

1. What is the purpose of the "Applied Steps" pane in Power Query?

It shows the sequence of transformations applied to your data (like filters, renames, merges). Each step is recorded in order and can be edited, removed, or reordered.

2. How do you remove duplicate rows in Power Query?

1. Using the Interface

- Select the column(s) you want to check for duplicates.
- Go to the **Home** tab → click **Remove Rows** → choose **Remove Duplicates**.
 - If you select one column → duplicates are removed based on that column only.
 - If you select multiple columns → duplicates are removed only if *all* selected columns match.
 - If you select no columns → duplicates are removed based on the *entire row*.

◆ 2. Using M code

```
#"Removed Duplicates" = Table.Distinct("#PreviousStepName")
```

👉 Example (removes duplicates based on the entire row):

```
#"Removed Duplicates" = Table.Distinct("#Changed Type")
```

👉 Example (remove duplicates based only on CustomerID):

```
#"Removed Duplicates" = Table.Distinct("#Changed Type", {"CustomerID"})
```

3. What does the "Filter" icon do in Power Query?

In **Power Query**, the **Filter icon** (the little dropdown arrow next to each column header) lets you **filter rows** based on the values in that column.

◆ What you can do with the Filter icon:

- **Select/Deselect values** → keep or remove specific values.
- **Number filters** → e.g., greater than, less than, equals.
- **Date/Time filters** → e.g., before, after, between, year, month, day.
- **Text filters** → e.g., begins with, contains, does not contain.
- **Remove null/blank values.**

4. How would you rename a column from "CustID" to "CustomerID"?

1. In **Power Query (Power BI)**, you can rename a column with an M code step like this:

```
#"Renamed Columns" = Table.RenameColumns("#PreviousStepName",  
{{"CustID", "CustomerID"}})
```

👉 Replace "#PreviousStepName" with the step just before the renaming (for example, "#Changed Type").

So if your last step was "#Changed Type", the code would be:




```
#"Renamed Columns" = Table.RenameColumns("#Changed Type", {"CustID",  
"CustomerID"}) or
```

2. Using the interface

- Right-click on the column header **CustID** → select **Rename** → type **CustomerID**.

5. What happens if you click "Close & Apply" in Power Query?

In **Power BI**, when you click **Close & Apply** in Power Query:

1.  **All the transformations** (filters, renames, merges, etc.) you applied in Power Query are saved.
2.  The transformed data is **loaded into the Power BI data model** (in memory).
3.  You return from the Power Query Editor back to the main **Power BI Desktop** window, where you can build visuals, measures, and reports using the cleaned data.

6. Remove all rows where Quantity is less than 2.

In **Power Query (Power BI)** you can remove rows where $\text{Quantity} < 2$ by applying a filter.

 **M code:**

```
#"Filtered Rows" = Table.SelectRows("#PreviousStepName", each [Quantity] >= 2)
```

👉 Replace `"PreviousStepName"` with the name of the step before filtering (for example, `"Changed Type"`).

So if your last step was `"Changed Type"`, the code would be:

```
#"Filtered Rows" = Table.SelectRows("#Changed Type", each [Quantity] >= 2)
```

OR: Using the filter icon

1. Click the **filter icon** in the **Quantity** column header.
2. Choose **Number Filters** → **Greater Than or Equal To...**
3. Enter **2**.
4. Click **OK**.

7. Split the OrderDate column into separate "Year," "Month," and "Day" columns.

In **Power Query (Power BI)**, you can split a date column (like **OrderDate**) into **Year**, **Month**, and **Day** in two ways:

◆ 1. Using the Interface

1. Select the **OrderDate** column.
2. Go to the **Add Column** tab → **Date** menu.
 - Choose **Year** → **Year**
 - Choose **Month** → **Month**
 - Choose **Day** → **Day**

3. This will create three new columns: Year, Month, and Day.

◆ **2. Using M code**

If your last step was #"Changed Type", you can add:


```
#"Inserted Year" = Table.AddColumn(#"Changed Type", "Year", each  
Date.Year([OrderDate]), Int64.Type),  
#"Inserted Month" = Table.AddColumn(#"Inserted Year", "Month", each  
Date.Month([OrderDate]), Int64.Type),  
#"Inserted Day" = Table.AddColumn(#"Inserted Month", "Day", each  
Date.Day([OrderDate]), Int64.Type)
```

8. Replace all "Mouse" entries in the Product column with "Computer Mouse."

In Power Query you can do it directly from the **Home → Transform → Replace Values** menu, or write the M-code yourself.

Here's the clean **M-code** snippet:

```
Table.ReplaceValue(  
    Source,  
    "Mouse",  
    "Computer Mouse",  
    Replacer.ReplaceText,  
    {"Product"}  
)
```

This will change every "Mouse" entry in the **Product** column to "Computer Mouse". 

9. Sort the table by OrderDate (newest first).

Using the interface

- Click the dropdown on **OrderDate** column header.
- Choose **Sort Descending (Z → A / Newest to Oldest)**.

10. How would you handle null values in the Price column?

In **Power Query (Power BI)** there are several ways to handle **null values** in a column like Price, depending on what you need to do with them:

◆ **1. Remove rows with null Price**

If you don't want rows where Price is blank:

```
#"Removed Nulls" = Table.SelectRows("#PreviousStepName", each [Price] <> null)
```

◆ 2. Replace nulls with a default value

For example, replace null with 0:

```
#"Replaced Nulls" = Table.ReplaceValue("#PreviousStepName", null, 0, Replacer.ReplaceValue, {"Price"})
```

◆ 3. Fill nulls using nearby values

- **Fill Down** (copy the value from the row above):

```
#"Filled Down" = Table.FillDown("#PreviousStepName", {"Price"})
```

- **Fill Up** (copy from the row below):

```
#"Filled Up" = Table.FillUp("#PreviousStepName", {"Price"})
```

11. Write custom M-code to add a column calculating TotalSpent = Quantity * Price.

You can add a calculated column in **Power Query (M code)** for TotalSpent = Quantity * Price using Table.AddColumn.

Here's the M code:

```
#"Added Custom" = Table.AddColumn("#PreviousStepName", "TotalSpent", each [Quantity] * [Price], type number)
```

👉 Replace "#PreviousStepName" with the step before you want to add the column (for example, "#Changed Type" or "#Filtered Rows").

✅ Example if the last step was "#Changed Type":

```
#"Added Custom" = Table.AddColumn(#"Changed Type", "TotalSpent", each  
[Quantity] * [Price], type number)
```

This will create a new column **TotalSpent** with the multiplication result.

12. Group the table by CustID to show total spending per customer.

To group your table by **CustID** and calculate **total spending per customer**, you first need a column like $\text{TotalSpent} = \text{Quantity} * \text{Price}$.

Step 1 – Add TotalSpent column

```
Table.AddColumn(Source, "TotalSpent", each [Quantity] * [Price], type number)
```

Step 2 – Group by CustID

```
Table.Group(  
    #"Added TotalSpent",  
    {"CustID"},  
    {"TotalSpending", each List.Sum([TotalSpent]), type number})  
)
```

Or:

🔍 Go to the **Home** tab → click **Group By**.

🔍 In the **Group By** window:

- Group by: **CustID**.
- Operation: **Sum**.
- Column: **TotalSpent**.
- New column name: **TotalSpending**.

🔍 Click **OK**.

13. Fix inconsistent date formats (e.g., 01/10/2023 vs. 2023-01-10) in OrderDate.

You can fix inconsistent date formats in **Power Query** without writing M-code by forcing the column into a single **Date type**. Here's how:

1. **Select the OrderDate column.**
2. On the ribbon, go to **Transform → Data Type → Date**.
 - If prompted, choose **Using Locale**.
 - Pick **Date** as the data type, then select the correct locale (for example, *English (United States)* or *English (United Kingdom)*) depending on whether your data is in MM/DD/YYYY or DD/MM/YYYY).
3. Power Query will automatically convert all values to a **standard internal date format**.
4. Once converted, the preview shows dates consistently (e.g., 2023-01-10).

💡 In Power Query, dates are always stored in one universal format; the difference you see (01/10/2023 vs. 2023-01-10) is just **display formatting**. After converting to **Date**, Power BI will treat them consistently, and you can later control how they display in your report.

14. Create a conditional column: Label orders as "High Value" if Price > 100.

Here's how you can do that in **Power Query** without M-code:

1. In Power Query Editor, go to the **Add Column** tab.
2. Click **Conditional Column**.
3. In the dialog:
 - Column Name → Price
 - Operator → is greater than
 - Value → 100
 - Output → "High Value"
4. Add an **Else** clause → "Normal" (or leave it blank if you don't want a label).
5. Name the new column → ValueLabel.
6. Click **OK**.

✅ Now each row will show "High Value" if Price > 100, otherwise "Normal".

15. Optimize the query to reduce refresh time (e.g., remove unused columns early).

In **Power Query**, optimization is mostly about doing the “heavy” steps early and letting the query engine (like SQL Server, Excel, etc.) handle the work efficiently. Here are the key ways to **reduce refresh time**:

◆ 1. Remove unused columns early

- Use **Home** → **Remove Columns** right after importing.
 - The fewer columns you carry through, the faster each later step runs.
-

◆ 2. Filter rows as early as possible

- If you only need 2023 data, apply the filter before merges or calculations.
 - This reduces the number of rows being processed later.
-

◆ 3. Use query folding

- When connecting to databases, let Power Query push transformations back to the source (SQL, etc.).
 - You'll know folding is happening if you see "**View Native Query**" in the step options.
-

◆ 4. Disable loading of intermediate queries

- In **Queries pane**, right-click helper queries → **Disable Load**.
 - Keeps your model smaller and refresh faster.
-

◆ 5. Combine steps where possible

- Instead of many small transformations, group them (e.g., renaming multiple columns in one step).
 - Each step creates overhead, so fewer steps = better performance.
-

◆ 6. Avoid row-by-row operations if possible

- Aggregations (like **Group By**) are faster than custom column loops.

