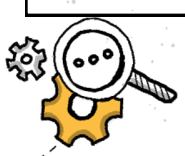
FactSales
SalesID
ProductID
CustomerID
SalesAmount
Quantity

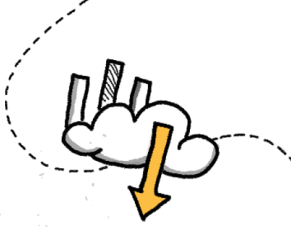
DimProduct
ProductID
ProductName
ProductCategory

DimCustomer
CustomerName
ProductName
ProductCategory

Consider the example of a simple star schema with FactSales, DimProduct, and DimCustomer tables

Here, ProductID and CustomerID are related between FactSales and DimProduct, and FactSales and DimCustomer respectively.





- **DAX RELATIONSHIP FUNCTIONS**

1. **RELATED:** Returns related value from another table.

```
Product Name = RELATED(DimProduct[ProductName])
```

2. **RELATEDTABLE:** Returns a table that has been filtered to include all the rows related to the current row.

```
Customer Purchases = CALCULATE(COUNTROWS('FactSales'),RELATEDTABLE('FactSales'))
```

3. **USERELATIONSHIP:** Allows you to specify which relationship to use in a calculation when there are multiple relationships between tables.

```
Sales Amount (Inactive Relationship) =
CALCULATE(
    SUM('FactSales'[SalesAmount]),
    USERELATIONSHIP('FactSales'[OtherProductID], 'DimProduct'[ProductID])
)
```

DAX HIERARCHY FUNCTIONS

- **PATH :** Returns a delimited text string with identifiers of all the parents of the current identifier, starting with the oldest.

```
EmployeePath = PATH(Employee[EmployeeID], Employee[ManagerID])
```

Here, the PATH function creates a string that identifies the hierarchy of employees based on who manages whom.

- **PATHITEM :** Returns the item at the specified position in a path. This function is often used with PATH.

```
ManagerID = PATHITEM(Employee[EmployeePath], 1)
```

Here, PATHITEM returns the first item (the top-level manager) in the path created by the PATH function.

- **PATHLENGTH :** Returns the number of items in a path.

```
PathLength = PATHLENGTH(Employee[EmployeePath])
```

Here, PATHLENGTH calculates the number of levels in the hierarchy for each employee.

Power BI API and PowerShell Cmdlets

IMPORTING AND PUBLISHING REPORTS

To work with PowerShell and the Power BI REST API, you need the Power BI Management module. Install it using PowerShell. After the module is installed, you can use Login-PowerBI to authenticate your Power BI account.

- **IMPORTING REPORTS** *(Powershell)*

```
New-PowerBIReport -Path ".\myReport.pbix" -WorkspaceId "workspace-id" -Name "My New Report"
```

- **PUBLISHING REPORTS**

```
New-PowerBIReport -Path ".\myReport.pbix" -WorkspaceId "workspace-id" -Name "My Published Report"
```

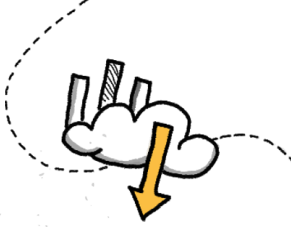
MANAGING WORKSPACES AND DATASETS

- **GET ALL WORKSPACES**

```
Get-PowerBIWorkspace
```

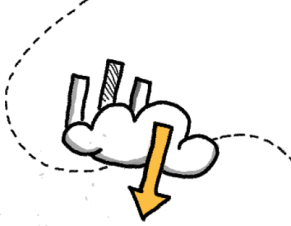
- **CREATE A WORKSPACE**

```
New-PowerBIWorkspace -Name 'My New Workspace'
```



- **REMOVE A WORKSPACE**
  `Remove-PowerBIWorkspace -Id 'workspace-id'`
- **GET DATASETS IN A WORKSPACE**
  `Get-PowerBIDataset -WorkspaceId 'workspace-id'`
- **REFRESH A DATASET**
  `Invoke-PowerBIDatasetRefresh -DatasetId 'dataset-id' -WorkspaceId 'workspace-id'`
- **REMOVE A DATASET**
  `Remove-PowerBIDataset -Id 'dataset-id' -WorkspaceId 'workspace-id'`

The Godfather of Talent | Futurense



How to use ChatGPT Effectively for Power BI Workflow

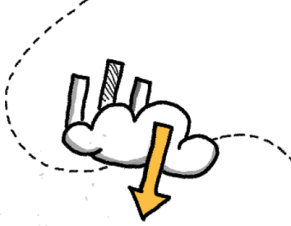
ChatGPT and CodeGPT Prompts for Power BI

BASIC TOPICS

- **CREATING A NEW MEASURE USING DAX IN POWER BI**
  - **ChatGPT:** *"What steps should I follow to create a new measure using DAX in Power BI?"*
  - **CodeGPT:** *"Provide a DAX example of creating a new measure."*
- **APPLYING FILTERS TO DATA IN POWER BI**
  - **ChatGPT:** *"Can you guide me on how to apply filters to my data in Power BI?"*
  - **CodeGPT:** *"Demonstrate how to apply filters in Power Query M."*
- **VISUALIZATIONS IN POWER BI**
  - **ChatGPT:** *"What are the steps to create a bar chart visualization in Power BI?"*
  - **CodeGPT:** *"Provide a Python script to create a bar chart using matplotlib within Power BI."*
- **SCHEDULING REFRESH FOR POWER BI REPORTS**
  - **ChatGPT:** *"Can you explain how to schedule data refresh for my Power BI reports?"*
  - **CodeGPT:** *"Show me a PowerShell cmdlet that can be used to refresh a dataset in Power BI."*
- **USING R SCRIPTS IN POWER BI**
  - **ChatGPT:** *"How can I use R scripts for data transformation in Power BI?"*
  - **CodeGPT:** *"Provide an example of an R script used for data transformation in Power BI."*
- **IMPLEMENTING MACHINE LEARNING MODELS IN POWER BI**
  - **ChatGPT:** *"What is the procedure to implement a machine learning model from scikit-learn in Power BI?"*
  - **CodeGPT:** *"Give an example of implementing a linear regression model using scikit-learn within a Python script in Power BI."*

ADVANCED TOPICS

- **DVANCED DAX TECHNIQUES FOR TIME INTELLIGENCE IN POWER BI**
  - **ChatGPT:** *"Can you explain advanced DAX techniques for time intelligence functions like SAMEPERIODLASTYEAR and TOTALYTD in Power BI?"*
  - **CodeGPT:** *"Show me DAX code that uses SAMEPERIODLASTYEAR and TOTALYTD functions for time intelligence."*
- **CREATING AND MANAGING POWER BI WORKSPACES USING POWERSHELL CMDLETS**
  - **ChatGPT:** *"Can you guide me on how to create and manage Power BI workspaces using PowerShell cmdlets?"*
  - **CodeGPT:** *"Demonstrate PowerShell cmdlets for creating a new workspace and adding a user to it in Power BI."*



- **MERGING AND APPENDING QUERIES IN POWER BI**
  - **ChatGPT:** *"What is the method for merging and appending queries in Power BI using the M language?"*
  - **CodeGPT:** *"Provide M language syntax to merge and append queries in Power BI."*
- **ERROR HANDLING IN POWER QUERY M LANGUAGE IN POWER BI**
  - **ChatGPT:** *"Can you explain how to handle errors in Power Query M language in Power BI?"*
  - **CodeGPT:** *"Show me an example of error handling in Power Query M language."*
- **USING PYTHON FOR MACHINE LEARNING IN POWER BI**
  - **ChatGPT:** *"How can I use Python's scikit-learn library for machine learning within Power BI"*
  - **CodeGPT:** *"Give an example of a Python script that uses a scikit-learn machine learning model in Power BI."*
- **CREATING COMPLEX RELATIONSHIPS AND HIERARCHIES IN POWER BI DATA MODELING**
  - **ChatGPT:** *"What steps should I follow to create complex relationships and hierarchies in Power BI data modeling?"*
  - **CodeGPT:** *"Demonstrate a DAX script that creates a hierarchy in Power BI."*
- **USING R SCRIPTS FOR ADVANCED DATA TRANSFORMATIONS IN POWER BI**
  - **ChatGPT:** *"Can you guide me on using R scripts for advanced data transformations in Power BI?"*
  - **CodeGPT:** *"Provide an example of an R script used for advanced data transformation in Power BI."*
- **IMPLEMENTING ROW LEVEL SECURITY IN POWER BI**
  - **ChatGPT:** *"Can you guide me on how to implement row level security in Power BI?"*
  - **CodeGPT:** *"Show me a DAX example implementing row level security in Power BI."*
- **EMBEDDING POWER BI REPORTS IN APPLICATIONS**
  - **ChatGPT:** *"What is the process of embedding Power BI reports into an application?"*
  - **CodeGPT:** *"Provide an example of embedding a Power BI report in an HTML page using JavaScript."*
- **ADVANCED QUERY TRANSFORMATIONS IN POWER QUERY M**
  - **ChatGPT:** *"Can you explain some advanced transformations I can do in Power Query M in Power BI?"*
  - **CodeGPT:** *"Show me an advanced transformation using Power Query M in Power BI."*
- **USING PYTHON OR R FOR ADVANCED DATA CLEANING IN POWER BI**
  - **ChatGPT:** *"What are the steps to use Python or R for advanced data cleaning in Power BI?"*
  - **CodeGPT:** *"Provide a Python/R script example for advanced data cleaning in Power BI."*