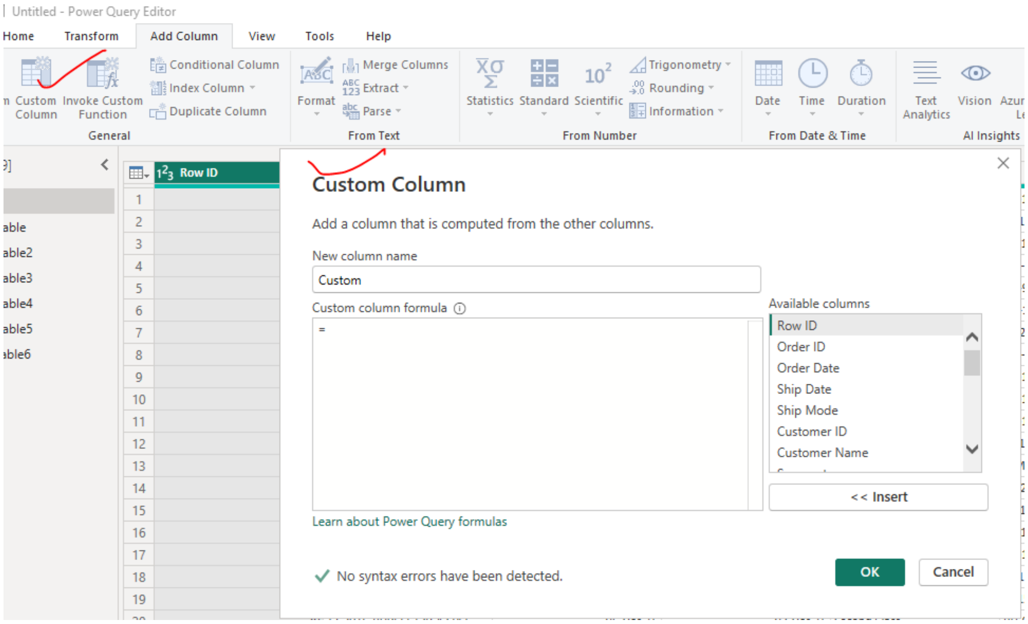
Chapter-21 Creating a Custom Column in Power BI

A **Custom Column** in Power BI allows you to create a new column using a custom formula or expression. This is typically done in the **Power Query Editor** using the **M language**, which allows for more advanced data transformation logic. Custom columns are useful when you need to perform calculations, apply transformations, or manipulate data that cannot be achieved using standard options like conditional columns.

Steps to Create a Custom Column in Power BI



Custom Column

Add a column that is computed from the other columns.

New column name

Custom

Custom column formula

=

Available columns

Row ID
Order ID
Order Date
Ship Date
Ship Mode
Customer ID
Customer Name

Learn about Power Query formulas

✓ No syntax errors have been detected.

Step 1: Open Power Query Editor

1. Open your Power BI Desktop file.
2. In the **Home** tab, click **Transform Data** to open the **Power Query Editor**.

Step 2: Select the Table

1. In the **Queries** pane on the left, select the table where you want to add the custom column.

Step 3: Add a Custom Column

1. In the **Power Query Editor**, go to the **Add Column** tab on the ribbon.
2. Click **Custom Column**. This opens the **Custom Column** dialog box.

Step 4: Define the Custom Column Formula

1. **Custom Column Name**: Give the new column a descriptive name.
2. **Custom Column Formula**: Enter your formula or expression. Power BI uses the **M language** for creating custom formulas.
   - For basic operations, you can use standard operators like +, -, *, /.
   - You can reference other columns by typing their names in the formula.
   - Power BI provides a rich set of built-in functions to transform data, such as

     Text.Upper(), Date.AddDays(), List.Sum(), etc.

   Example: If you want to create a custom column that multiplies the **Sales Amount** by 1.1 (for a 10% increase), your formula would look like:

   ```
   [Sales Amount] * 1.1
   ```

3. **Error Handling**: You can handle potential errors in your formula using functions like try...otherwise to prevent issues with invalid data.
4. **Use Built-in Functions**: Power BI offers many built-in functions such as:
   - Text: Text.Upper([Column Name]), Text.Length([Column Name])
   - Date: Date.AddDays([DateColumn], 10), Date.Year([DateColumn])
   - List: List.Sum([ListColumn]), List.Average([ListColumn])

Step 5: Apply the Formula

1. Once you've defined your formula, click **OK** to create the custom column. The column will appear in the data preview.

Step 6: Close and Apply Changes

1. After creating the custom column, go back to the **Home** tab in the Power Query Editor.
2. Click **Close & Apply** to apply the changes and return to the Power BI report view.

Example of Creating a Custom Column

Let's say you have a **Sales** table with **Quantity** and **Unit Price** columns. You want to create a new column called **Total Sales**, which is the product of **Quantity** and **Unit Price**.

1. **Select the Sales Table** in Power Query Editor.
2. Go to **Add Column** > **Custom Column**.

3. Name the column **Total Sales**.
4. In the **Custom Column Formula** box, enter:

```
[Quantity] * [Unit Price]
```

5. Click **OK**.

Now, you have a new **Total Sales** column that calculates the total sales for each row.

More Complex Example: Adding a Conditional Custom Column

You can also create custom columns that include conditional logic. For instance, you might want to create a **Discounted Sales** column, where you apply a 10% discount to sales greater than $500, and no discount otherwise.

1. **Select the Sales Table** in Power Query Editor.
2. Go to **Add Column** > **Custom Column**.
3. Name the column **Discounted Sales**.
4. In the **Custom Column Formula** box, enter:

```
if [Total Sales] > 500 then [Total Sales] * 0.9 else [Total Sales]
```

5. Click **OK**.

This formula checks if **Total Sales** is greater than 500 and applies a 10% discount if the condition is met.

Using M Functions in Custom Columns

You can utilize various built-in **M functions** to manipulate and transform data more efficiently.

1. Text Functions:

- Text.Upper([Column Name]): Converts text to uppercase.
- Text.Lower([Column Name]): Converts text to lowercase.
- Text.Trim([Column Name]): Removes leading and trailing spaces.
- Text.Contains([Column Name], "value"): Checks if the text contains a specific substring.

Example:

```
Text.Upper([Product Name])
```

2. Date Functions:

- `Date.AddDays([Date Column], 10)`: Adds 10 days to a date.
- `Date.Year([Date Column])`: Extracts the year from a date.
- `Date.Month([Date Column])`: Extracts the month from a date.

Example:

```
Date.Year([Order Date])
```

3. Numeric Functions:

- `Number.Round([Column Name], 2)`: Rounds a number to 2 decimal places.
- `Number.Abs([Column Name])`: Converts negative numbers to positive.

Example:

```
Number.Round([Sales Amount], 2)
```

Benefits of Custom Columns

1. **Flexibility**: Custom columns give you full flexibility to perform any calculation, transformation, or manipulation that isn't possible with simple operations or conditional columns.
2. **Advanced Data Transformation**: You can apply complex formulas or logic, including conditional statements, to create custom metrics or categorizations.
3. **Optimized Reports**: Custom columns enable more accurate data transformations before loading it into Power BI, ensuring that your reports use the right data for visualization.

Best Practices for Creating Custom Columns

1. **Keep Formulas Simple**: While you can write complex formulas, avoid overly complicated expressions. If the logic is too complex, consider breaking it into multiple columns.
2. **Avoid Overuse of Custom Columns**: Custom columns can slow down your data model if overused. Whenever possible, leverage existing Power BI features (like calculated measures) or keep the number of custom columns low.
3. **Document Your Logic**: If you're applying complex formulas or logic, consider adding comments (using //) to your M code to explain the reasoning behind the calculations.

4. **Optimize Performance**: Be mindful of performance when creating custom columns. Large datasets with many custom columns or complex formulas can impact the performance of your report.