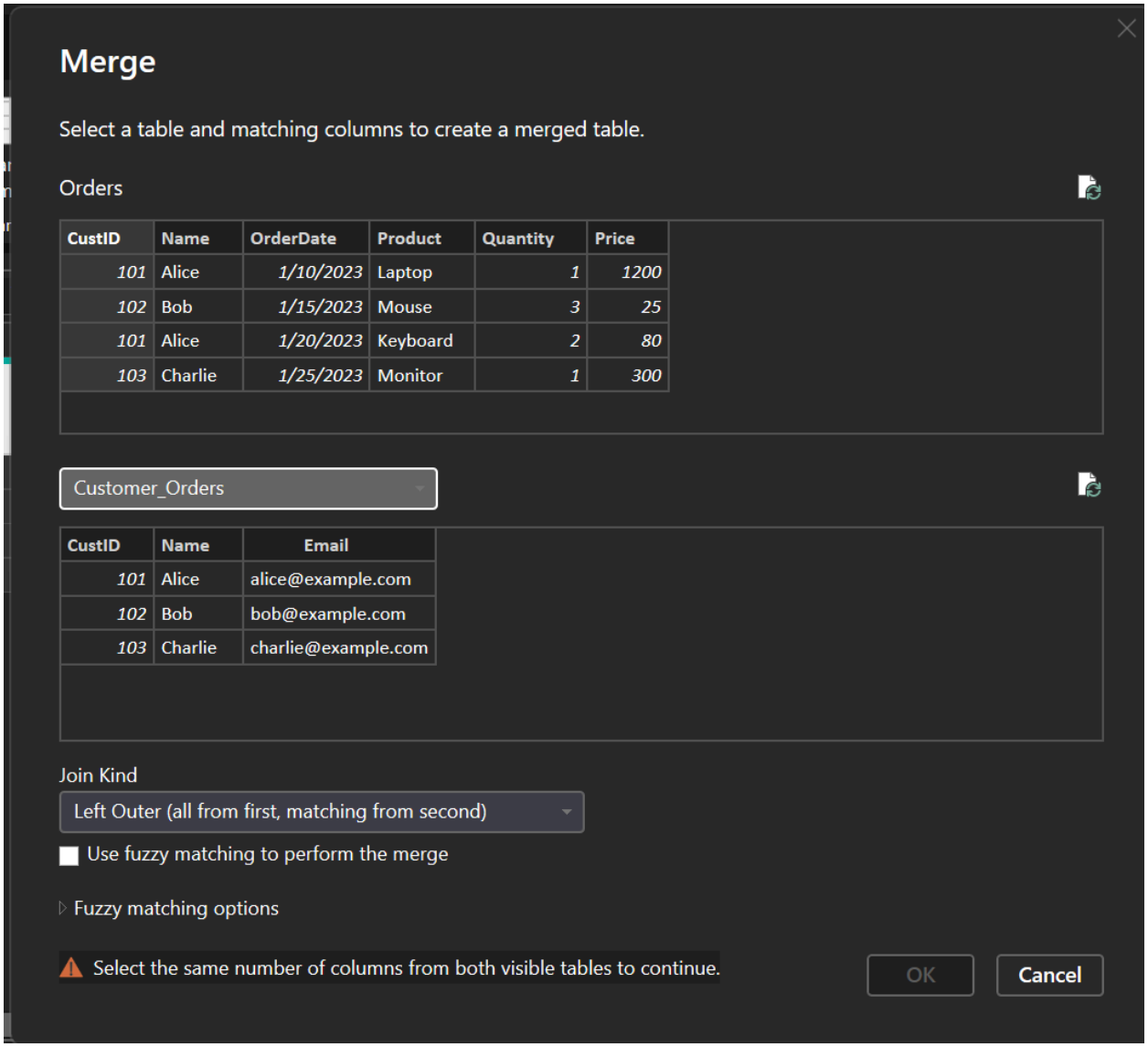
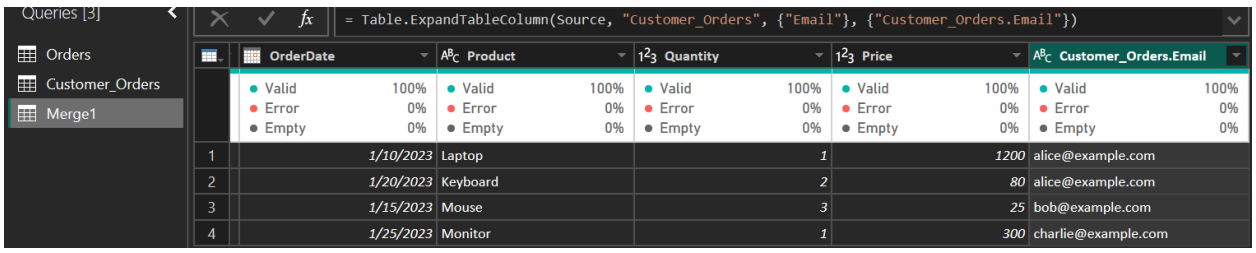
Lesson - 4 homework

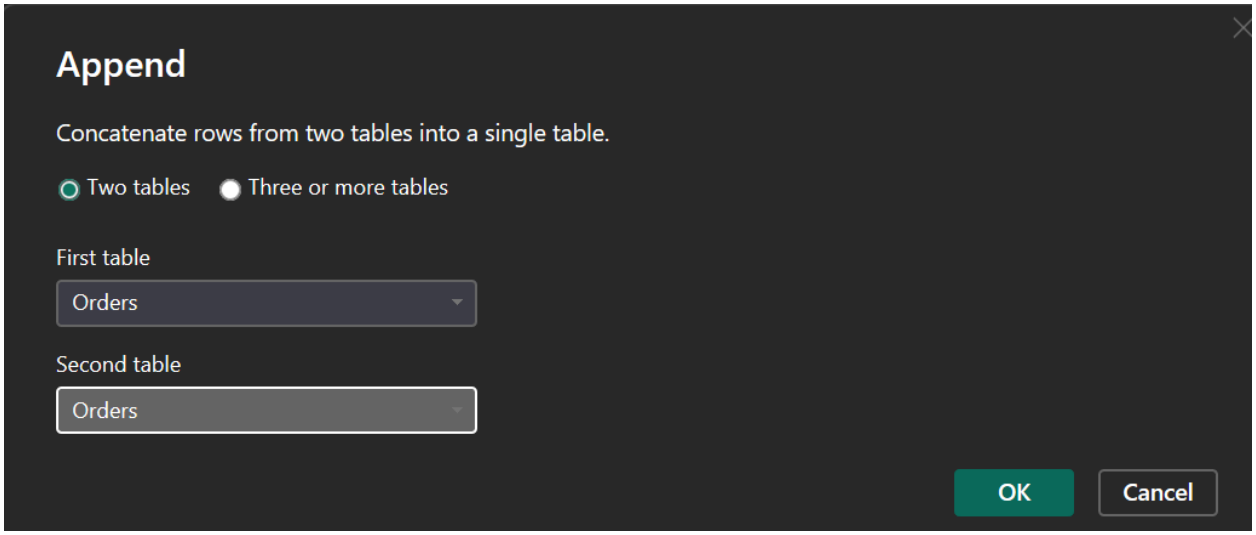
`Question 1:` What is the difference between "Merge" and "Append" in Power Query?

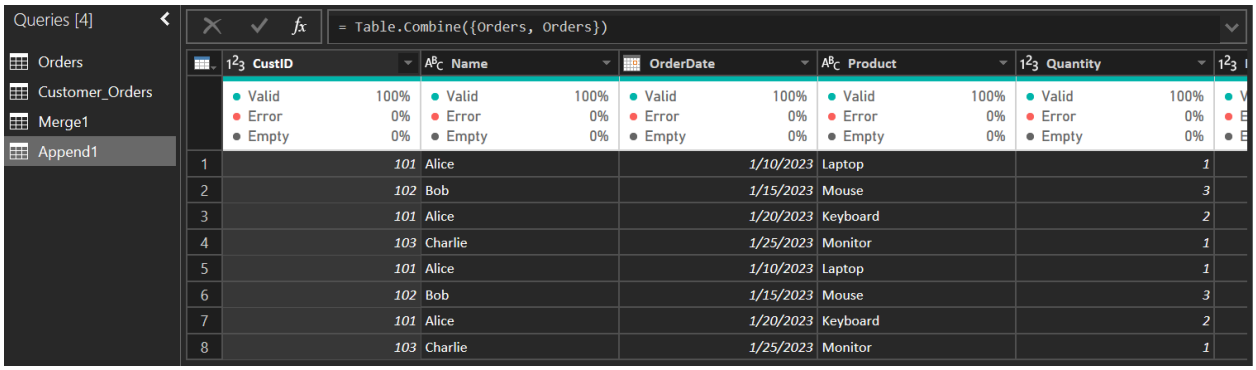
Merge is for combining tables side-by-side (like a VLOOKUP). You use it to add new columns from one table to another using a shared key, like adding customer names to an orders table using CustomerID.





Append is for stacking tables on top of each other. You use it to add new rows, like combining monthly sales tables into one yearly sales table.





	CustID	Name	OrderDate	Product	Quantity
1	101	Alice	1/10/2023	Laptop	1
2	102	Bob	1/15/2023	Mouse	3
3	101	Alice	1/20/2023	Keyboard	2
4	103	Charlie	1/25/2023	Monitor	1
5	101	Alice	1/10/2023	Laptop	1
6	102	Bob	1/15/2023	Mouse	3
7	101	Alice	1/20/2023	Keyboard	2
8	103	Charlie	1/25/2023	Monitor	1

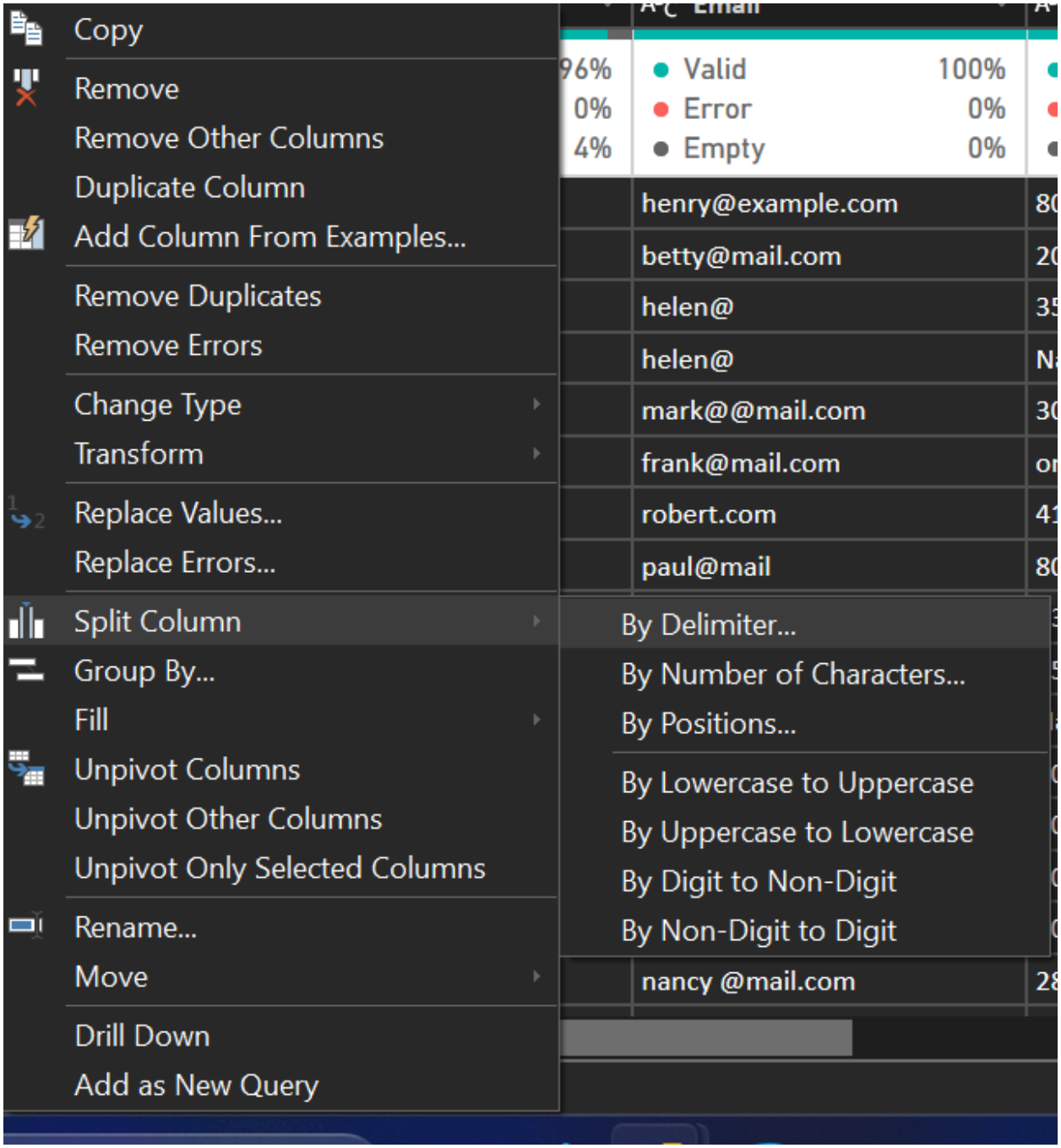
`Question 2:` How do you split a "Full Name" column into "First Name" and "Last Name"?

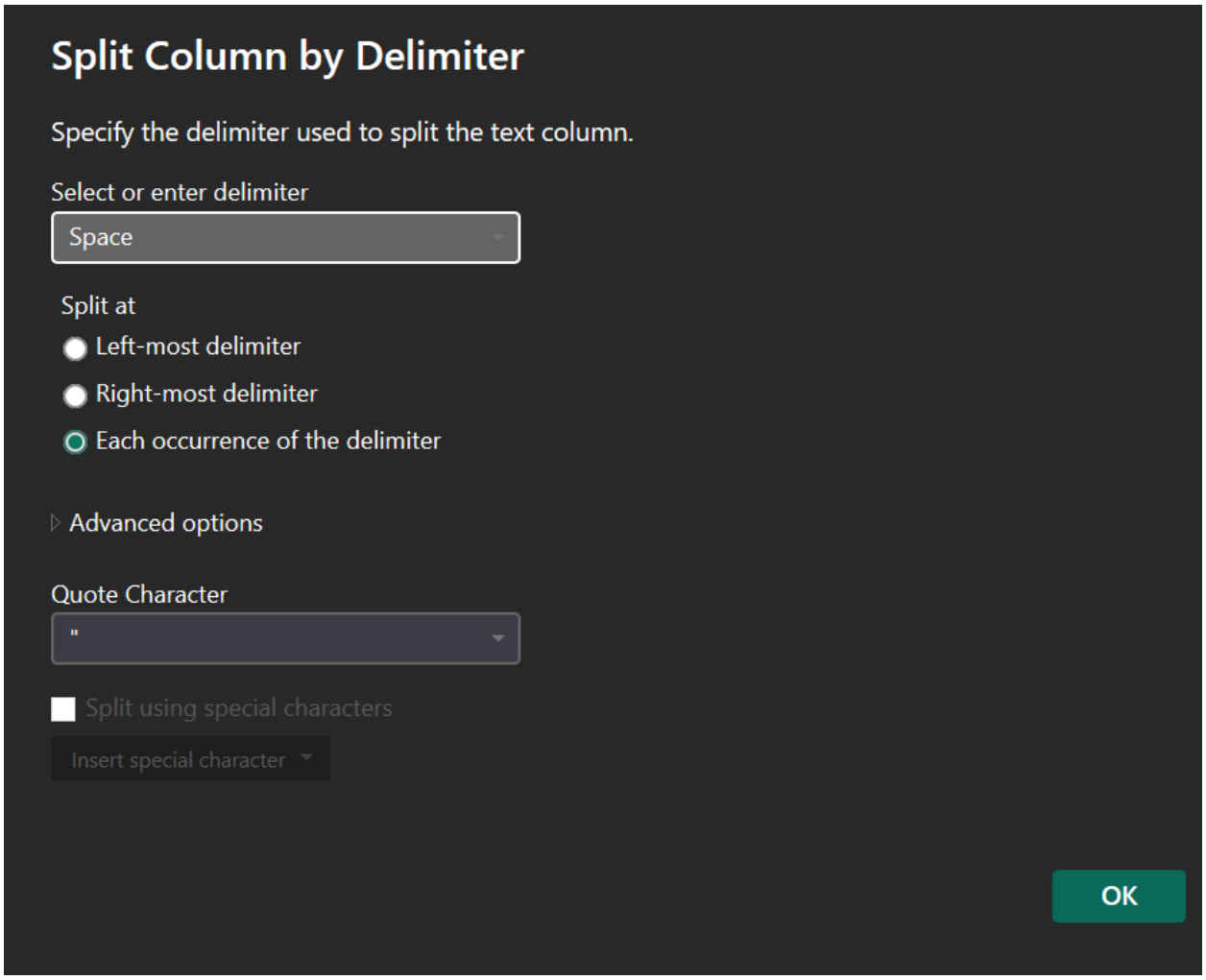
1. Select the Column: Click on the "Full Name" column header.
2. Go to the Transform Tab: Click this tab at the top.
3. Click 'Split Column':
4. Choose 'By Delimiter':
5. Select the Space: Choose Space as the delimiter.
6. Choose 'Each occurrence of the delimiter' (This is the key step for names with middle names).

Before:

	CustomerID	Name	Age	Email	PurchaseAmount
	Valid 100% Error 0% Empty 0%	Valid 100% Error 0% Empty 0%	Valid 96% Error 0% Empty 4%	Valid 100% Error 0% Empty 0%	Valid 98% Error 0% Empty 2%
1	1001	eva martin	70	henry@example.com	800.5
2	1002	mary jones	sixty	betty@mail.com	200
3	1003	Mark Anderson	Thirty-five	helen@	350
4	1004	Charlie Anderson	60	helen@	NaN
5	1005	Helen Jones	Twenty-two	mark@@mail.com	300
6	1006	CHRISTOPHER GARCIA	60	frank@mail.com	one hundred
7	1007	D@niel Thomas	Thirty-five	robert.com	410
8	1008	Frank Johnson	60	paul@mail	800.5
9	1009	Mary Martin	29	barbara@mail.com	430
10	1010	J@ckson Davis	Twenty-two	frank@mail.com	650
11	1011	HENRY WILLIAMS	25	nancy@mail.com	NaN

Process:


Copy
Remove
Remove Other Columns
Duplicate Column
Add Column From Examples...
Remove Duplicates
Remove Errors
Change Type
Transform
Replace Values...
Replace Errors...
Split Column
Group By...
Fill
Unpivot Columns
Unpivot Other Columns
Unpivot Only Selected Columns
Rename...
Move
Drill Down
Add as New Query
By Delimiter...
By Number of Characters...
By Positions...
By Lowercase to Uppercase
By Uppercase to Lowercase
By Digit to Non-Digit
By Non-Digit to Digit
Valid
Error
Empty
henry@example.com
betty@mail.com
helen@
helen@
mark@@mail.com
frank@mail.com
robert.com
paul@mail
nancy @mail.com



After:

Valid 100%	Valid 91%	Valid 100%
Error 0%	Error 0%	Error 0%
Empty 0%	Empty 9%	Empty 0%
1012	Bob	Jones
1013	James	Williams
1014	David	Williams
1015	Thomas	Thompson
1016	Sophia	Anderson
1017	Thom@S	Davis
1018	N@Ncy	Davis
1019	Patricia	Jones
1020	Kevin	Thompson
1021	Fr@Nk	Smith

`Question 3:` What is "Pivot Columns" used for?

`Pivot Columns is used to reshape your data from a "long" format into a "wide" format. It takes the unique values from one column and transforms them into multiple new columns, summarizing another column's values for each of those new categories.`

Imagine you have a tall, long table like this:

Product	Month	Sales
Apples	Jan	100
Apples	Feb	150
Oranges	Jan	75
Oranges	Feb	120

After pivoting the "Month" column, you get a wide, summary table:

Product	Jan	Feb
Apples	100	150
Oranges	75	120

`Question 4:` What is "Pivot Columns" used for?

1. **In the Power Query Editor, look at the "Query Settings" pane on the right.**
2. **You'll see a list called "Applied Steps". This is a history of everything you've done.**
3. **To undo the last thing you did, simply click on the step above it in the list.**
4. **The query will instantly revert to how it was at that point, effectively "deleting" all the steps that came after.**
5. **To delete a specific step: Click the X next to the step's name. Be careful, as this will also delete all steps that come after it, since each step builds on the previous one.**

`Question 5:` What is "Pivot Columns" used for?

Duplicate: Creates a full, independent copy of the query. Changing the original or the copy does not affect the other. Use it to experiment without messing up your original.

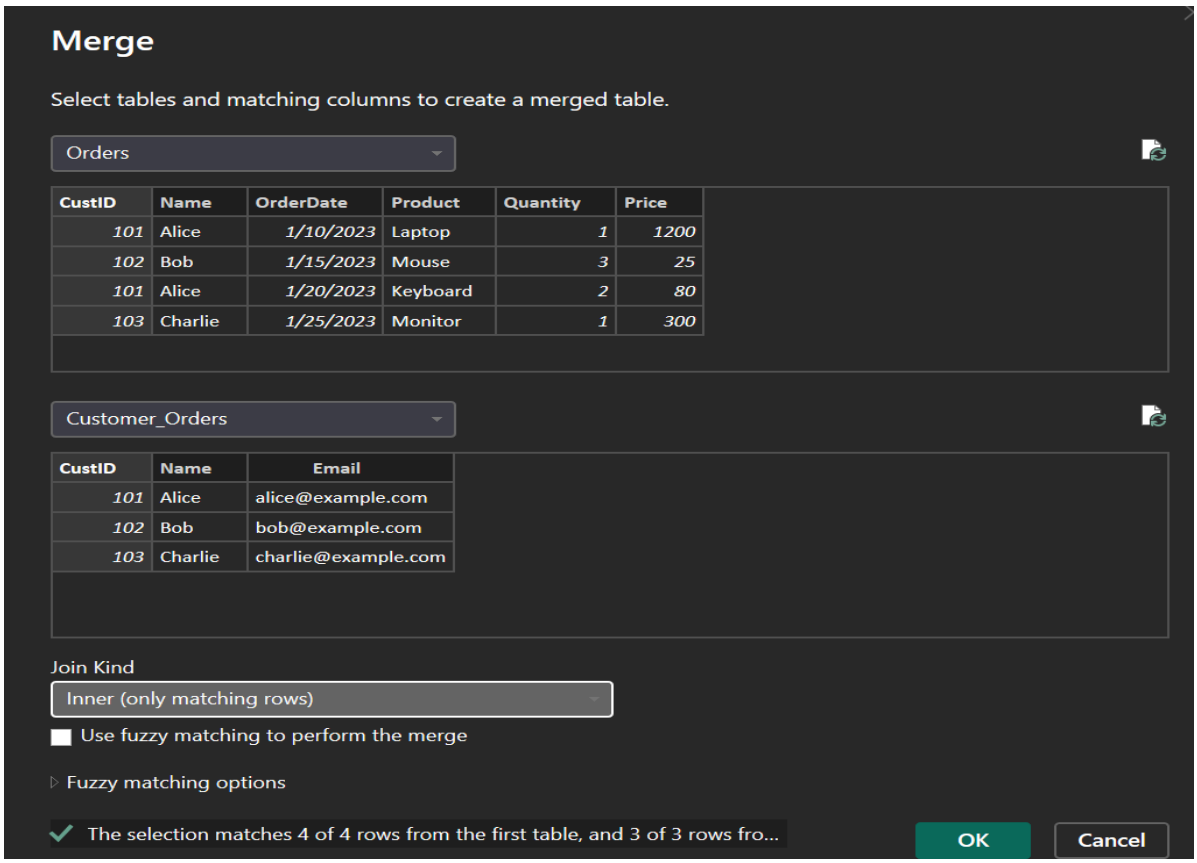
Reference: Creates a new query that points to the original. If you change the original query later (like adding a step), the reference will automatically get those changes too. Use it to build different tables (like a summary) from the same cleaned data source.

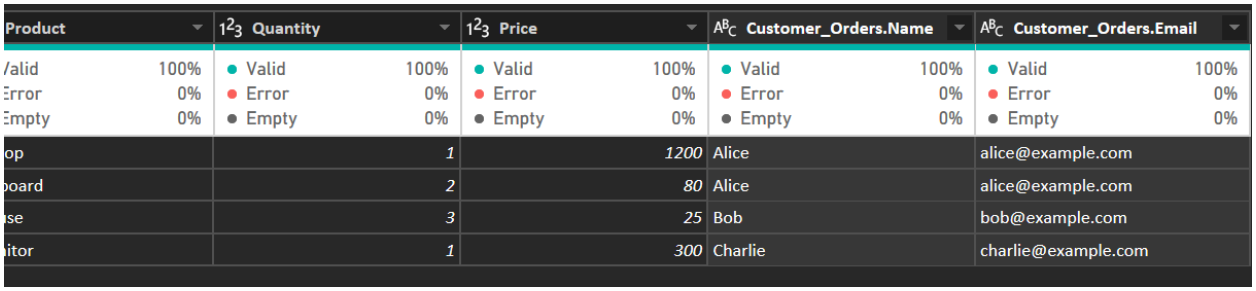
Analogy:

Duplicate is like making a photocopy of a document. Editing the original or the copy doesn't change the other.

Reference is like creating a shortcut or a link to the original document. If you update the original, the shortcut shows the latest version.

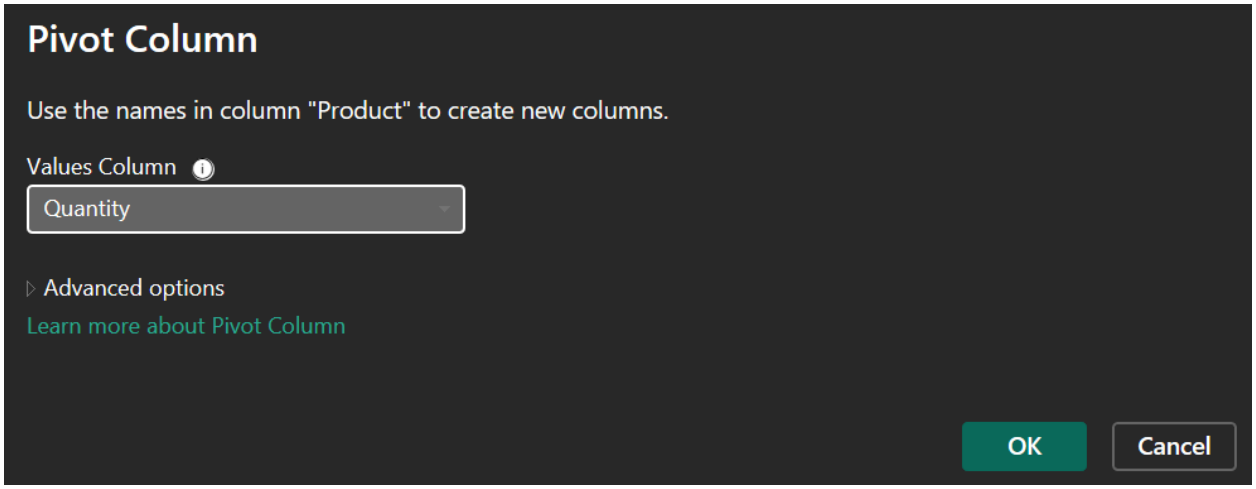
`Question 6:` Merge Orders.csv and Customers.xlsx on CustID (inner join).





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

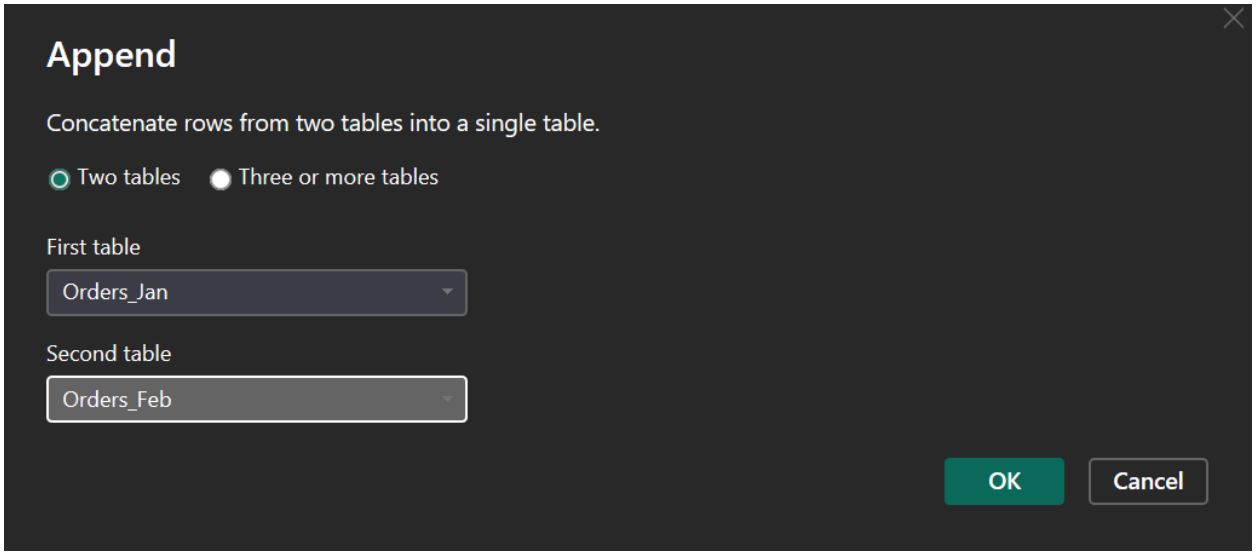
1. **Select the Product column.**
2. **Go to the Transform tab.**
3. **Click Pivot Column.**
4. **In the window that pops up:**
5. **Values Column: Choose Quantity from the dropdown.**
6. **Aggregate Value Function: Ensure Sum is selected (it usually is by default).**



123 Laptop	123 Keyboard	123 Mouse	123 Monitor
Valid 25%	Valid 25%	Valid 25%	Valid 25%
Error 0%	Error 0%	Error 0%	Error 0%
Empty 75%	Empty 75%	Empty 75%	Empty 75%
1	null	null	null
null	2	null	null
null	null	3	null
null	null	null	1

`Question 8:` Append two tables with identical columns (e.g., Orders_Jan.csv + Orders_Feb.csv).

1. **Go to the Home tab.**
2. **Click Append Queries (As new)**
3. **In the window, select your second table (e.g., Orders_Feb).**

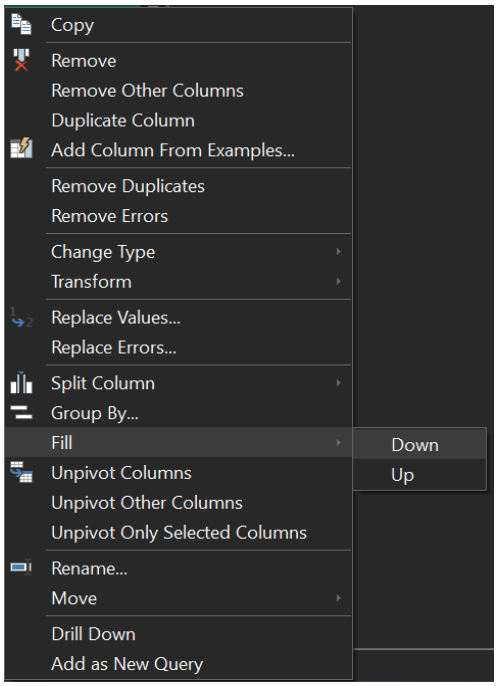


	CustID	Name	OrderDate	Product	Quantity	
	Valid 100%	Valid 100%	Valid 100%	Valid 100%	Valid 100%	
	Error 0%	Error 0%	Error 0%	Error 0%	Error 0%	
	Empty 0%	Empty 0%	Empty 0%	Empty 0%	Empty 0%	
1	101	Alice	1/10/2023	Laptop	1	
2	102	Bob	1/15/2023	Mouse	3	
3	101	Alice	1/20/2023	Keyboard	2	
4	103	Charlie	1/25/2023	Monitor	1	
5	101	Alice	2023-02-10	Laptop	1	
6	102	Bob	2023-02-15	Mouse	3	
7	101	Alice	2023-02-20	Keyboard	2	
8	103	Charlie	2023-0-25	Monitor	1	

`Question 9:` Use "Fill Down" to replace nulls in the Email column with the previous value.

1. **Select the Email column.**
2. **Go to the Transform tab.**
3. **Click Fill → Down.**

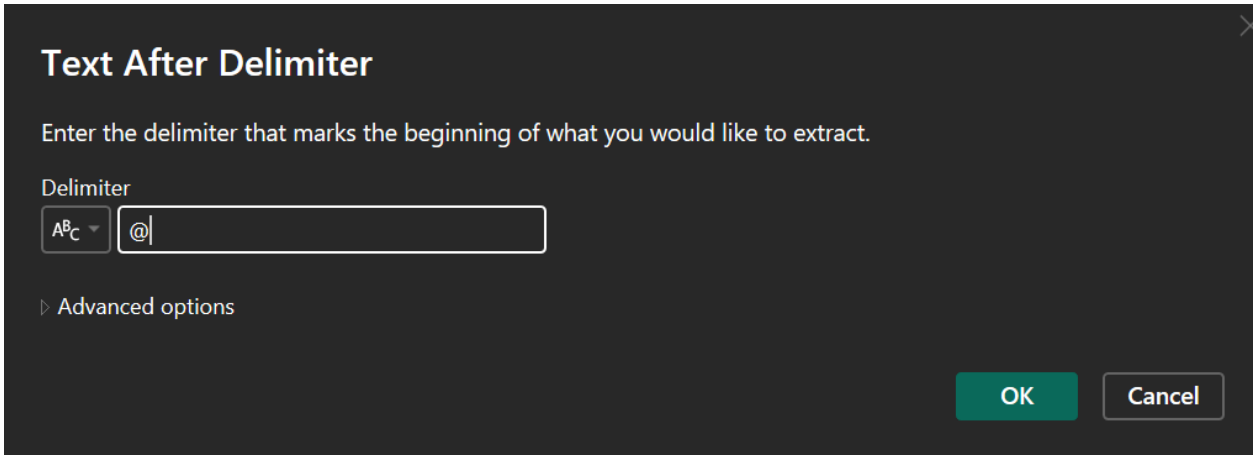
All the nulls in the Email column will be replaced with the value from the cell above them. Done.



	CustID	Name	Email
	Valid 100%	Valid 100%	Valid 100%
	Error 0%	Error 0%	Error 0%
	Empty 0%	Empty 0%	Empty 0%
1	101	Alice	alice@example.com
2	102	Bob	bob@example.com
3	103	Charlie	charlie@example.com

Question 10: Extract the domain (e.g., "example.com") from the Email column.

Select the Email column.
Go to the Add Column tab.
Click Extract → Text After Delimiter.



	CustID	Name	Email	domain
	Valid 100% Error 0% Empty 0%	Valid 100% Error 0% Empty 0%	Valid 100% Error 0% Empty 0%	Valid 100% Error 0% Empty 0%
1	101	Alice	alice@example.com	example.com
2	102	Bob	bob@example.com	example.com
3	103	Charlie	charlie@example.com	example.com

Question 11: Write M-code to merge queries dynamically based on a parameter (e.g., JoinType = "Inner").

```
let
    // This converts your JoinType parameter to the correct JoinKind
    GetJoinKind =
        if JoinType = "Inner" then JoinKind.Inner
        else if JoinType = "Left" then JoinKind.LeftOuter
        else if JoinType = "Right" then JoinKind.RightOuter
        else if JoinType = "Full" then JoinKind.FullOuter
        else JoinKind.LeftOuter, // Fallback to LeftOuter if something goes wrong

    // Your merge - now dynamic!
    Source = Table.NestedJoin(
        Orders,
        {"CustID"},
        Customer_Orders,
        {"CustID"},
        "Customer_Orders",
        GetJoinKind // Using the parameter here instead of hardcoded JoinKind.LeftOuter
    )
in
    Source
```

Price	Customer_Orders.CustID	Customer_Orders.Name	Customer_Orders.Email	Customer_Orders
Valid 100% Error 0% Empty 0%	Valid 100% Error 0% Empty 0%	Valid 100% Error 0% Empty 0%	Valid 100% Error 0% Empty 0%	Valid Error Empty
1200	101	Alice	alice@example.com	example.com
80	101	Alice	alice@example.com	example.com
25	102	Bob	bob@example.com	example.com
300	103	Charlie	charlie@example.com	example.com

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

```
let
    Source = Jan-Feb,
    // Convert the text dates to proper date format first
    ChangeType = Table.TransformColumnTypes(Source,{{"Month", type date}}),
    // Extract month name
    ExtractMonth = Table.TransformColumns(
        ChangeType,
        {"Month", each Date.MonthName(_), type text}
    ),
    // Get short month names (optional)
    ShortMonth = Table.TransformColumns(
        ExtractMonth,
        {"Month", each Text.Start(_, 3), type text}
    )
in
    ShortMonth
```

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

```
let
    Source = Csv.Document(File.Contents("C:\Users\apple servise\Documents\Data_Analytics\Power_BI Classes\bi_class_4\powerbi_dirty_dataset_fullnames_100.csv"),[Delimiter=",", Columns=7, Encoding=1252, QuoteStyle=QuoteStyle.None]),
    #"Promoted Headers" = Table.PromoteHeaders(Source, [PromoteAllScalars=true]),
    #"Removed Other Columns" = Table.SelectColumns(#"Promoted Headers",{"CustomerID", "PurchaseAmount"}),
    #"Changed Type" = Table.TransformColumnTypes(#"Removed Other Columns",{{"PurchaseAmount", Currency.Type}}),
    #"Replaced Errors" = Table.ReplaceErrorValues(#"Changed Type", {{"PurchaseAmount", null}}),


    #"Added Custom" = Table.AddColumn(
        #"Replaced Errors",
        "AdjustedPurchase",
        each try [PurchaseAmount] * 1.1 otherwise 0,
        type number
    )
in
    #"Added Custom"
```

	CustomerID	PurchaseAmount	AdjustedPurchase
	Valid 100%	Valid 81%	Valid 81%
	Error 0%	Error 0%	Error 0%
	Empty 0%	Empty 19%	Empty 19%
24	1024	-150.00	-165
25	1025	900.00	990
26	1026	400.75	440.825
27	1027	285.00	313.5
28	1028	null	null
29	1029	null	null
30	1030	-150.00	-165
31	1031	450.75	495.825
32	1032	800.50	880.55
33	1033	300.00	330

Question 14: Create a function in Power Query to clean phone numbers (e.g., remove dashes).

```
(phoneNumber as text) as text =>
let
    Cleaned = Text.Remove(phoneNumber, {" ", "-", "(", ")", ".", "+"}),

    Result = Cleaned
in
  Result
```

Question 15: Optimize a query with 10+ steps—identify bottlenecks and simplify.

Applied Steps:

1. **Source: Import from C:\Data\Customers.csv (has 20 columns)**
2.
3. **Promoted Headers: First row contains column names.**
4.
5. **Changed Type: Changed all columns to type text to prevent errors.**
6.
7. **Renamed Columns: Renamed Phone Number to Phone.**
8.
9. **Added Custom Column: Created FullName by combining FirstName and LastName columns.**
10.
11. **Removed Columns: Removed FirstName and LastName (now that we have FullName).**
12.
13. **Added Custom Column: Used a complex Text.Remove formula to clean the Phone column.**
14.
15. **Filtered Rows: Kept only rows where Country = "USA".**
16.
17. **Changed Type (again): Changed SignUpDate from text to a date type.**
18.
19. **Replaced Errors: For any rows where the date conversion failed, replaced with null.**
20.
21. **Filtered Rows (again): Kept only rows where SignUpDate is not null.**
22.
23. **Reordered Columns: Moved FullName to the first column.**