Power BI Data Transformation Order Date Parsing Documentation

Prepared by:

Mostafa Ayman Reda

Mostafa Khaled Farouk

Mariam Atef Gamal Eldin

Sandra Hassem Sobhy

Project: Superstore Sales Data Cleaning and Modeling

Tool: Power BI Desktop

# Introduction

This document outlines the step by step approach used to clean and correctly parse the Order Date column in the Superstore Sales dataset imported into Power BI Desktop. The primary objective was to ensure that Power BI could correctly interpret the Order Date column for time-based analysis and reporting.

# Step 1: Importing the Excel Dataset

The source file was an Excel spreadsheet titled 'Superstore\_Sales\_mostafa Table.xlsx'. It was imported into Power BI using the 'Get Data > Excel' option. The dataset included 18 columns such as Customer Name, Segment, Country, Product ID, Category, Product Name, Sales, and importantly, Order Date.

# Step 2: Issue Encountered with Order Date

Upon importing the data, Power BI attempted to automatically detect the data types. When trying to convert the Order Date column to type 'Date', Power BI threw an error:  
DataFormat.Error: We couldn't parse the input provided as a Date value. Details: 15/04/2018  
  
This happened because Power BI's default locale interprets dates in the MM/DD/YYYY format, whereas the dataset uses the DD/MM/YYYY format.

# Step 3: Manual Parsing Solution in Power Query

To resolve the issue, the following steps were applied:

## 3.1 Convert Order Date to Text

The Order Date column was first converted to Text. This ensured that Power BI would not attempt to parse it prematurely, allowing for full manual control over the parsing process.

## 3.2 Add Custom Column to Parse Dates

A new custom column was added using Power Query (Add Column > Custom Column) with the following M code:

try   
 let   
 cleanText = Text.Trim([Order Date]),  
 parts = Text.Split(cleanText, "/")  
 in   
 if List.Count(parts) = 3 then  
 #date(Number.FromText(parts{2}), Number.FromText(parts{1}), Number.FromText(parts{0}))  
 else null  
otherwise null

This formula performs the following:  
- Trims leading and trailing spaces.  
- Splits the date string by '/' to isolate day, month, and year.  
- Validates that there are exactly three parts.  
- Rearranges them into YYYY-MM-DD format and creates a valid date.  
- Uses try/otherwise to handle errors and replace them with null.

## 3.3 Change the New Column Type to Date

After the custom column was added, its data type was explicitly set to 'Date' using the Power Query interface. The transformation now successfully parsed valid dates while leaving invalid or empty values as nulls.

# Step 4: Final Clean-up and Applying Changes

The original Order Date column can optionally be removed or renamed. Once satisfied, the changes were applied by clicking 'Close & Apply' in Power Query to load the cleaned data into the Power BI model.

# Step 5: Create a Date Table

To support advanced time intelligence functions like Year-to-Date or Month-over-Month calculations, a Date Table can be created using the following DAX formula:

DateTable = CALENDAR(MIN('Sales'[Order Date]), MAX('Sales'[Order Date]))

You can relate this Date Table to the 'Order Date' field in your model and use it in slicers or time-based visuals.

# Conclusion

By manually parsing the Order Date column and applying data cleaning in Power Query, the dataset is now ready for accurate time-based analysis. This ensures robust and error-free visualizations in Power BI.