Lesson-3

1. What is the purpose of the "Applied Steps" pane in Power Query?
   Answer
   If you filter a table and then rename a column, each action appears as a separate step (e.g., "Removed Rows," "Renamed Columns"). Clicking a step shows the intermediate result, making debugging easier.

   This feature is crucial for maintaining control over your ETL (Extract, Transform, Load) process in Power BI, Excel, or other Power Query-integrated tools.

2. How do you remove duplicate rows in Power Query?
   Answer
   Select Columns to Check for Duplicates
   If you want to remove duplicates based on all columns, you don't need to select anything.
   If you want to remove duplicates based on specific columns, hold Ctrl and click each column header.
   Remove Duplicates
   Go to the Home tab.
   Click Remove Rows → Remove Duplicates

3. What does the "Filter" icon do in Power Query?
   Answer:
   The **"Filter" icon** in Power Query (found in the column header dropdown menu or the **Home/Transform** ribbon) allows you to **filter rows** based on specific conditions in your dataset. Here's what it does:

   ### **Key Functions of the Filter Icon:**
   1. **Basic Text/Numeric Filters**
      - Filter rows by values (e.g., "Equals," "Does Not Equal," "Contains," "Starts With").
      - For numbers: "Greater Than," "Less Than," "Between," etc.

   2. **Date Filters** (if the column contains dates)
      - Options like "Before," "After," "In the Previous Month," etc.

   3. **Dynamic vs. Static Filtering**
      - **Manual Selection**: Check/uncheck specific values to keep/exclude them.
      - **Custom Conditions**: Use logical rules (e.g., "Column A > 100").

   4. **Clearing Filters**
      - Remove filters to revert to the original dataset.

   ### **How It Works:**
   - Clicking the **▼ filter icon** in a column header opens a dropdown with filtering options.

- Selecting a filter adds a step in the **"Applied Steps"** pane (e.g., "Filtered Rows").
- The filter is applied to the entire query, reducing rows early in the transformation process (improving efficiency).
Example Use Cases:
- Remove blank/null values.
- Keep only sales data from a specific region.
- Exclude outliers (e.g., orders with negative quantities).

Note:
Filters in Power Query are **applied during data load**, unlike Excel table filters, which are interactive. Changes require refreshing the query.

Would you like a step-by-step guide on applying a specific filter type?

4. How would you rename a column from "CustID" to "CustomerID"?
   Answer:
   Right-Click Rename (Quickest)
   Open Power Query Editor (in Power BI, Excel, etc.).
   Locate the "CustID" column in the data preview.
   Right-click the column header and select "Rename".
   Type the new name ("CustomerID") and press Enter.
5. What happens if you click "Close & Apply" in Power Query?
   Answer:
   When you click "Close & Apply" in Power Query (in Power BI, Excel, or other supported tools), it performs the following actions:

   1. Closes the Power Query Editor
   Exits the query editing interface and returns you to the main application (e.g., Power BI Desktop, Excel).

   2. Applies All Transformations
   Executes the entire sequence of steps listed in the "Applied Steps" pane.

   The data is processed according to your transformations (e.g., filters, merges, column renames).

   3. Loads the Final Dataset
   In Power BI:

   The transformed data is loaded into the Data Model (unless you changed settings to "DirectQuery" or "Live Connection").

   You can now use the data in reports, visuals, or DAX calculations.

   In Excel:

The data is loaded into a table, PivotTable, or worksheet (depending on your selection when creating the query).
4. Refreshes Dependencies (If Any)
If other queries or reports depend on this data, they may also refresh (unless refresh is deferred).
Key Implications:
Irreversible Changes: Once applied, transformations become permanent unless you reopen Power Query and modify the steps.
Performance Impact: Complex queries may take time to process, especially with large datasets.
Data Model Update: In Power BI, the updated data becomes part of the .pbix file.
Difference Between "Close & Apply" vs. "Apply" (Power BI Only)
"Close & Apply": Saves changes, closes the editor, and loads data.
"Apply" (Power BI only): Applies changes without closing the editor (useful for checking intermediate results).
Pro Tip:
Use "Close & Apply" only after verifying your steps—or use "Keep Preview" (Power BI) to test before finalizing.

6. Remove all rows where Quantity is less than 2.
Answer:
Here's how to **remove all rows where "Quantity" is less than 2** in Power Query:

Method 1: Using the Filter Icon (Simplest)
1. **Open Power Query Editor** (in Power BI, Excel, etc.).
2. **Click the filter icon (▼)** in the **"Quantity"** column header.
3. Select **"Number Filters"** → **"Greater Than Or Equal To"**.
4. Enter **`2`** in the dialog box and click **OK**.
5. **"Close & Apply"** to save changes.

Method 2: Using the "Remove Rows" Option
1. Go to the **"Home"** tab.
2. Click **"Remove Rows"** → **"Remove Rows Based on Condition"**.
3. Choose the **"Quantity"** column.
4. Set the condition:
   - **"Value"** → **"is greater than or equal to"** → **`2`**.
5. Click **OK**.

Method 3: Manual M Code (Advanced)
1. Open the **Advanced Editor**.
2. Locate the step where the table is defined and add:
   ```powerquery
   = Table.SelectRows(PreviousStep, each [Quantity] >= 2)
7. Split the OrderDate column into separate "Year," "Month," and "Day" columns.
Answer:
Using the GUI (Recommended for Beginners)
   ```

Open Power Query Editor
Select the query containing your OrderDate column.
Select the OrderDate Column
Click on the column header.
Split by Date Parts
Go to the "Transform" tab.
Click "Date" → "Year" → "Year".
(This creates a new "Year" column.)
Repeat for "Month" and "Day":
"Date" → "Month" → "Month"
"Date" → "Day" → "Day"
Rename Columns (Optional)
Right-click each new column and rename them to Year, Month, Day.
"Close & Apply" to save.

8. Replace all "Mouse" entries in the Product column with "Computer Mouse."
   Answer:
   Here's how to replace all **"Mouse"** entries in the **Product** column with **"Computer Mouse"** in Power Query:

   ### **Method 1: Replace Values (GUI - Easiest)**
   1. **Open Power Query Editor**
      - Select your query.

   2. **Select the "Product" Column**
      - Click the column header.

   3. **Replace Values**
      - Go to the **"Transform"** tab.
      - Click **"Replace Values"**.
      - In the dialog box:
        - **"Value To Find"**: Enter `Mouse`
        - **"Replace With"**: Enter `Computer Mouse`
      - Click **OK**.

   4. **"Close & Apply"** to save.
   ### **Method 2: Conditional Column (If Partial Matches Exist)**
   1. **Add a Custom Column**
      - Go to **"Add Column"** → **"Conditional Column"**.
      - Set:
        - **New column name**: `Updated Product`
        - **Column name**: `Product`
        - **Operator**: "contains"
        - **Value**: `Mouse`

- **Output**: `Computer Mouse`
- **Else**: `[Product]` (keeps original value if no match).

2. **Remove Original Column (Optional)**
- Right-click the **"Product"** column → **"Remove"**.
- Rename the new column to **"Product"**.

Method 3: Using M Code (Advanced)

1. Open the **Advanced Editor** and modify the code:

```powerquery
= Table.ReplaceValue(
    PreviousStep,
    each [Product],
    each if Text.Contains([Product], "Mouse") then "Computer Mouse" else [Product],
    Replacer.ReplaceText,
    {"Product"}
)
```

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

Answer:

Using the Sort Button (Quickest)

Open Power Query Editor

Select your query.

Select the OrderDate Column

Click the column header.

Sort Descending

Go to the "Home" or "Transform" tab.

Click the "Sort Descending" button (↓A↓ icon).

(Alternatively, click the filter icon (▼) in the column header and select "Sort Descending".)

"Close & Apply" to save changes.

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

Answer:

Remove Rows with Null Prices

Use Case: If null prices are invalid and should be excluded.

Steps:

Select the Price column.

Click the filter icon (▼) in the column header.

Uncheck "(null)" (or "(blank)") → Click OK.

Result: Rows with null prices are deleted.
(Appears as "Filtered Rows" in Applied Steps.)

M Code Alternative:

powerquery
= Table.SelectRows(PreviousStep, each [Price] <> null)

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

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

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

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

```
 #”Total SpentCustomer”     = Table.Group(
  Source,
  {"CustID"},
  {
    "TotalSpent",
    each List.Sum([Quantity] * [Price]),  // Calculates sum of (Qty*Price) directly
    type number
  }
)
```

How to Implement

Open Advanced Editor in Power Query

Add this as a new step after your data is loaded and cleaned

Replace "Source" with your actual previous step name

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

```
#"Changed DateTime"= Table.TransformColumnTypes(
   PreviousStep,
   {{"OrderDate", type date}}  // Power Query will attempt automatic conversion
)
```

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

Using the GUI (Recommended for Beginners)
Open Power Query Editor

Select your table.

Add Conditional Column

Go to the "Add Column" tab → Click "Conditional Column".

Set Up the Rule

New column name: OrderValue

Column name: Price

Operator: is greater than

Value: 100

Output: "High Value"

Else: "Standard" (or leave blank for null)

Click OK → "Close & Apply".

15. Optimize the query to reduce refresh time (e.g., remove unused columns early).
Answer:
Here's a **step-by-step optimization guide** to reduce Power Query refresh time by removing unused columns early and applying other best practices:

---

1. Remove Unused Columns First
Why: Fewer columns = less memory usage and faster processing.
How:
```powerquery
= Table.SelectColumns(
   Source,
   {"CustID", "OrderDate", "Product", "Quantity", "Price"}  // Keep only needed columns
)
```
Pro Tip:
- Do this **immediately after `Source`** to minimize downstream processing.

---

2. Filter Rows Early

Why: Fewer rows = faster transformations.
Example (remove test/dummy records):
```powerquery
= Table.SelectRows(PreviousStep, each [CustID] <> "TEST" and [Quantity] > 0)
```

---

3. Optimize Data Types
Why: Correct types (e.g., `Int64` instead of `text` for IDs) reduce memory usage.
How:
```powerquery
= Table.TransformColumnTypes(
   PreviousStep,
   {
      {"CustID", Int64.Type},
      {"OrderDate", type date},
      {"Price", Currency.Type}  // Uses less memory than "decimal"
   }
)
```

---

4. Disable Automatic Type Detection
Why: Prevents Power Query from scanning data twice.
How:
- Go to **File → Options → Data Load → Power Query Editor**
- Uncheck **“Automatically detect column types and headers”**.

---

5. Replace Heavy Operations
Avoid:
- `Table.Group` with dynamic aggregations (use `List.Sum` instead).
- Nested `if` conditions (simplify logic).

Better:
```powerquery
// Instead of multiple conditional columns:
= Table.AddColumn(
   PreviousStep,
   "DiscountCategory",
   each
      if [Price] > 200 then "Premium"
      else if [Price] > 100 then "Standard"
```

```
      else "Basic",
   type text
)
```

---

6. Use Query Folding
Why: Pushes operations to the data source (SQL, etc.) instead of local processing.
Check: Right-click a step → **“View Native Query”** (if grayed out, folding isn’t working).
Enable:
- Use **SQL-friendly functions** (e.g., `Table.SelectRows` over `Table.AddColumn` with custom logic).
- Avoid `Table.Buffer`, which breaks folding.

---

7. Merge Steps Where Possible
Example: Combine column renames:
```powerquery
= Table.RenameColumns(
   PreviousStep,
   {
      {"CustID", "CustomerID"},
      {"Prod", "Product"}  // Multiple renames in one step
   }
)
```

---

8. Disable Privacy Firewall Warnings
Why: Prevents unnecessary checks across data sources.
How:
- Go to **File → Options → Privacy → Ignore Privacy Levels**.

---

9. Benchmark with “Query Diagnostics”
How:
1. Go to **Tools → Query Diagnostics → Start Diagnostics**.
2. Refresh the query.
3. Analyze **“Duration”** and **“CPU Usage”** per step.

---

10. Final Cleanup
```powerquery
// Remove intermediate columns after use:
= Table.RemoveColumns(
    PreviousStep,
    {"TempColumn1", "TempColumn2"}  // Columns used for calculations but no longer needed
)
```

Result:
- **Faster refresh times** (often 30–50% improvement).
- Lower memory usage.
- More maintainable code.

Key Check:
After optimizing, verify results with **“View Native Query”** (for folded steps) and **Query Diagnostics**.

Need help with a **specific query bottleneck**? Share your code, and I’ll analyze it!