Lesson-4

1. What is the difference between "Merge" and "Append" in Power Query?

   1. Merge (Join Tables)
   Purpose: Combines columns from two tables based on a matching key (like a SQL JOIN).

   Use Case: When you need to look up or add columns from one table to another (e.g., adding customer names to an orders table using a shared CustomerID).

   How It Works:
   Select a key column (e.g., ID) in both tables.
   Choose a join type (e.g., Inner, Left Outer, Full Outer) to define how unmatched rows are handled.
   The result is a single table with columns from both tables.
   2. Append (Stack Tables)
   Purpose: Combines rows from two or more tables with identical columns (like a SQL UNION).

   Use Case: When you need to stack data vertically (e.g., combining monthly sales tables into one).

   How It Works:

   Tables must have the same column names/structure.
   Rows from the second table are added below the first.
   The result is a single table with all rows but no new columns.
   Example:
   Table1: January Sales (Product, Revenue)
   Table2: February Sales (Product, Revenue)
   Append Result: A single table with all rows from January and February.
   Types:
   Union: Combines all rows (default).
   Ignore/Include: Handles mismatched columns (optional in Power Query).

2. How do you split a "Full Name" column into "First Name" and "Last Name"?
   Using the GUI (Recommended for Beginners)
   Open Power Query Editor

   Select your table in Power BI/Excel → Click "Transform Data" to open the editor.

   Select the "Full Name" Column

   Click the column header.

   Split by Delimiter

Go to the "Transform" tab → Click "Split Column" → "By Delimiter".

In the dialog box:

Delimiter: Select Space (" ").

Split At: Choose "Each occurrence of the delimiter" (or "Left-most delimiter" if names have middle names).

Advanced Options: Select "Columns" and enter 2 (for First/Last Name).

Rename Columns

Right-click the new columns → "Rename" to First Name and Last Name.

Handle Edge Cases (Optional)

If some names have middle initials (e.g., "John D. Doe"), use "Split into Rows" or manually adjust.

Click "Close & Apply" to save.

3. In Power Query, "Pivot Columns" is a transformation that reshapes your data by converting unique values from one column into new columns, while aggregating values from another column. It's similar to creating a pivot table in Excel but is part of the data preparation process.
   When to Use Pivot Columns
   Use this feature when you need to:
   Summarize data (e.g., sales per product by month).
   Convert long data to wide format (e.g., rows of monthly sales → columns for each month).
   Aggregate metrics (e.g., sum, average, count) across categories.

   Open Power Query Editor:

   Select your table → Click "Transform Data" (Power BI/Excel).

   Select the Column to Pivot:

   Highlight the column whose values will become new headers (Month in this case).

   Click "Pivot Column":

   Go to the "Transform" tab → Click "Pivot Column".

   Configure the Pivot:

   Values Column: Select the column to aggregate (Sales).

   Aggregation Function: Choose Sum, Average, Count, etc. (Default: Sum).

Click OK

4. How do you undo a step in Power Query?
   Delete a Single Step
   Open Power Query Editor

   In Power BI/Excel, go to "Transform Data" → "Transform Data" (Power Query Editor).

   Locate the "Applied Steps" Pane

   On the right side of the editor, you'll see a list of all transformations (e.g., "Removed Columns," "Renamed Columns").

   Delete the Step

   Click the "X" next to the step you want to undo.

   All subsequent steps will recalculate automatically.

   ⚠️ Warning: Deleting a step also removes all steps after it.

5. What is the purpose of "Reference" vs. "Duplicate" in queries?
   1. Reference
   Purpose: Creates a new query that points to the output of an existing query. Changes to the original query will propagate to the referenced query.
   Use Case:

   When you want to build upon an existing query without repeating transformations.

   To create multiple derivative datasets (e.g., one query filters data, another aggregates it).

   How It Works:

   Right-click the original query → Select "Reference".

   The new query starts with the final output of the original query.

   Add new steps (e.g., filters, calculations) to the referenced query.

   Key Features:

   Linked Dependency: If the original query changes, the referenced query updates automatically.

   Efficient: Avoids reprocessing the same steps twice (saves memory/performance).

No Isolation: Errors in the original query will break the referenced query.

Example:

Original Query: Cleans sales data (removes nulls, renames columns).

Referenced Query: Aggregates the cleaned data by region.

2. Duplicate
Purpose: Creates a fully independent copy of the original query, including all its steps. Changes to the original query do not affect the duplicate.
Use Case:

When you need to experiment with transformations without affecting the original.

To reuse logic but diverg later (e.g., same source, different filters).

How It Works:

Right-click the original query → Select "Duplicate".

The new query is an exact clone of the original, including all steps.

Modify the duplicate freely (add/delete steps).

Key Features:

Isolated: Changes to the original or duplicate do not affect each other.

Redundant Processing: Both queries run their steps separately (may impact performance).

Flexibility: Safe for testing or branching logic.

Example:

Original Query: Imports and cleans customer data.

Duplicate Query: Modifies the cleaning logic for a different report.

6. Merge Orders.csv and Customers.xlsx on CustID (inner join).
Load Both Files into Power Query
For Orders.csv:

Go to Data tab → Get Data → From File → From Text/CSV.

Select Orders.csv → Click Transform Data (to open Power Query Editor).

For Customers.xlsx:

Go to Data tab → Get Data → From File → From Excel.

Select Customers.xlsx → Choose the sheet → Click Transform Data.

2. Merge the Queries (Inner Join)
In the Power Query Editor for Orders.csv:

Go to the Home tab → Click Merge Queries → Merge Queries as New.

(Or use Merge Queries to merge into the current query.)

In the Merge dialog box:

Top table (Orders) → Select the CustID column.

Bottom table (Customers) → Select the CustID column.

Join Kind: Select Inner Join (only matching CustID rows will be kept).

Click OK.

3. Expand the Merged Customer Columns
After merging, a new column named Customers appears (with table objects).

Click the expand icon (⍰) in the Customers column header.

Check the columns you want to include (e.g., CustomerName, Email).

Uncheck "Use original column name as prefix" (optional, for cleaner names).

Click OK.

7. Pivot the Product column to show total Quantity per product.

Step-by-Step Instructions

1. Load Your Data into Power Query

Go to Data tab → Get Data → Import your data source (Excel, CSV, etc.).

Click Transform Data to open Power Query Editor.

2. Select the Product Column for Pivoting

In the Power Query Editor, select the column containing product names (e.g., "Product").

3. Pivot the Column

Go to the Transform tab.

Click Pivot Column.

In the Pivot dialog box:

Values Column: Select "Quantity" (or your numeric column to aggregate).

Aggregate Value Function: Choose Sum (or another function like Count, Average).

Click OK.

8. Append two tables with identical columns (e.g., Orders_Jan.csv + Orders_Feb.csv).
Load Both CSV Files:

Open Power BI Desktop.

Click Home > Get Data > Text/CSV.

Select Orders_Jan.csv, then repeat to load Orders_Feb.csv.

Open Power Query Editor:

Go to Home > Transform Data to open the Power Query Editor.

Append Queries:

In the Power Query Editor, go to Home > Append Queries > Append Queries as New.

In the dialog:

Select Orders_Jan as the first table.

Select Orders_Feb as the second table.

Click OK.
Review & Apply:
A new query (e.g., Append1) will be created containing data from both tables.
You can rename it (e.g., to Orders_All).
Click Home > Close & Apply to return to Power BI.

9. Use "Fill Down" to replace nulls in the Email column with the previous value.
   1. Load Your Data into Power Query
   Go to Data tab → Get Data → Import your data (Excel, CSV, etc.).

   Click Transform Data to open Power Query Editor.

   2. Select the Email Column
   Click the header of the Email column (or any column with nulls you want to fill).

   3. Apply "Fill Down"
   Go to the "Transform" tab.

   Click "Fill" → "Down".
   This replaces null values with the last non-null value above them.
10. Extract the domain (e.g., "example.com") from the Email column.
    Step-by-Step Guide
    1. Load Your Data into Power Query
    Go to Data tab → Get Data → Import your data (Excel, CSV, etc.).

    Click Transform Data to open Power Query Editor.

    2. Select the Email Column
    Click the header of the Email column (or any column with nulls you want to fill).

    3. Apply "Fill Down"
    Go to the "Transform" tab.

    Click "Fill" → "Down".
    This replaces null values with the last non-null value above them.
11. Write M-code to merge queries dynamically based on a parameter (e.g., JoinType = "Inner").

```
// Create a parameter named 'JoinType' with dropdown options
JoinType =
  type text
  meta [
    Documentation.FieldCaption = "Select Join Type",
    Documentation.AllowedValues = {
      "Inner",
      "Left Outer",
      "Right Outer",
```

```
            "Full Outer",
            "Left Anti",
            "Right Anti"
        }
    ]
  (Orders as table, Customers as table, JoinColumn as text, JoinType as text) as table =>
let
    // Convert JoinType text to Power Query's JoinKind
    JoinKind =
        if JoinType = "Inner" then JoinKind.Inner
        else if JoinType = "Left Outer" then JoinKind.LeftOuter
        else if JoinType = "Right Outer" then JoinKind.RightOuter
        else if JoinType = "Full Outer" then JoinKind.FullOuter
        else if JoinType = "Left Anti" then JoinKind.LeftAnti
        else if JoinType = "Right Anti" then JoinKind.RightAnti
        else JoinKind.Inner, // Default fallback

    // Perform the merge
    Merged = Table.NestedJoin(
        Orders,
        {JoinColumn},
        Customers,
        {JoinColumn},
        "MergedData",
        JoinKind
    ),

    // Expand all columns (optional - customize as needed)
    Expanded = Table.ExpandTableColumn(
        Merged,
        "MergedData",
        Table.ColumnNames(Customers),
        List.Transform(Table.ColumnNames(Customers), each "Customer_" & _)
    )
in
  #” Expanded”
```

12. Unpivot a table with columns like "Jan_Sales," "Feb_Sales" into a "Month" and "Sales" format.

```
#”Unpivoted”= Table.Unpivot(
    #" Expanded ",
    List.Select(Table.ColumnNames(#"PreviousStep"), each Text.Contains(_, "_Sales")),
    "Month",
    "Sales"
)
#”Unpivoted”
```

13. Handle errors in a custom column (e.g., division by zero) using try...otherwise.

Use try...otherwise with record fields:

```
powerquery
each let
   result = try [A]/[B]
in
   if result[HasError] then
      "Failed: " & result[Error][Message]  // Access error message
   else
      result[Value]  // Successful resul
```

14. Create a function in Power Query to clean phone numbers (e.g., remove dashes).
    Step-by-step: Create a Cleaning Function
    Open Power Query Editor:

    Click Home > Transform Data.

    Go to Home > Advanced Editor, and create a new blank query (Right-click Queries pane > New Query > Blank Query).

    Define the function:

    ```
    m
    let
       CleanPhoneNumber = (phone as text) as text =>
          Text.Select(phone, {"0".."9"})
    in
       CleanPhoneNumber
    ```

    Rename this query to CleanPhoneNumber (Right-click on the query name > Rename).

    ☑ Apply the Function to a Column
    Go to your main table in Power Query.

    Add a Custom Column:

    Go to Add Column > Custom Column.

    Use the formula:

    ```
    CleanPhoneNumber([Phone])
    ```

    Click OK, and you'll get a new column with cleaned phone numbers.
15. Optimize a query with 10+ steps—identify bottlenecks and simplify.
    Step-by-Step Query Optimization Strategy
    1. Open the Query Dependencies View
    Go to View > Query Dependencies.

This shows how queries relate to each other and can reveal redundant chains or circular references.

2. Review the Applied Steps (Right Pane)
Look at the Applied Steps list in Power Query Editor:

🛑 Common Bottlenecks:
Type of Step Optimization Tip
Changed Type (many times) Keep only one final Changed Type step.
Filtered Rows repeatedly Combine filters into a single step.
Added columns one-by-one Combine multiple Add Column steps into a single Custom Column.
Merge/Join with large tables Use keyed joins or reduce source rows before joining.
Grouped Rows Avoid grouping too early in the process.
Sort Sorting is expensive—only sort if truly necessary.

3. Use Buffering for Repeated Queries
If you're referencing the same table multiple times (e.g., in merges, joins, or calculations), buffer it:

m
Копировать
Редактировать
```
let
   Source = Table.Buffer(YourTableName)
in
   Source
```
This avoids recalculating the same query repeatedly.

4. Use Native Query Folding (for databases)
When connected to a data source like SQL Server, Power Query can "fold" steps into SQL.

Right-click a step > View Native Query. If it's grayed out, query folding has stopped.

Move heavy operations (filters, projections) to early steps to ensure folding continues as far as possible.

5. Remove Unused Columns Early
Drop unnecessary columns early to reduce memory usage and step processing.

m
Копировать
Редактировать
```
Table.SelectColumns(Source, {"OnlyNeededColumn1", "OnlyNeededColumn2"})
```
6. Consolidate Steps

Before:

m
Копировать
Редактировать
Step1 = Table.AddColumn(...),
Step2 = Table.ReplaceValue(Step1, ...),
Step3 = Table.RemoveColumns(Step2, ...)
After:

Combine operations inside a single step if possible.

🧠 General Best Practices
Use Table.Buffer() cautiously — only when needed, as it loads data into memory.

Avoid unnecessary "Changed Type" steps for every transformation.

Avoid loading intermediate queries into the data model unless needed.

Disable query load for helper queries (Right-click > Disable Load).

Rename columns only once.

📊 Example of Simplified M Code
Before (long and inefficient):

m
Копировать
Редактировать
Source = Excel.CurrentWorkbook(){[Name="Sales"]}[Content],
ChangedType = Table.TransformColumnTypes(Source, {...}),
FilteredRows = Table.SelectRows(ChangedType, each [Country] = "USA"),
RemovedCols = Table.RemoveColumns(FilteredRows, {"UnneededCol"}),
SortedTable = Table.Sort(RemovedCols, {{"Date", Order.Ascending}}),
...
After (simplified):

m
let
  Source = Excel.CurrentWorkbook(){[Name="Sales"]}[Content],
  Cleaned = Table.SelectRows(Source, each [Country] = "USA"),
  Final = Table.TransformColumnTypes(
          Table.RemoveColumns(Cleaned, {"UnneededCol"}),
          {{"Date", type date}, {"Amount", type number}}
)

in

Final