Data Analysis Expressions (DAX)

DAX is a formula language designed for data modeling and analysis, used primarily with Microsoft Power BI, Power Pivot, and Analysis Services. It enables users to create custom calculations, aggregations, and advanced data analysis.

Introduction to DAX

**Definition**: DAX (Data Analysis Expressions) is a collection of functions, operators, and constants used to create formulas and expressions in Power BI, Power Pivot, and SQL Server Analysis Services Tabular models.

**Use Case**: DAX is used to perform data calculations, create measures and calculated columns, and enhance data analysis capabilities.

Difference Between M Query and DAX Query

* **M Query**:
  + **Definition**: A language used in Power Query Editor for data transformation.
  + **Use Case**: Used to extract, transform, and load (ETL) data before it is loaded into the data model.
  + **Syntax Example**:

let  
    Source = Excel.Workbook(File.Contents("C:\Data\SalesData.xlsx"), null, true),  
    SalesData\_Sheet = Source{[Item="SalesData",Kind="Sheet"]}[Data]  
in  
    SalesData\_Sheet

* **DAX Query**:
  + **Definition**: A formula language used for data analysis and creating calculations in the data model.
  + **Use Case**: Used to create measures, calculated columns, and tables for data analysis within the data model.
  + **Syntax Example**:

EVALUATE  
TOPN(10, SUMMARIZE(Orders, Orders[ProductID], "TotalSales", SUM(Orders[SalesAmount])))

DAX Data Types and DAX Operators

**Data Types**:

* **Numeric**: Integer, Decimal, Currency.
* **Boolean**: TRUE/FALSE.
* **Text**: String.
* **DateTime**: Date and time.
* **Variant**: Special type that can hold any other type.

**Operators**:

* **Arithmetic Operators**: +, -, \*, /, ^.
* **Comparison Operators**: =, <>, <, <=, >, >=.
* **Text Concatenation**: &.
* **Logical Operators**: AND, OR, NOT.

DAX Measures and Calculations

**Measures**:

* **Definition**: Dynamic calculations that are evaluated based on the filter context of the report.
* **Syntax**:

Total Sales = SUM(Orders[SalesAmount])

* **Example**:

Total Revenue = SUM(Orders[Revenue])

* **Use Case**: Use measures for aggregations and dynamic calculations in reports and dashboards.

**Calculated Columns**:

* **Definition**: Columns added to the data model that are calculated row by row.
* **Syntax**:

Year = YEAR(Orders[OrderDate])

* **Example**:

Profit Margin = Orders[Profit] / Orders[Sales]

* **Use Case**: Use calculated columns for row-wise calculations that do not change based on the filter context.

ROW Context and Filter Context

* **Row Context**:
  + **Definition**: The context of a single row in a table.
  + **Use Case**: Used in calculated columns and row-level calculations.
  + **Example**:

Extended Price = Orders[Quantity] \* Orders[UnitPrice]

* **Filter Context**:
  + **Definition**: The set of filters applied to data when calculating measures.
  + **Use Case**: Used in measures and dynamic calculations.
  + **Example**:

Total Sales = CALCULATE(SUM(Orders[SalesAmount]), Orders[Region] = "North America")

Aggregation Functions

**SUM**:

* **Definition**: Adds all the numbers in a column.
* **Syntax**:

SUM(<column>)

* **Example**:

Total Sales = SUM(Orders[SalesAmount])

* **Use Case**: To calculate the total sum of a column.

**SUMX**:

* **Definition**: Adds the result of an expression evaluated for each row in a table.
* **Syntax**:

SUMX(<table>, <expression>)

* **Example**:

Total Revenue = SUMX(Orders, Orders[Quantity] \* Orders[UnitPrice])

* **Use Case**: To perform row-wise calculations and then sum the results.

**COUNT**:

* **Definition**: Counts the number of values in a column.
* **Syntax**:

COUNT(<column>)

* **Example**:

Order Count = COUNT(Orders[OrderID])

* **Use Case**: To count the number of non-blank entries in a column.

**COUNTX**:

* **Definition**: Counts the number of rows that result from evaluating an expression for each row in a table.
* **Syntax**:

COUNTX(<table>, <expression>)

* **Example**:

Large Orders Count = COUNTX(Orders, IF(Orders[Quantity] > 10, 1, BLANK()))

* **Use Case**: To count rows based on a condition.

**MIN**:

* **Definition**: Returns the smallest numeric value in a column.
* **Syntax**:

MIN(<column>)

* **Example**:

Min Sales = MIN(Orders[SalesAmount])

* **Use Case**: To find the minimum value in a column.

**MAX**:

* **Definition**: Returns the largest numeric value in a column.
* **Syntax**:

MAX(<column>)

* **Example**:

Max Sales = MAX(Orders[SalesAmount])

* **Use Case**: To find the maximum value in a column.

**AVERAGE**:

* **Definition**: Calculates the average of the values in a column.
* **Syntax**:

AVERAGE(<column>)

* **Example**:

Average Sales = AVERAGE(Orders[SalesAmount])

* **Use Case**: To calculate the average value of a column.

**AVERAGEX**:

* **Definition**: Calculates the average of an expression evaluated for each row in a table.
* **Syntax**:

AVERAGEX(<table>, <expression>)

* **Example**:

Average Revenue = AVERAGEX(Orders, Orders[Quantity] \* Orders[UnitPrice])

* **Use Case**: To perform row-wise calculations and then average the results.

Text Functions

**CONCATENATE**:

* **Definition**: Combines two strings.
* **Syntax**:

CONCATENATE(<text1>, <text2>)

* **Example**:

FullName = CONCATENATE(Customers[FirstName], " " & Customers[LastName])

* **Use Case**: To combine text values from different columns.

**LEFT**:

* **Definition**: Extracts the specified number of characters from the start of a string.
* **Syntax**:

LEFT(<text>, <num\_chars>)

* **Example**:

FirstInitial = LEFT(Customers[FirstName], 1)

* **Use Case**: To extract the initial part of a string.

**RIGHT**:

* **Definition**: Extracts the specified number of characters from the end of a string.
* **Syntax**:

RIGHT(<text>, <num\_chars>)

* **Example**:

LastFourDigits = RIGHT(Customers[Phone], 4)

* **Use Case**: To extract the ending part of a string.

**LEN**:

* **Definition**: Returns the number of characters in a string.
* **Syntax**:

LEN(<text>)

* **Example**:

NameLength = LEN(Customers[FirstName])

* **Use Case**: To find the length of a string.

**UPPER**:

* **Definition**: Converts a string to uppercase.
* **Syntax**:

UPPER(<text>)

* **Example**:

UpperCaseName = UPPER(Customers[FirstName])

* **Use Case**: To convert text to uppercase.

**LOWER**:

* **Definition**: Converts a string to lowercase.
* **Syntax**:

LOWER(<text>)

* **Example**:

LowerCaseName = LOWER(Customers[FirstName])

* **Use Case**: To convert text to lowercase.

Date and Time Functions

**CALENDAR**:

* **Definition**: Returns a table with a single column named "Date" containing a contiguous set of dates.
* **Syntax**:

CALENDAR(<start\_date>, <end\_date>)

* **Example**

:

  Date = CALENDAR(MIN(Orders[OrderDate]), MAX(Orders[OrderDate]))

* **Use Case**: To create a date table for use in time intelligence calculations.

**CALENDARAUTO**:

* **Definition**: Creates a date table over a range based on data in the model.
* **Syntax**:

CALENDARAUTO()

* **Example**:

Date = CALENDARAUTO()

* **Use Case**: To automatically generate a date table based on the data model.

**Year, Quarter, Month, Day**:

* **Syntax**:

Year = YEAR(Orders[OrderDate])  
Quarter = QUARTER(Orders[OrderDate])  
Month = MONTH(Orders[OrderDate])  
Day = DAY(Orders[OrderDate])

* **Use Case**: To extract individual date parts for further analysis.

Filter Functions

**ALL**:

* **Definition**: Removes all filters from the specified columns in the table.
* **Syntax**:

ALL(<table>[<column>])

* **Example**:

AllSales = CALCULATE(SUM(Orders[SalesAmount]), ALL(Orders))

* **Use Case**: To clear filters from specific columns or tables.

**ALLEXCEPT**:

* **Definition**: Removes all filters except for the specified columns.
* **Syntax**:

ALLEXCEPT(<table>, <column>[, <column> [, …]])

* **Example**:

AllSalesExceptDate = CALCULATE(SUM(Orders[SalesAmount]), ALLEXCEPT(Orders, Orders[OrderDate]))

* **Use Case**: To retain specific filters while clearing others.

**ALLSELECTED**:

* **Definition**: Removes filters from the outer context.
* **Syntax**:

ALLSELECTED(<table>[<column>])

* **Example**:

AllSelectedSales = CALCULATE(SUM(Orders[SalesAmount]), ALLSELECTED(Orders))

* **Use Case**: To retain user-applied filters but remove context filters.

**FILTER**:

* **Definition**: Returns a table that represents a subset of another table.
* **Syntax**:

FILTER(<table>, <expression>)

* **Example**:

FilteredOrders = FILTER(Orders, Orders[Quantity] > 10)

* **Use Case**: To create a filtered subset of a table.

**CALCULATETABLE**:

* **Definition**: Modifies a table expression by applying filters.
* **Syntax**:

CALCULATETABLE(<table>, <filters>)

* **Example**:

FilteredSalesTable = CALCULATETABLE(Orders, Orders[SalesAmount] > 1000)

* **Use Case**: To create a table with specific filters applied.

**RANKX**:

* **Definition**: Returns the ranking of a number in a list of numbers for each row in a table.
* **Syntax**:

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

* **Example**:

SalesRank = RANKX(ALL(Orders), Orders[SalesAmount])

* **Use Case**: To rank items based on a numeric value.

**SELECTEDVALUE**:

* **Definition**: Returns the value when the context for columnName has been filtered down to one distinct value.
* **Syntax**:

SELECTEDVALUE(<column>[, <alternateResult>])

* **Example**:

SelectedCustomer = SELECTEDVALUE(Customers[CustomerName])

* **Use Case**: To return a single selected value from a column.

**SUMMARIZE**:

* **Definition**: Returns a summary table for the requested totals over a set of groups.
* **Syntax**:

SUMMARIZE(<table>, <groupBy\_columnName>[, <groupBy\_columnName>]…, [<name>, <expression>]…)

* **Example**:

SummaryTable = SUMMARIZE(Orders, Orders[OrderDate], "TotalSales", SUM(Orders[SalesAmount]))

* **Use Case**: To create a summary table based on grouped data.

**TOPN**:

* **Definition**: Returns the top N rows of the specified table.
* **Syntax**:

TOPN(<n\_value>, <table>, <orderBy\_expression>[, <order>])

* **Example**:

Top10Sales = TOPN(10, Orders, Orders[SalesAmount], DESC)

* **Use Case**: To retrieve the top N items based on a specific criteria.

Most Versatile DAX Function: CALCULATE

**Definition**: CALCULATE evaluates an expression in a modified filter context.

**Syntax**:

CALCULATE(<expression>, <filter1>, <filter2>, …)

**Example**:

YTDSales = CALCULATE(SUM(Orders[SalesAmount]), DATESYTD('Date'[Date]))

**Use Case**: CALCULATE is used to change the context in which data is evaluated, allowing for complex and versatile calculations.

**Steps to Use**:

1. **Identify the base expression**: This is the calculation you want to perform.
2. **Apply filters**: Add the filters you want to modify the context.
3. **Evaluate**: Use CALCULATE to evaluate the expression with the modified context.

**When to Use**: Use CALCULATE when you need to perform calculations in a specific context or with specific filters applied.

**Why Use**: It provides flexibility in data analysis by allowing you to dynamically change the filter context of your calculations.

IF and SWITCH in DAX

**IF**:

* **Definition**: Performs a logical test and returns one value for TRUE and another for FALSE.
* **Syntax**:

IF(<logical\_test>, <value\_if\_true>, <value\_if\_false>)

* **Example**:

HighSales = IF(SUM(Orders[SalesAmount]) > 1000, "High", "Low")

* **Use Case**: To perform conditional calculations based on a logical test.

**SWITCH**:

* **Definition**: Evaluates an expression against a list of values and returns one of multiple possible result expressions.
* **Syntax**:

SWITCH(<expression>, <value1>, <result1>, <value2>, <result2>, …, <else>)

* **Example**:

SalesCategory = SWITCH(TRUE(), SUM(Orders[SalesAmount]) > 1000, "High", SUM(Orders[SalesAmount]) > 500, "Medium", "Low")

* **Use Case**: To perform multiple conditional evaluations and return different results based on the evaluation.

Assignment

Example 1: Sales Performance Analysis

**Business Question**: What are the total sales, average sales, and sales rank for each product category?

**DAX Formulas**:

TotalSales = SUM(Orders[SalesAmount])  
AverageSales = AVERAGE(Orders[SalesAmount])  
SalesRank = RANKX(ALL(Orders), Orders[SalesAmount])

Example 2: Customer Segmentation

**Business Question**: Segment customers based on their total purchase amount.

**DAX Formulas**:

CustomerSegment = SWITCH(  
    TRUE(),  
    [TotalPurchase] > 1000, "High Value",  
    [TotalPurchase] > 500, "Medium Value",  
    "Low Value"  
)

Example 3: Year-over-Year Sales Growth

**Business Question**: Calculate the Year-over-Year sales growth.

**DAX Formulas**:

PreviousYearSales = CALCULATE(SUM(Orders[SalesAmount]), SAMEPERIODLASTYEAR('Date'[Date]))  
YOYSalesGrowth = DIVIDE([TotalSales] - [PreviousYearSales], [PreviousYearSales])

By understanding and applying these DAX functions, you can create sophisticated calculations and analyses in Power BI, allowing for deeper insights and data-driven decision-making.

..

Data Visualization in Power BI

**Introduction to Data Visualizations** Data visualization is the graphical representation of information and data. By using visual elements like charts, graphs, and maps, data visualization tools provide an accessible way to see and understand trends, outliers, and patterns in data.

Charts in Power BI

Power BI offers a variety of chart types to represent data visually:

1. **Bar and Column Charts**: Used to compare values across categories.
2. **Line and Area Charts**: Ideal for showing trends over time.
3. **Pie and Donut Charts**: Useful for showing proportions of a whole.
4. **Scatter and Bubble Charts**: Good for showing relationships between two or more variables.
5. **Waterfall Charts**: Show the cumulative effect of sequential positive or negative values.
6. **Funnel Charts**: Used to represent stages in a process.

**Example**: To create a bar chart showing sales by product category:

1. Select the bar chart visual.
2. Drag the "Product Category" field to the Axis.
3. Drag the "Sales Amount" field to the Values.

Matrices and Tables

**Matrix**: Similar to a pivot table in Excel. It allows for multi-dimensional data representation with row and column headers.

**Table**: A simple grid that displays data in rows and columns. Ideal for showing detailed and granular data.

**Example**: To create a table showing detailed sales data:

1. Select the table visual.
2. Drag fields such as "Product Name", "Sales Amount", and "Order Date" into the Values area.

Map Visualizations

Maps in Power BI can visualize geographical data:

1. **Basic Map**: Plots data points on a map.
2. **Filled Map**: Colors geographical areas based on data values.
3. **ArcGIS Map**: Advanced map with additional features provided by Esri.

**Example**: To create a map showing sales by region:

1. Select the map visual.
2. Drag the "Region" field to the Location.
3. Drag the "Sales Amount" field to the Size.

Gauges and Cards

**Gauge**: Displays a single value within a range, ideal for KPIs.

**Card**: Shows a single data point, such as total sales or number of customers.

**Example**: To create a card showing total sales:

1. Select the card visual.
2. Drag the "Sales Amount" field to the Values.

KPI Visuals

KPI (Key Performance Indicator) visuals track progress toward a goal. They display a single value along with a target value and status indication (good, bad, on track).

**Example**: To create a KPI visual:

1. Select the KPI visual.
2. Drag the "Current Sales" field to Indicator.
3. Drag the "Target Sales" field to Target.

Color Formatting and Configuring Visualizations

Color formatting helps in distinguishing different data series and highlighting key insights. Power BI allows conditional formatting based on values, data bars, color scales, etc.

**Example**: To apply color formatting to a bar chart:

1. Select the bar chart.
2. Go to the "Format" pane.
3. Expand the "Data colors" section.
4. Set the color options based on your preferences.

Shapes, Text Boxes, and Images

Adding shapes, text boxes, and images can enhance reports by providing context, labels, and additional information.

**Example**: To add a text box:

1. Go to the "Insert" tab.
2. Select "Text box".
3. Type in the desired text and format it as needed.

Page Layout and Z-order

Page layout involves arranging visuals on a report page. Z-order controls the layering of visuals, determining which visuals appear in front or behind others.

**Example**: To adjust the Z-order of visuals:

1. Select a visual.
2. Use the "Bring Forward" or "Send Backward" options in the "Format" pane.

What Are Custom Visuals?

Custom visuals are additional visualization types created by the community or third-party developers. They can be imported into Power BI to extend the default visualization options.

**Example**: To import a custom visual:

1. Go to the "Visualizations" pane.
2. Click on the "..." (three dots) and select "Import from marketplace".
3. Search for the desired custom visual and add it to your report.

Assignment

1. **Create a Sales Dashboard**:
   * **Sales Overview**: Use a bar chart to show total sales by month.
   * **Sales by Region**: Use a map visual to display sales by region.
   * **Top Products**: Use a table to list top 10 products by sales.
   * **KPI**: Add a KPI visual to track the sales target.
2. **Analyze Customer Demographics**:
   * **Customer Age Distribution**: Use a histogram or bar chart.
   * **Income Level**: Use a pie chart to show customer distribution by income level.
3. **Product Performance**:
   * **Sales Trend**: Use a line chart to show sales trend over the last 12 months.
   * **Product Categories**: Use a column chart to compare sales across different product categories.
   * **Average Sales Price**: Use a card visual to display the average sales price.

By following these steps and examples, you can create comprehensive and insightful data visualizations in Power BI that help in making data-driven business decisions.

Introduction to Power BI Dashboard, Q&A, and Data Insights

**Power BI Dashboard**: A Power BI dashboard is a single-page, often called a canvas, that uses visualizations to tell a story. Because it is limited to one page, a well-designed dashboard contains only the most important elements of that story. Visualizations on a dashboard are called tiles, and they are pinned to the dashboard from reports.

Creating a Dashboard with Interactivity (Slicers and Filters)

**Slicers**: Slicers are visual filters in Power BI. They are used to segment and filter data dynamically based on user selections.

**Filters**: Filters in Power BI can be applied at different levels (visual, page, report) to refine data displayed in visuals.

**Example**: To create a dashboard with interactivity:

1. **Add Slicers**:
   * Go to the report view.
   * Select the slicer visual.
   * Drag a field (e.g., "Region") to the slicer to create a slicer for regions.
2. **Apply Filters**:
   * Go to the filter pane.
   * Drag a field (e.g., "Product Category") to the Filters on this page section.
   * Set filter conditions as needed.

Q & A

Power BI's Q&A feature allows users to ask questions about their data in natural language and get answers in the form of visuals.

**Example**: To use Q&A:

1. Add a Q&A visual to your report.
2. Type a question in the Q&A box, such as "Total sales by region".
3. Power BI will generate the visual based on the question.

Smart Narrative

Smart Narrative automatically generates text summaries that highlight insights about the data in your visuals.

**Example**: To add a Smart Narrative:

1. Select a visual on your report.
2. Go to the "Insert" tab and select "Smart Narrative".
3. Power BI will generate a text summary of the insights from the selected visual.

Decomposition Tree

The decomposition tree visual allows users to break down a measure into its contributing factors, providing insights into the root cause of a metric.

**Example**: To create a Decomposition Tree:

1. Select the decomposition tree visual.
2. Drag a measure (e.g., "Total Sales") to the Analyze field.
3. Drag categorical fields (e.g., "Region", "Product Category") to the Explain By field.
4. Click on the nodes to decompose the data further.

Key Influencers

The Key Influencers visual helps identify the factors that influence a specific metric. It analyzes the data and provides insights into what impacts the metric the most.

**Example**: To create a Key Influencers visual:

1. Select the Key Influencers visual.
2. Drag a field (e.g., "Sales Amount") to the Analyze field.
3. Drag other fields (e.g., "Region", "Product Category") to the Explain By field.
4. Power BI will analyze the data and show the key influencers.

Power BI Mobile

Power BI Mobile allows users to access their Power BI reports and dashboards on mobile devices, ensuring data insights are available on the go.

**Features**:

* Interactive dashboards and reports.
* Data alerts and notifications.
* QR code scanning for direct access to reports.
* Offline access to previously viewed reports.

**Example**: To use Power BI Mobile:

1. Download and install the Power BI app on your mobile device.
2. Sign in with your Power BI credentials.
3. Access and interact with your dashboards and reports on your mobile device.

Assignment

1. **Create an Interactive Sales Dashboard**:
   * **Sales Overview**: Use a bar chart to show total sales by month.
   * **Sales by Region**: Use a map visual to display sales by region.
   * **Top Products**: Use a table to list top 10 products by sales.
   * **Add Slicers**: Add slicers for "Region" and "Product Category".
   * **Apply Filters**: Apply page-level filters for a specific time period.
2. **Implement Q&A**:
   * Add a Q&A visual to your dashboard.
   * Ask questions like "Total sales by product category" and "Average sales by region".
3. **Generate Smart Narratives**:
   * Add Smart Narrative visuals to summarize key insights from your charts and tables.
4. **Use Decomposition Tree**:
   * Create a decomposition tree visual to analyze total sales by region and product category.
5. **Analyze Key Influencers**:
   * Use the Key Influencers visual to determine factors impacting total sales.
6. **Test on Power BI Mobile**:
   * Access your dashboard on the Power BI Mobile app.
   * Interact with the visuals and check for data alerts.

By completing these tasks, you will gain hands-on experience with the interactive features of Power BI and learn how to create insightful dashboards that can be accessed on various devices.

addCode

addText

Power BI Service

Power BI Service is the cloud-based platform that allows you to collaborate, share, and distribute your Power BI reports and dashboards. It provides a centralized space for managing all your Power BI content, enabling seamless access and sharing across your organization.

**Key Features**:

* Publish and share reports and dashboards.
* Create workspaces for collaboration.
* Schedule data refreshes.
* Manage user permissions.
* Analyze data in real-time with live dashboards.

Introduction to app.powerbi.com

**app.powerbi.com** is the web portal for Power BI Service. It is where you can create, share, and manage your Power BI reports and dashboards.

**Key Sections**:

* **Workspaces**: Collaborative environments where you can share and manage reports and dashboards with your team.
* **My Workspace**: Personal workspace for your own Power BI content.
* **Apps**: Packaged collections of related dashboards and reports.
* **Reports**: Pages of interactive data visualizations.
* **Dashboards**: Single-page summaries of multiple reports and visualizations.
* **Datasets**: Collections of data sources used in your reports and dashboards.

Schedule Refresh

Scheduling data refreshes ensures that your reports and dashboards are updated with the latest data at regular intervals.

**Steps to Schedule a Refresh**:

1. **Upload or Publish Your Report**:
   * Use Power BI Desktop to create your report.
   * Publish the report to the Power BI Service by clicking "Publish" in Power BI Desktop.
2. **Navigate to Dataset Settings**:
   * In the Power BI Service, go to the "Datasets" section in your workspace.
   * Find the dataset you want to refresh and click the ellipsis (three dots) next to it.
   * Select "Schedule refresh".
3. **Configure Refresh Settings**:
   * Enable "Keep your data up to date".
   * Set the frequency (daily, weekly) and time for the refresh.
   * Provide credentials for data sources if needed.
4. **Save and Apply Settings**:
   * Save the refresh schedule to ensure your dataset is updated automatically.

Data Flow and Using Power BI Online

**Data Flows**: Data flows in Power BI are used to extract, transform, and load data from various sources. They allow you to create reusable data transformations and make the data available across multiple reports and datasets.

**Creating Data Flows**:

1. **Go to a Workspace**:
   * In the Power BI Service, navigate to the workspace where you want to create the data flow.
2. **Create a Data Flow**:
   * Click on "New" and select "Dataflow".
   * Choose to define a new entity or import from a model.
3. **Define Data Source and Transformations**:
   * Connect to your data source (e.g., SQL Server, Excel).
   * Use Power Query Editor to transform and clean your data.
4. **Save and Refresh**:
   * Save the data flow and set a refresh schedule if needed.

**Using Power BI Online**:

* Create and edit reports directly in the Power BI Service.
* Share reports and dashboards with colleagues.
* Access shared content from anywhere using a web browser.

Download Data as Live in PowerPoint and More

**Download as Live Data in PowerPoint**: Power BI allows you to embed live Power BI reports into PowerPoint presentations, ensuring your data is always up-to-date.

**Steps**:

1. **Open Report**:
   * Open the report in Power BI Service that you want to embed in PowerPoint.
2. **Export to PowerPoint**:
   * Click on the "File" menu.
   * Select "Export to PowerPoint".
3. **Choose Live Data Option**:
   * Ensure the "Export with live data" option is selected.
   * Download the PowerPoint file.
4. **Embed in PowerPoint**:
   * Open the PowerPoint file.
   * The live Power BI report will be embedded as a slide.
   * Ensure you are connected to the internet to refresh the data.

Assignment

1. **Publish and Share Reports**:
   * Create a report in Power BI Desktop.
   * Publish the report to Power BI Service.
   * Share the report with colleagues in a workspace.
2. **Schedule Data Refresh**:
   * Set up a daily data refresh for your published dataset.
3. **Create a Data Flow**:
   * Create a data flow in Power BI Service.
   * Connect to a data source and apply transformations.
   * Use the data flow in a new report.
4. **Embed Live Data in PowerPoint**:
   * Export a live report to PowerPoint.
   * Present the live data report in a PowerPoint presentation.

By completing these tasks, you will gain a comprehensive understanding of how to use Power BI Service for managing, sharing, and refreshing your Power BI content, as well as how to integrate live data into presentations.

Power BI Direct Connectivity

Power BI offers multiple ways to connect to your data, each with its own advantages and use cases. The main connectivity modes include Import, DirectQuery, and Live Connection.

**Import Mode**:

* **Description**: Data is imported into Power BI and stored in its in-memory engine.
* **Use Case**: Best for scenarios where data doesn't need to be updated in real-time.
* **Performance**: Fast querying since data is cached.
* **Refresh**: Scheduled refresh required to update data.

**DirectQuery Mode**:

* **Description**: Queries are sent directly to the data source in real-time without importing data into Power BI.
* **Use Case**: Ideal for large datasets and when up-to-date data is crucial.
* **Performance**: Dependent on the underlying data source's performance.
* **Refresh**: No scheduled refresh needed; data is always up-to-date.

**Live Connection**:

* **Description**: Similar to DirectQuery but typically used with Analysis Services models.
* **Use Case**: For connecting to live data models like SQL Server Analysis Services (SSAS).
* **Performance**: Utilizes the processing power of the source system.
* **Refresh**: Data is always current as it queries the live model directly.

Data Gateways

Data Gateways act as bridges to securely transfer data between on-premises data sources and Power BI Service. They are essential for enabling DirectQuery and Live Connection modes for on-premises databases.

**Types of Data Gateways**:

1. **On-premises Data Gateway**:
   * Supports multiple users and data sources.
   * Suitable for enterprise-level scenarios.
2. **On-premises Data Gateway (Personal Mode)**:
   * Designed for single-user scenarios.
   * Suitable for personal use cases.

Connecting to On-Premises SQL Server Database Using Gateway

**Steps to Connect**:

1. **Install Data Gateway**:
   * Download the On-premises Data Gateway from the Power BI website.
   * Install and configure the gateway.
   * Sign in with your Power BI credentials.
2. **Add Data Source in Power BI Service**:
   * Go to the Power BI Service (app.powerbi.com).
   * Navigate to "Settings" > "Manage Gateways".
   * Add a new data source and select the gateway you configured.
   * Provide the necessary connection details for your SQL Server database.
3. **Connect in Power BI Desktop**:
   * Open Power BI Desktop.
   * Select "Get Data" and choose "SQL Server".
   * Enter the server and database details.
   * Choose "DirectQuery" as the connectivity mode.
   * Authenticate with the required credentials.
4. **Publish to Power BI Service**:
   * Create your report in Power BI Desktop using DirectQuery.
   * Publish the report to the Power BI Service.
   * The report will use the gateway to query the on-premises SQL Server in real-time.

Assignment

1. **Set Up Data Gateway**:
   * Download, install, and configure the On-premises Data Gateway.
   * Add an on-premises SQL Server database as a data source in the Power BI Service.
2. **Create Report with DirectQuery**:
   * Connect to the on-premises SQL Server using DirectQuery in Power BI Desktop.
   * Create a report showing real-time data (e.g., sales transactions).
3. **Publish and Test**:
   * Publish the report to the Power BI Service.
   * Verify that the report displays real-time data using the configured gateway.
4. **Compare Connectivity Modes**:
   * Create a simple report using Import mode.
   * Create a similar report using DirectQuery.
   * Analyze the performance differences and use cases for each mode.

By completing this assignment, you will gain practical experience with Power BI's connectivity options, particularly using Data Gateways to connect to on-premises data sources for real-time analytics

.

Total\_Sales = CALCULATE(sum(Orders[Sales]))

✅Current MTD = CALCULATE(TOTALMTD([Total\_Sales],DATESMTD(Orders[Order Date]))) ✅Current QTD = CALCULATE(TOTALQTD([Total\_Sales],DATESQTD(Orders[Order Date])))

✅Current YTD = CALCULATE(TOTALYTD([Total\_Sales],DATESYTD(Orders[Order Date]))) ✅Previous Month Total Sales = CALCULATE([Total\_Sales],PREVIOUSMONTH(DATESMTD(Orders[Order Date])))

✅Previous Quarter Total Sales = CALCULATE([Total\_Sales],PREVIOUSQUARTER(DATESQTD(Orders[Order Date])))

✅Previous Year Total Sales = CALCULATE([Total\_Sales],PREVIOUSYEAR(DATESYTD(Orders[Order Date])))

✅Previous MTD = ([Previous Month Total Sales] + [Current MTD])

✅Previous QTD = ([Previous Quarter Total Sales]+[Current QTD])

✅Previous YTD = ([Previous Year Total Sales]+[Current YTD])

✅SPLY MTD = CALCULATE([Current MTD],SAMEPERIODLASTYEAR(Orders[Order Date]))

✅SPLY QTD = CALCULATE([Current QTD],SAMEPERIODLASTYEAR(Orders[Order Date]))

✅SPLY YTD = CALCULATE([Current YTD],SAMEPERIODLASTYEAR(Orders[Order Date]))