Microsoft Excel: Power Query

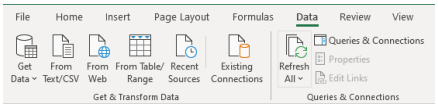
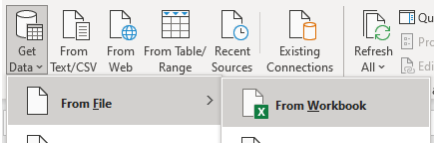
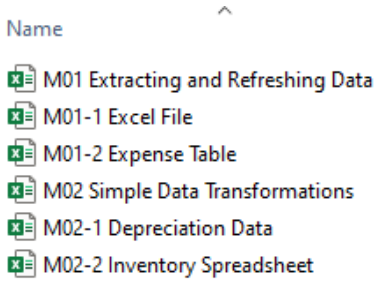
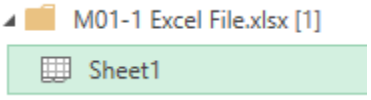
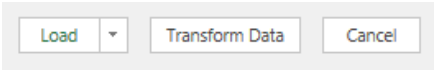
Importing, cleaning, and managing data

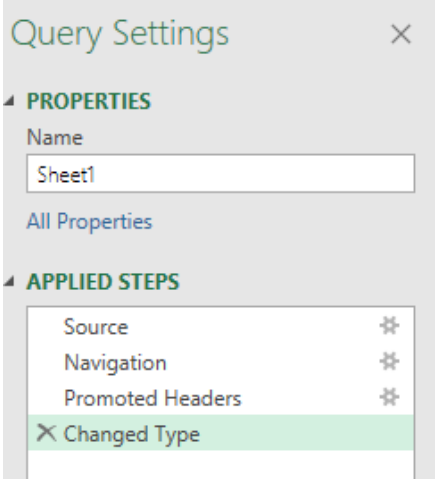
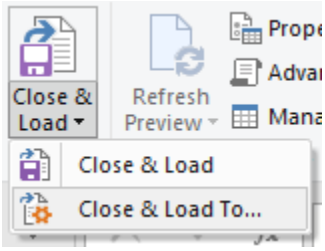
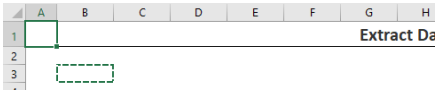
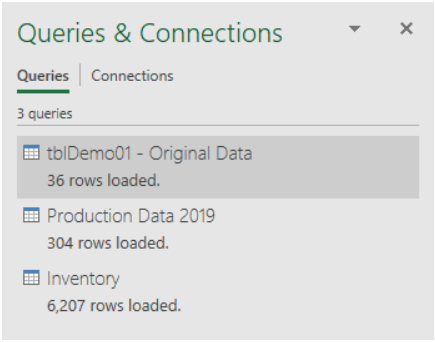
Contents

Module 01: Extracting and Using Data

Power Query's responsibilities in Excel are termed 'ETL,' which stands for **Extract, Transform, and Load**. In this module, we will explore the first stage – extraction of data. In this module, we pull data from outside data sources into our Excel file.

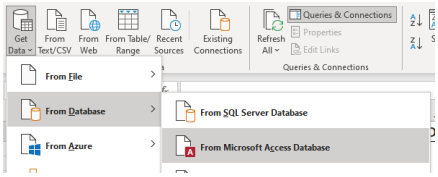
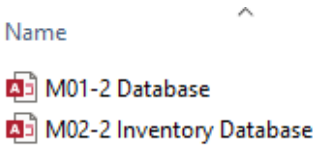
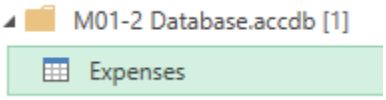
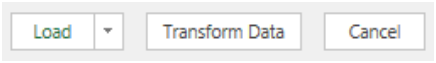
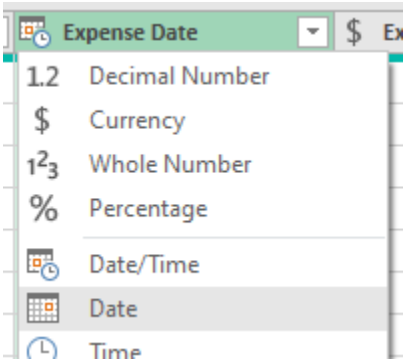
Exercise 01: Extracting Data from Excel

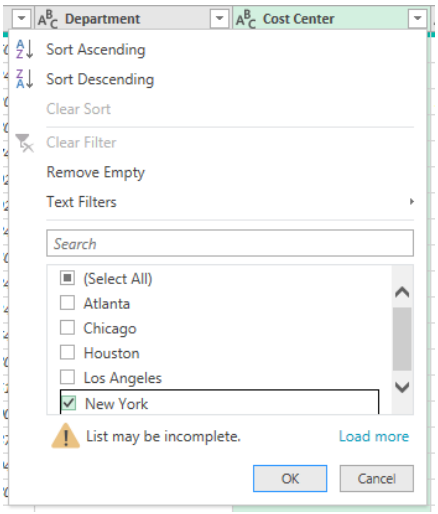
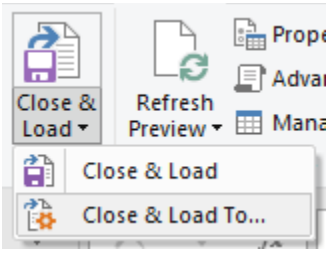
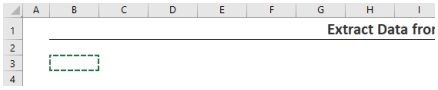
Open the first module's work file	Open **M01 Extracting and Refreshing Data.xlsx** and find tab **L01 Extract from Excel**.	
Find the Power Query tools	Click the **Data Tab**, and investigate the Groups labeled **Get & Transform Data** and **Queries & Connections**. All the tools in these two groups relate to Power Query.	
Query an Excel file	Click **Get Data > From File > From Workbook**.	
Choose the file	In the File Explorer, find the location of your Working Files folder, and select **M01-1 Excel File**.	
Choose the data	In the Navigator, choose **Sheet1** as the location of your data.	
Open the Power Query Editor	Click **Transform Data**.	

Investigate the steps taken on the data	By default, the **Applied Steps** panel should contain the steps for finding the Source, Navigation to the appropriate tab within the file, Promoting Headers to the tops of the columns, and detecting and changing the Data Types.	
Rename the query	Notice that your query is currently named after the tab *Sheet1*, and that this is inadequate for identifying the query over time. Click the **Name** box and rename the query to ***Inventory***.	
Load the data	The final stage of ETL is **Loading** the data to our spreadsheet. Click **Close & Load > Close & Load To...**	
Choose the load location	Now, choose cell **B3** on the L01 tab as the location. Click **OK**.	
Note your existing queries	When the query is loaded, the **Queries & Connections** panel should open with the existing queries in the file. If this does not open, click **Data Tab > Queries & Connections**.	

In this exercise, rather than opening, selecting, copying, and pasting the data into our Excel file, we have *extracted* the data through Power Query. This data can also be refreshed as the original Excel file is updated. Next, we'll extract the data from a database.

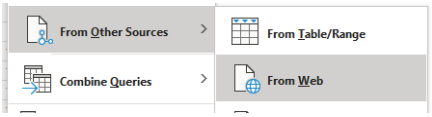
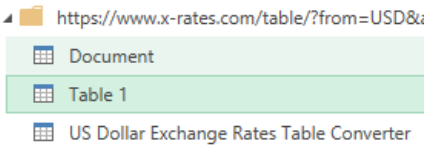
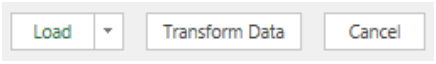
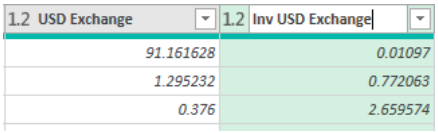
Exercise 02: Extracting Data from a Database

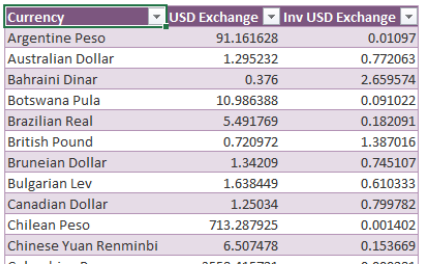
Open the appropriate tab	Select **L02 Extract from Database**.	
Query a database	Click **Data Tab > Get Data > From Database > From Access Database**.	
Find the appropriate database	Query **M01-2 Database.accdb**.	
Choose the appropriate table	Select the **Expenses** table.	
Open Power Query Editor	Choose **Transform Data**.	
Change data type	On the **Expense Date** column, click the icon for Date/Time, and switch the data type to **Date**.	

Filter the data	From the **Cost Center** drop-down, choose only the checkbox for **New York**.	
Load the data	Click **Home Tab > Close & Load > Close & Load To...**	
Load to the appropriate location	Choose **Existing Worksheet**, and load the data to cell **B3**.	

In this exercise, we see that not only can we also query a database, but we can query only a subset of the data (the New York Cost Center, for example) from a much larger dataset.

Exercise 03: Extracting Data from a Website

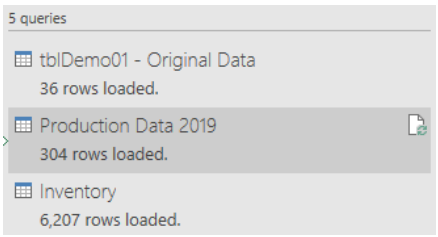
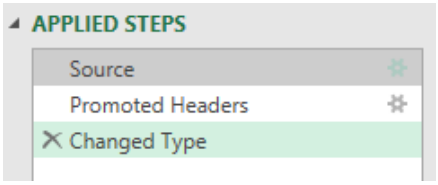
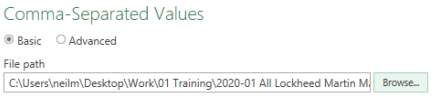
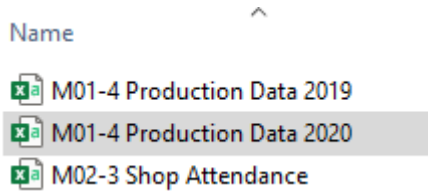
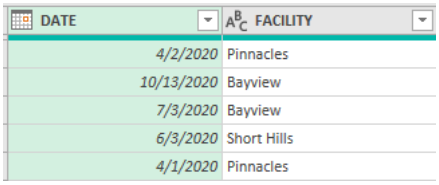
Open the appropriate tab	Navigate to **L03 Extract from Website**.	
Get the website information	Note the address in the white rectangle. Copy this URL for use in the query. If you'd like, use a web browser to investigate the website before querying.	
Query the website	Click **Data Tab > Get Data > From Other Sources > From Web**.	
Paste the address into the dialog	Using the **Basic** setting, enter the URL you copied.	
Choose the appropriate data	Notice that the site has multiple tables available. Click each to investigate the contents, then choose **Table 1**.	
Open the Power Query Editor	Choose **Transform Data**.	
Rename columns	**Double-click** each column header, and rename them **Currency**, **USD Exchange**, and **Inv USD Exchange**.	

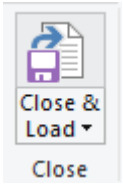
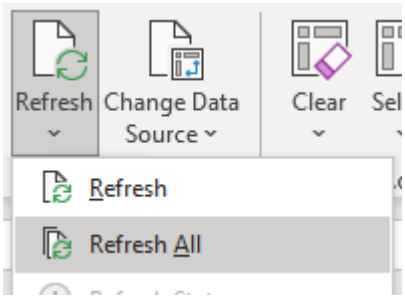
Rename the query	In the **Name** box, name this query **Exchange Rates**.	
Load the data	Click **Home Tab > Close & Load > Close & Load To...** Then choose the appropriate cell to load the data.	

Currency	USD Exchange	Inv USD Exchange
Argentine Peso	91.161628	0.01097
Australian Dollar	1.295232	0.772063
Bahraini Dinar	0.376	2.659574
Botswana Pula	10.986388	0.091022
Brazilian Real	5.491769	0.182091
British Pound	0.720972	1.387016
Bruneian Dollar	1.34209	0.745107
Bulgarian Lev	1.638449	0.610333
Canadian Dollar	1.25034	0.799782
Chilean Peso	713.287925	0.001402
Chinese Yuan Renminbi	6.507478	0.153669

In this third exercise, we have now seen that Power Query can pull data from Excel files, databases, and even live websites. We'll now move forward with the *transformation* of the data, rather than focusing on each of the data sources we can query.

Exercise 04: Refreshing Queries Used in a Pivot Table

Navigate to the next sheet	Select the sheet **L04 Update and Refresh Queries**.	
Investigate the scenario	On the left side of the page, the data table you see is being pulled from a file called **M01-4 Production Data 2019.csv**. The two Pivot Tables and two Pivot Charts are then based on the data being extracted from that file. We will now update this query from the 2019 data to the 2020 data, automatically updating all the analysis.	
Open existing query	On the **Queries & Connections** panel, **double-click** the query **Production Data 2019** to open in the Power Query Editor.	
Modify a step	In the **Applied Steps** panel, click the **Gear icon** for the entry **Source** to change the source file.	
Find the new file	On the **Comma-Separated Values** dialog, click **Browse** to find the new CSV file.	
Navigate to the file	Find **M01-4 Production Data 2020.csv**.	
Extract the new data	Click **OK**, and watch the data in the Editor update to 2020.	

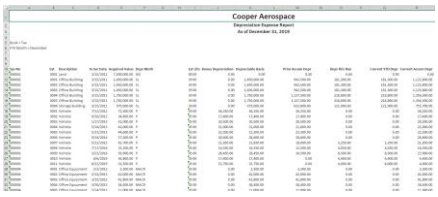
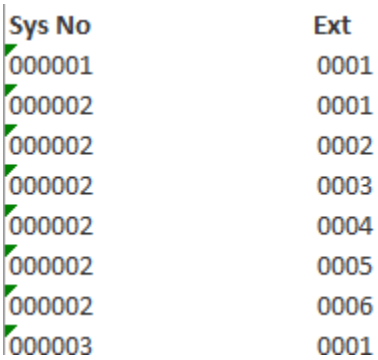
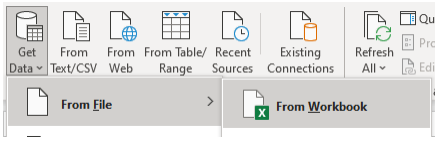
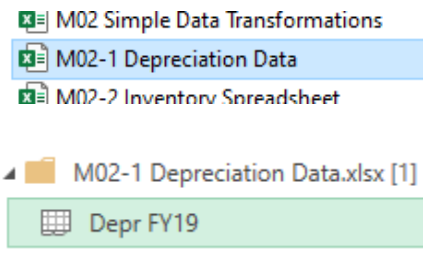
Refresh the data	Now, click the **Close & Load** button to reload the data into the spreadsheet.	
Refresh the Pivot Tables	Now, select one of the Pivot Tables and click **Analyze > Refresh > Refresh All**.	

The benefit of creating a query in Power Query is that is establishes *how* to bring in external data. If the external data changes, or if we simply have a new dataset with the same structure, we should easily be able to perform the same actions on the new data.

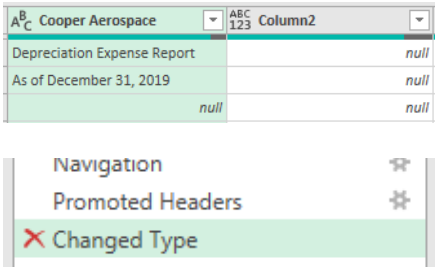
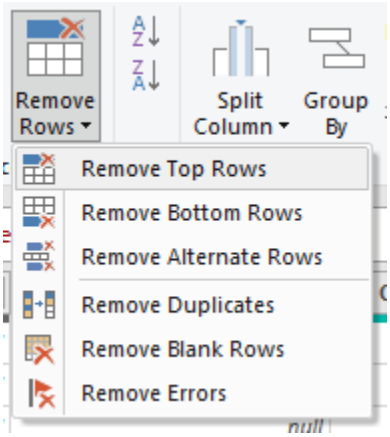
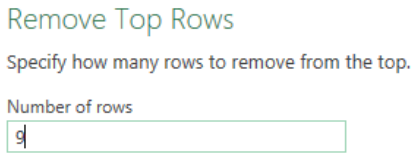
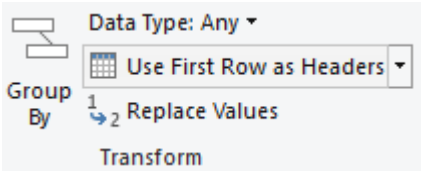
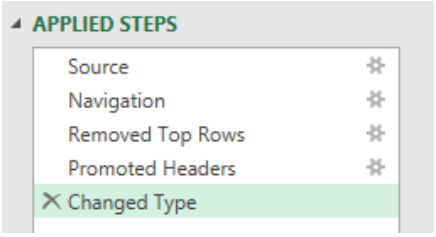
Module 02: Simple Transformations with Power Query

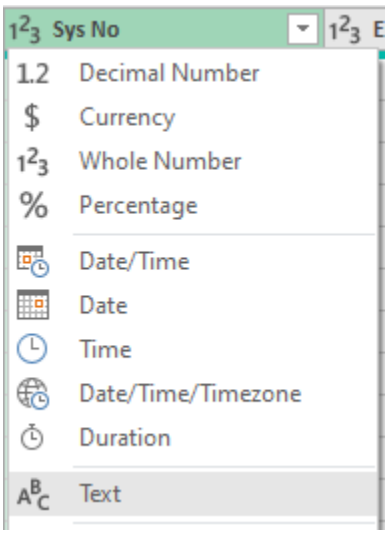
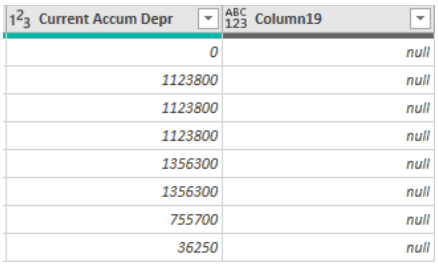
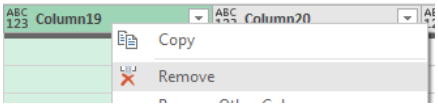
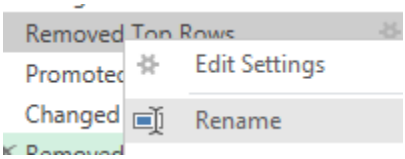
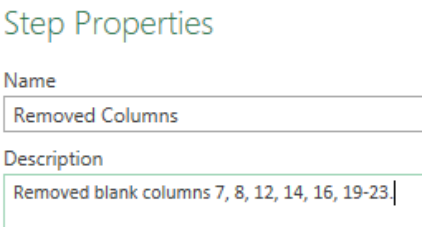
Once the data has been extracted, there is often significant 'clean up' required before it becomes a useable data table. In the next two modules, we'll investigate common data issues and the transformations that solve those problems.

Exercise 01: Row and Column Transformations

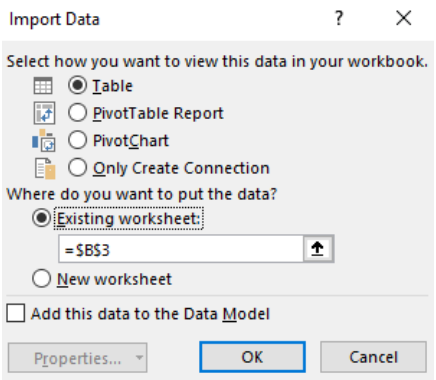
Investigate the data	Before beginning the exercise, open the file **M02-1 Depreciation Data.xlsx** to see the potential issues.	
Recognize row concerns	Currently, the top of the worksheet includes headings and blank rows that will need to be removed.	
Recognize column issues	Columns **G:H** are blank; columns **A:B** use values that appear numeric but include many leading zeroes.	
Open the working file	Close **M02-1 Depreciation Data**, and open **M02 Simple Data Transformations.xlsx**. Find **L01 Rows and Columns**.	
Query the first dataset	Click **Data Tab > Get Data > From File > From Workbook**.	
Find the dataset	Navigate to **M02-1 Depreciation Data**, then select **Depr FY19**. Click **Transform Data**.	

Sys No	Ext
000001	0001
000002	0001
000002	0002
000002	0003
000002	0004
000002	0005
000002	0006
000003	0001

Remove two incorrect steps	Notice that in this scenario, Power Query assumed the first row was our header row, and then tried to determine appropriate data types for the various columns. In the **Applied Steps** panel, click the **X icon** for these two steps to delete them.	
Remove unnecessary rows	Note that rows 1:9 are the titles and blank rows at the top of the dataset. Click **Home Tab > Remove Rows > Remove Top Rows** to select the rows to be removed.	
Specify the appropriate number of rows	Type **9** into the Remove Top Rows dialog, and click **OK**.	
Promote headers	Click **Home Tab > Use First Row as Headers** to promote the new first row into the headers of the columns.	
Recognize incorrect assumptions	When the Promoted Headers step was added to the **Applied Steps** panel, note that Power Query automatically added a **Changed Type** step with the default choices.	

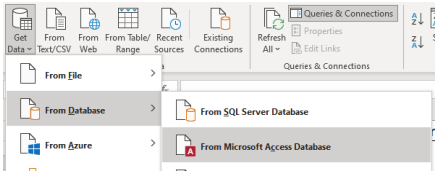
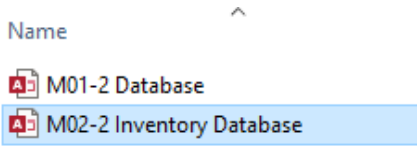
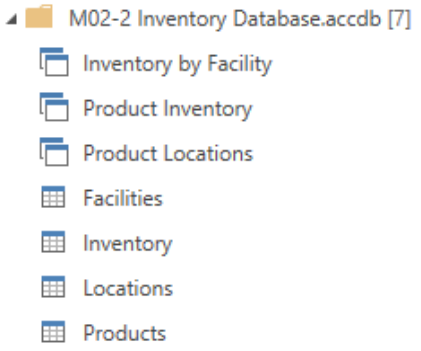
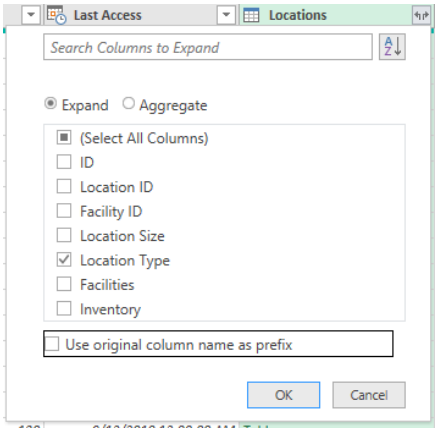
Modify current step	With the **Changed Type** step selected at right, use the data type drop-down for **Sys No** and for **Ext** to change these columns from **Whole Number** to **Text** (keeping the leading zeroes).	
Find blank columns	Under each column header is a colored indicator – dark grey for blank, blue-green for valid, and red for errors. Identify columns **7, 8, 12, 14, 16, 19-23** as empty columns.	
Remove blank columns	For each blank column, **right-click** and choose **Remove**.	
Add documentation	Notice the generic nature of the Step names on the right side: **Removed Top Rows** and **Removed Columns**, for example. We should improve on this information so we can pinpoint what needs to change in the future. **Right-click** the step **Removed Top Rows** and choose **Rename** to **Removed Top 9 Rows**.	
Add comments	**Right-click** the step **Removed Columns**, and choose **Properties...** In the Description field, add the text *"Removed blank columns 7, 8, 12, 14, 16, 19-23"*	

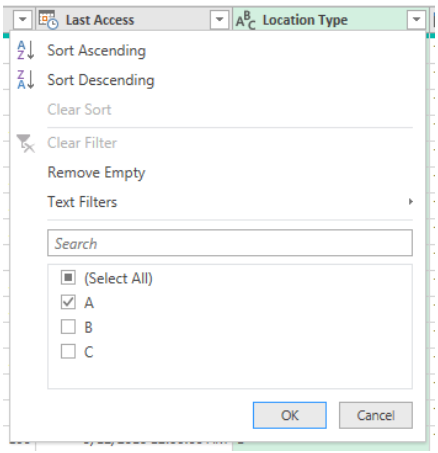
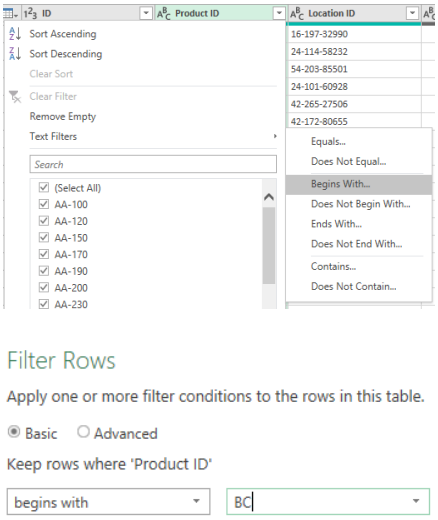
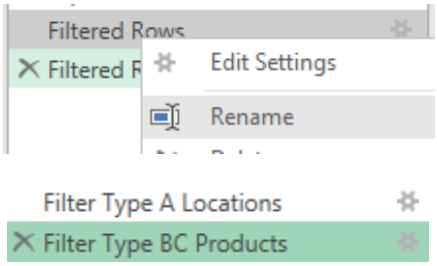
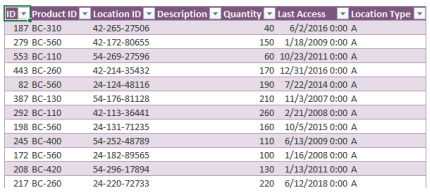
Consume the new documentation	**Hover** your mouse above the white bubble next to **Removed Columns** and see the pop-up give you more information about that step.
Load the data	Click **Home Tab > Close & Load > Close & Load To...** and choose the appropriate location in your spreadsheet to load the data.



Not only does Power Query handle which rows and columns should be included, but every step of the process is documented for later modification. Adding clear information to this documentation makes later work much easier.

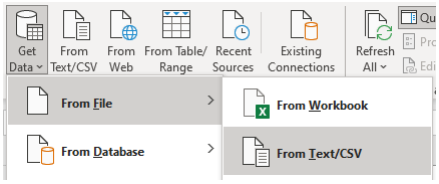
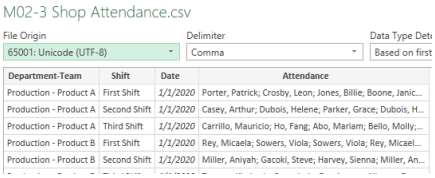
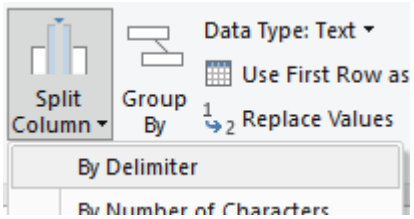
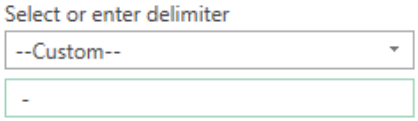
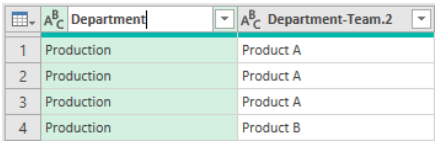
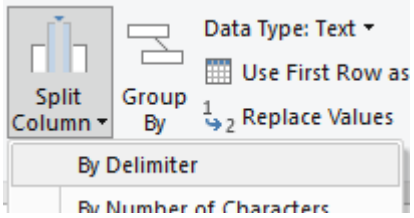
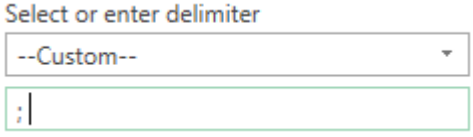
Exercise 02: Filtering Rows

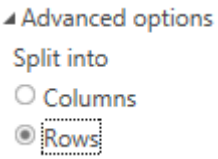
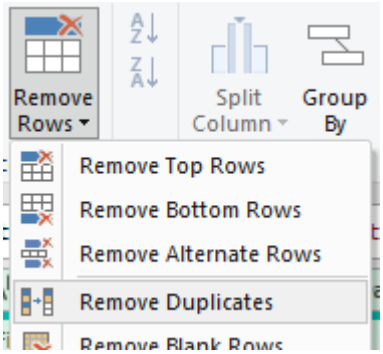
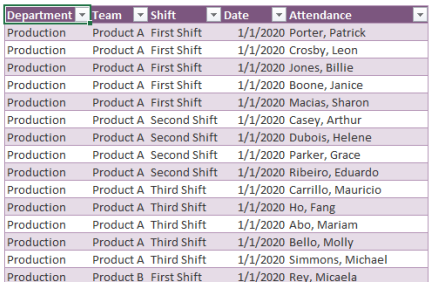
Move to the next worksheet	Select **L02 Filtering**	
Query a database	Click **Data Tab > Get Data > From Database > From Access Database**.	
Select the appropriate file	Choose **M02-2 Inventory Database**.	
Choose the appropriate table	Select **Inventory** from the list of tables and queries.	
Expand the Locations	Because the database has a relationship between the **Inventory** table and the **Locations** table, Power Query automatically makes a **Locations** column available. Click the arrow button and choose only the checkbox for **Location Type**.	

Filter by Location Type	Click the drop-down arrow for the **Location Type** column and leave only type **A** selected.	
Filter by product type	We are only interested in the Beckwith products, all of which have an ID number that begins with ***BC***. Click the drop-down arrow for the **Product ID** column, and choose **Text Filters > Begins with**. Enter **BC** as the value every ID number should begin with.	
Change the documentation	**Right-click** the step **Filtered Rows** and **Rename** it *Filter Type A Locations*. **Right-click** the step **Filtered Rows1** and **Rename** it *Filter Type BC Products*.	
Load the data	Click **Home Tab > Close & Load > Close & Load To...** and choose to load the data to the **L02 Filtering** sheet.	

Filtering the query guarantees that only those records you need in your dataset are included.

Exercise 03: Text to Columns

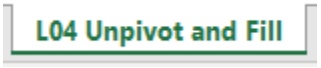
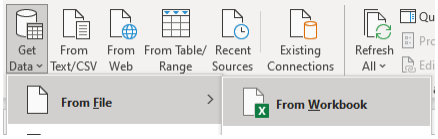
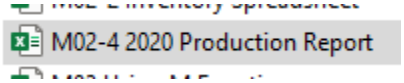
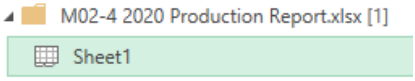
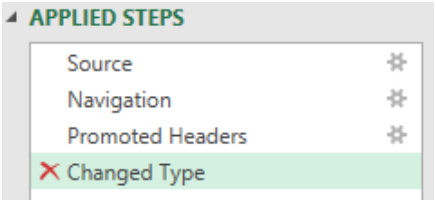
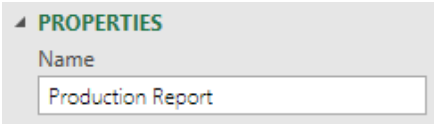
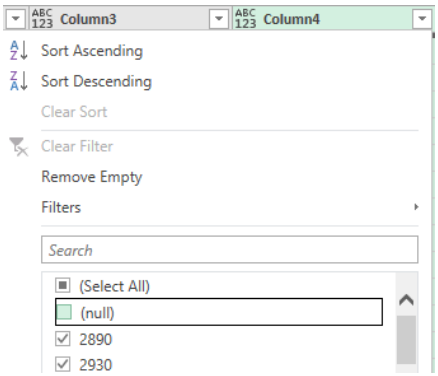
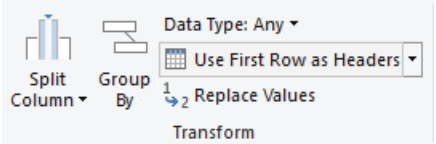
Move to the next sheet	Select **L03 Text to Columns**.	
Query the dataset	Click **Data Tab > Get Data > From File > From Text/CSV**.	
Select the file	Navigate to **M02-3 Shop Attendance**. Click **Transform Data**	
Split a column	Select the **Department-Team** column. Click **Home Tab > Split Column**. Choose **By Delimiter**.	
Specify the delimiter	The delimiter is the dash with spaces on either side, so choose **Custom** and type " - " into the textbox. Click **OK**.	
Rename columns	You have two columns named **Department-Team.1** and **Department-Team.2**. Rename them to *Department* and *Team*.	
Split Attendance	Recognize that you don't want the full attendance on each shift for the raw data; split the **Attendance** column **By Delimiter**.	
Set the delimiter	The custom delimiter is the semicolon *and* the space, so choose **Custom** and type "; " into the textbox.	

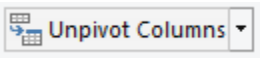
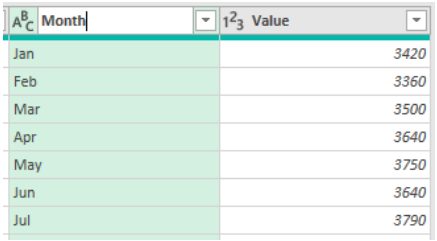
Split into rows	Now, click the **Advanced options drop-down arrow** and choose **Rows**. Click **OK**.	
Remove duplicates	Select all the columns of the dataset and click **Home Tab > Remove Rows > Remove Duplicates**.	
Load the data	Click **Home Tab > Close & Load > Close & Load To...** and choose the appropriate cell location.	

Department	Team	Shift	Date	Attendance
Production	Product A	First Shift	1/1/2020	Porter, Patrick
Production	Product A	First Shift	1/1/2020	Crosby, Leon
Production	Product A	First Shift	1/1/2020	Jones, Billie
Production	Product A	First Shift	1/1/2020	Boone, Janice
Production	Product A	First Shift	1/1/2020	Macias, Sharon
Production	Product A	Second Shift	1/1/2020	Casey, Arthur
Production	Product A	Second Shift	1/1/2020	Dubois, Helene
Production	Product A	Second Shift	1/1/2020	Parker, Grace
Production	Product A	Second Shift	1/1/2020	Ribeiro, Eduardo
Production	Product A	Third Shift	1/1/2020	Carrillo, Mauricio
Production	Product A	Third Shift	1/1/2020	Ho, Fang
Production	Product A	Third Shift	1/1/2020	Abo, Mariam
Production	Product A	Third Shift	1/1/2020	Bello, Molly
Production	Product A	Third Shift	1/1/2020	Simmons, Michael
Production	Product B	First Shift	1/1/2020	Rey, Micaela

Split Columns in Power Query is superior to Text to Columns in Excel for many reasons; in this exercise, we saw custom, multi-character delimiters, and the ability to split into *rows*.

Exercise 04: Unpivot Columns and Fill Down

Move to the next sheet	Select **L04 Unpivot and Fill**.	
Query the appropriate dataset	Click **Data Tab > Get Data > From File > From Workbook**.	
Find the data	Select **M02-4 2020 Production Report**.	
Get the appropriate sheet	Select **Sheet1**. Click **Transform**	
Un-promote	Delete the steps **Changed Type** and **Promoted Headers** from the **Applied Steps** panel.	
Rename query	Enter the name for the query **Production Report**.	
Remove unnecessary rows	Click the drop-down on **Column4** and filter **(null)** entries.	
Promote headers	Click **Home Tab > Use First Row as Headers**.	

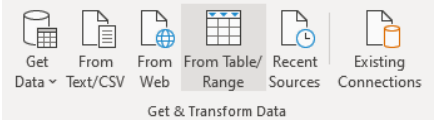
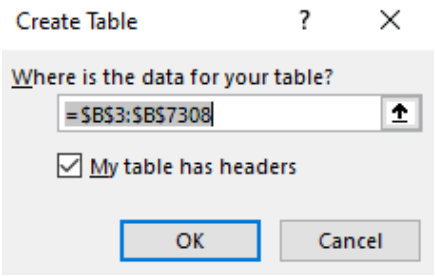
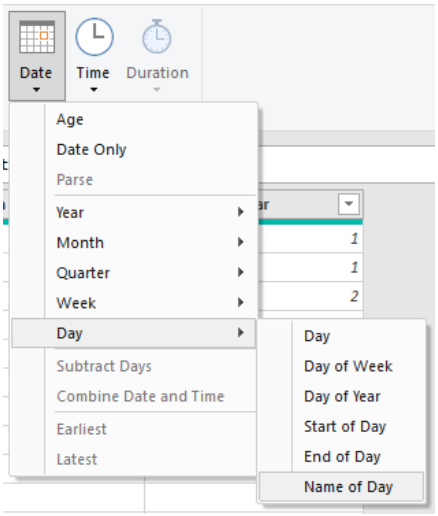
Fill down	Select the **Facility** column and choose **Transform > Fill > Down**.	
Unpivot monthly values	Click column **Jan**, hold **[Shift]**, and select column **Dec**. Click **Transform Tab > Unpivot Columns**.	
Rename columns	You should now have two columns, *Attribute* and *Value*. Rename them to **Month** and **Amount**.	
Load the data	Click **Home Tab > Close & Load > Close & Load To..** and choose the appropriate location.	

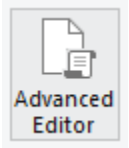
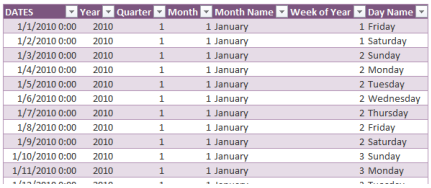
Power Query converts thousands of rows of data in seconds, solving otherwise impossible problems!

Module 03: Solving Problems with M Functions

Similarly to Excel, there are several categories of functions you can use to create new, calculated information. These functions are in a language called M.

Exercise 01: Date Functions

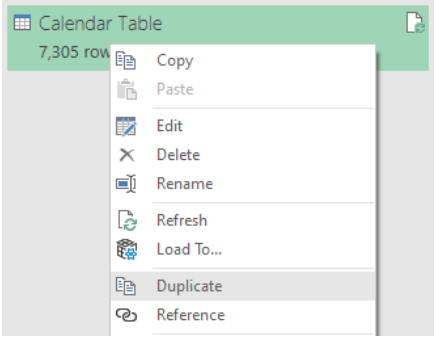
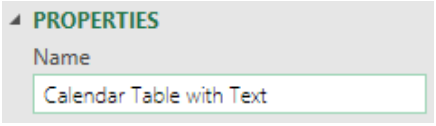
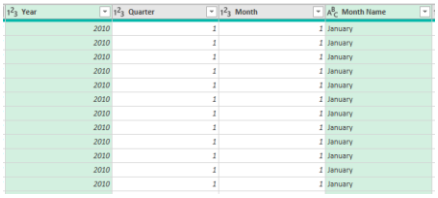
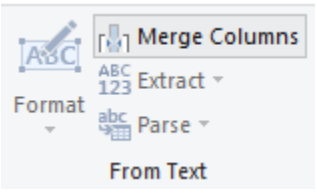
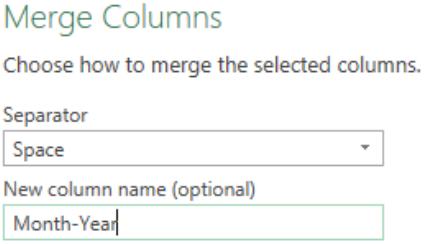
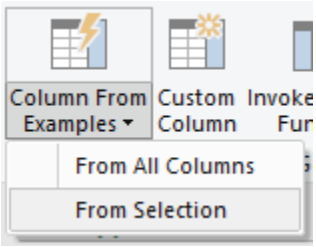
Open the working file	Find and open **M03 Using M Functions.xlsx** to the tab **L01 Date Functions**.	
Load the data into Power Query	Click cell **B5** (within the column of dates), and click **Data Tab > From Table/Range**.	
Convert the data into a table	In Excel, data being used by Power Query must first be converted into a Table, so click **OK** to the window that opens.	
Calculate conventional date information	Select the **DATES** column, and choose the following items from **Add Column Tab > Date**: • Year • Quarter of Year • Month • Name of Month • Week of Year • Name of Day	

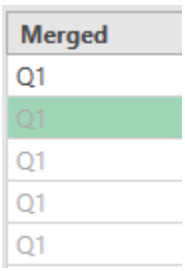
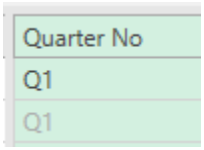
Investigate the calculations	Click **View Tab > Advanced Editor**, and look at the M functions written on your behalf.	
Rename query	Rename the query to **Calendar Table**	
Load data	Click **Home Tab > Close & Load > Close & Load To...** and choose the appropriate location.	

DATES	Year	Quarter	Month	Month Name	Week of Year	Day Name
1/1/2010 0:00	2010	1	1	January		Friday
1/2/2010 0:00	2010	1	1	January		Saturday
1/3/2010 0:00	2010	1	1	January		Sunday
1/4/2010 0:00	2010	1	1	January		Monday
1/5/2010 0:00	2010	1	1	January		Tuesday
1/6/2010 0:00	2010	1	1	January		Wednesday
1/7/2010 0:00	2010	1	1	January		Thursday
1/8/2010 0:00	2010	1	1	January		Friday
1/9/2010 0:00	2010	1	1	January		Saturday
1/10/2010 0:00	2010	1	1	January		Sunday
1/11/2010 0:00	2010	1	1	January		Monday

Power Query has several date functions to use for any valuable calendar-based information.

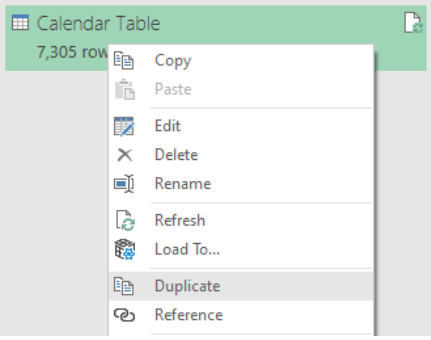
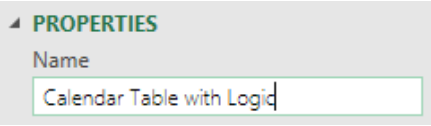
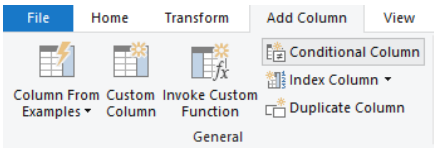
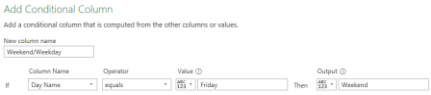
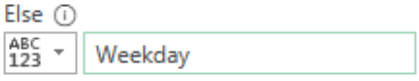
Exercise 02: Text Functions

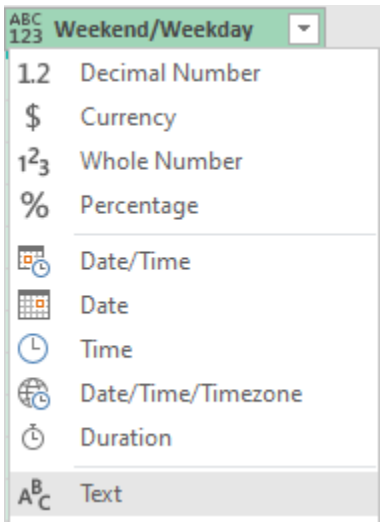
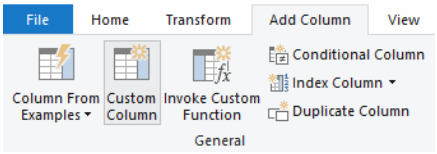
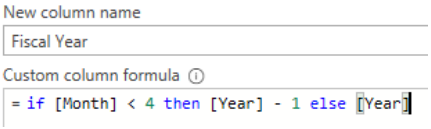
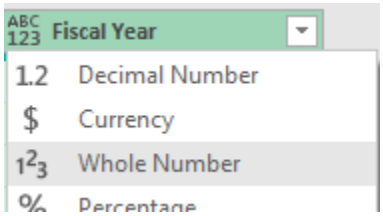
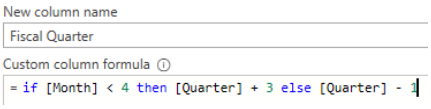
Duplicate the query	In the **Queries & Connections** panel, **right-click** the Calendar Table query and choose **Duplicate**.	
Rename query	Note that you now have both the **Calendar Table** and **Calendar Table (2)**. Rename the second query **Calendar Table with Text**.	
Merge columns	First, select the column **Month Name**. Then, hold **[CTRL]** and select the column **Year**. Power Query remembers the order of selection.	
Perform merge operation	Click **Add Columns Tab > Merge Columns**.	
Choose delimiter	Place the **Space** delimiter between the columns, and name the new column ***Month-Year***.	
Use Column from Examples	Select the **Quarter** column, and click **Add Column > Column from Examples > From Selection**.	

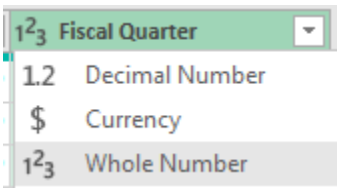
Type the quarter	Into the new column, type **"Q1"** (matching the *1* in the Quarter column). Press **[ENTER]**.	
Note the function	Above the dataset, Power Query shows you the function it is building to create Q1.	
Rename the column	**Double-click** the text **Merged** and type **Quarter No.**	
Press OK	**Click OK**, and note the new column added.	
Investigate the functions	Click **View Tab > Advanced Editor**, and note the functions that are created that: 1. Convert number values to text; 2. Concatenate text and numbers	`Text.Combine({[Month Name], Text.From([Year],`
Load the data to the page	Click **Home Tab > Close & Load > Close & Load To..** and choose the appropriate location.	

Text functions can merge and split text; however, if you are dealing with content that is numeric, M functions are necessary to convert numbers into text before calculation.

Exercise 03: Logical Functions

Duplicate the calendar table	**Right-click the Calendar Table** in the Queries & Connections panel, and choose **Duplicate.**	
Rename the query	Name the new query **Calendar with Logic**.	
Create a Conditional Column	Click **Add Column > Conditional Column**.	
Build a logical test for weekends	If we assume a 4-day workweek, with Friday-Sunday off, we can create a three-clause custom column: • Name: ***Weekend/Weekday*** • If: ***Day Name*** equals ***Friday*** then ***Weekend***	
Add a clause	Click **Add Clause**	
Add test	Else If: ***Day Name*** equals ***Saturday*** then ***Weekend***	
Add a clause	Click **Add Clause**	
Add test	Else If: ***Day Name*** equals ***Sunday*** then ***Weekend***	
Provide the final clause	Else: ***Weekday***	

Complete column	Click **OK**. Then, click the data type drop-down for the column and choose **Text**.	
Create another logical column	If your logical column requires additional calculations, you will need to create a Custom Column instead of a Conditional Column. They can both contain If-Then-Else logic. Click **Add Column Tab > Custom Column**.	
Calculate Fiscal Year	Assume we are dealing with a Fiscal Year that begins April 1. If the Month (number) is less than 4, our Fiscal Year will be our calendar year – 1. `= if [Month] < 4 then [Year] - 1 else [Year]`	
Set data type	From the Fiscal Year data type drop-down, choose **Whole Number**.	
Calculate Fiscal Quarter	With a Fiscal Year beginning April 1, our logic will be: `= if [Month] < 4 then [Quarter] + 3 else [Quarter] - 1`	

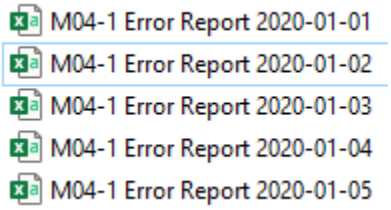
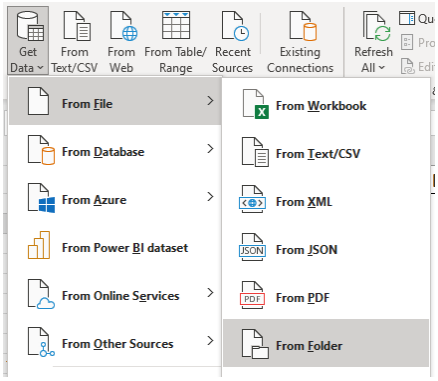
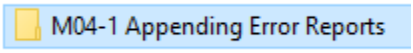
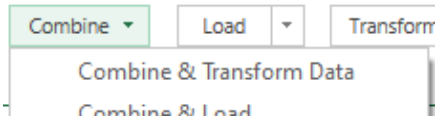
Set data type	From the Fiscal Quarter data type drop-down, choose **Whole Number**.	
Load data	Click **Home Tab > Close & Load > Close & Load To...**, and select the appropriate location.	

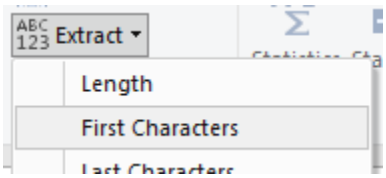
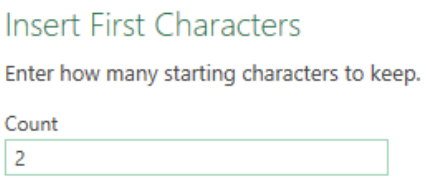
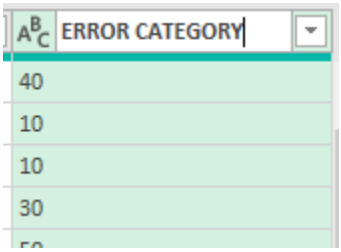
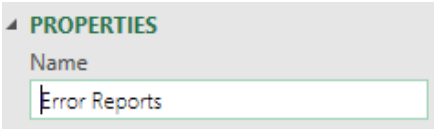
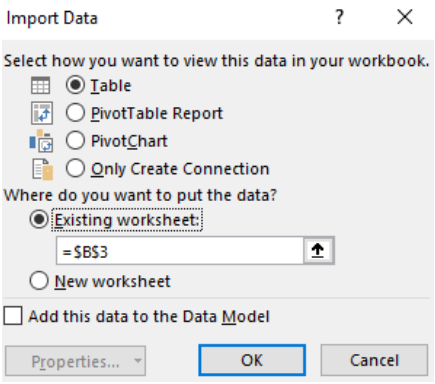
With Conditional Columns, we can judge various logical tests. However, if the output of our function must be calculated (like adding or subtracting from year, quarter, or month), then we must use the more flexible **Custom Column**.

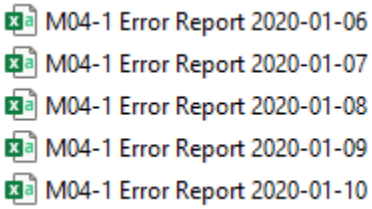
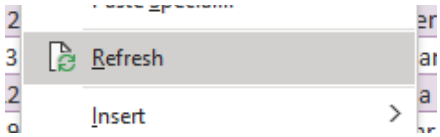
Module 04: Merging and Appending

In this module, we will both *merge* and *append* data tables. Appending tables means creating a single, longer table from similar data – for example, compiling all the expenses from various employees. Merging tables, however, means comparing and combining related tables – for example, combining employee data with office data.

Exercise 01: Append Files from a Folder

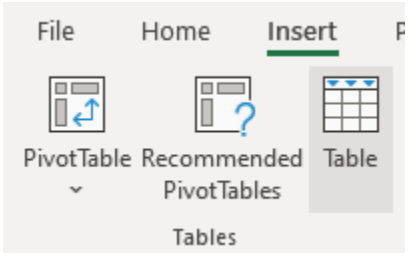
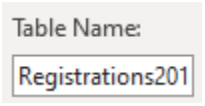
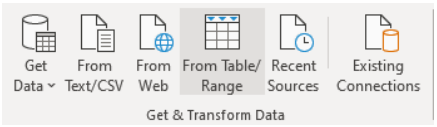
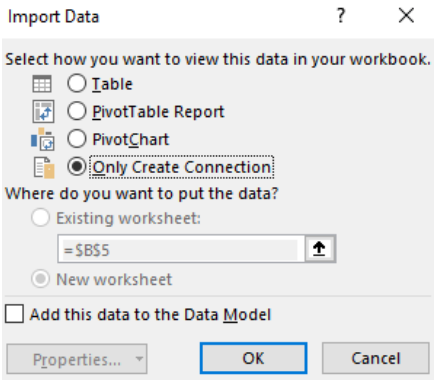
Open the working file	Open **M04 Joining and Appending Data.xlsx** to the tab **L01 Append from Folder**.	
Recognize the scenario	Look at both the Working Files folder and the **M04-1 Appending Error Reports** folder. Note that there are CSV files for every day Jan 1 – Jan 10. It will be our responsibility to compile a list of all errors as the reports come in.	
Query the folder	Click **Data Tab > Get Data > From File > From Folder**.	
Select the folder	Select **M04-1 Appending Error Reports**	
Combine the files	Investigate the first dialog box, and note that there are five CSV files to be combined currently. Click **Combine > Combine & Transform Data**.	
Guarantee the files have similar structure	Click the drop-down for **Sample File**, and note that you can preview each of the CSV files. Guarantee they all have the same columns and types of data. Click **OK**.	

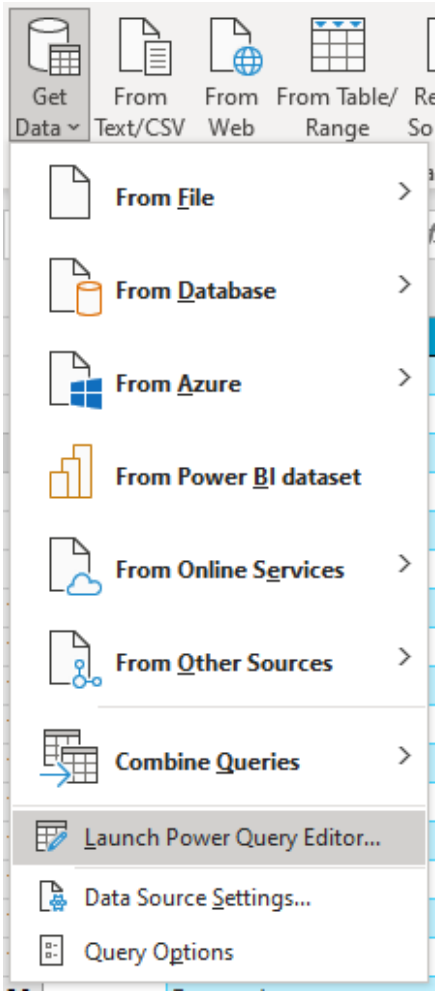
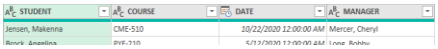
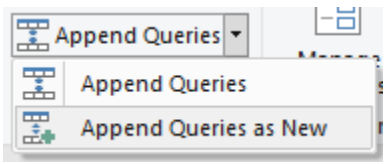
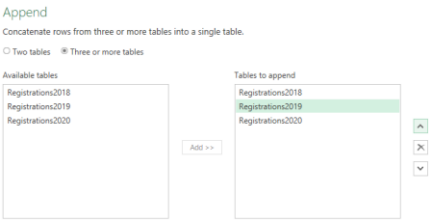
Create an additional column	Select the **ERROR CODE** column. Click **Add Column Tab > Extract > First Characters**. (we could also choose Before Delimiter)	
Pull the first two characters	Select the count of **2** for the Insert First Characters dialog.	
Rename column	Rename the column **First Characters** to ***ERROR CATEGORY***.	
Rename query	Rename the query **Error Reports**.	
Load the data	Click **Home Tab > Close & Load > Close & Load To...** and choose the appropriate location.	

Add more files	One at a time, **drag-and-drop** the CSV files **M04-1 Error Report 2020-01-06.csv**, etc. into the folder.	
Refresh the query	**Right-click** the table full of error reports and choose **Refresh** to see the latest data included in the table.	

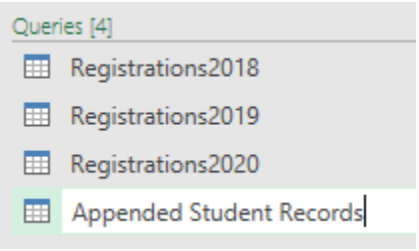
Power Query is capable of continuing to extract data from additional files, as the contents of the folder change. If you remove a file from this folder, the data will not be included in the results any longer.

Exercise 02: Append Tables from a File

Navigate to the appropriate sheet	Select **L02 Append Tables**.	
Convert the three lists into tables	Click cell **B4** (the 2018 course registrations), and choose **Insert Tab > Table**. On the **Design Tab**, click within the **Table Name** box and add the name **Registrations2018**. Repeat for 2019 and 2020. (These could easily be on separate tabs, but we've included them on one sheet for simplicity)	 
Query all three tables	In succession, select the 2018, 2019, and 2020 registration tables, and click **Data Tab > From Table/Range**. Then, when it is open in the Power Query Editor, click **Home Tab > Close & Load > Close & Load To...** Choose to load these three tables as **Only Create Connection**.	 

Return to the Power Query Editor	Click **Data Tab > Get Data > Launch Power Query Editor…**	
Note and fix differences	Notice that Registrations2020 has two columns named differently – rename **STUDENT** to **EMPLOYEE** and **MANAGER** to **TEAM LEADER** (or rename the 2018 and 2019 data to match 2020, alternatively).	
Append the three tables	Click **Home Tab > Append Queries > Append Queries as New**.	
Choose the appropriate tables	Select the 2018, 2019, and 2020 data to be appended (3 or more tables).	

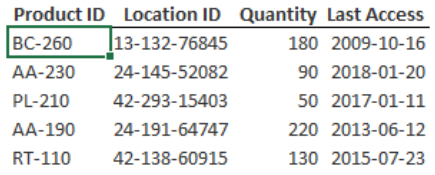
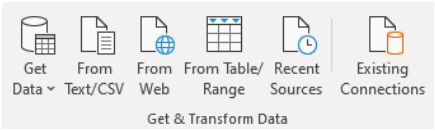
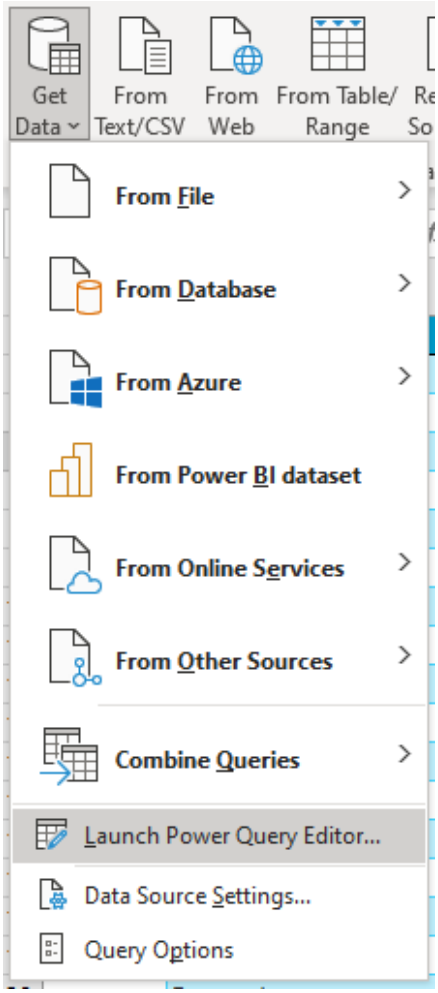
Rename the query	Change the query name from **Append1** to **Appended Student Records**.

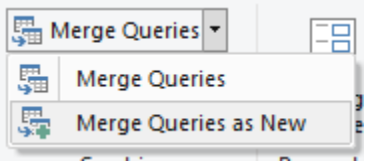
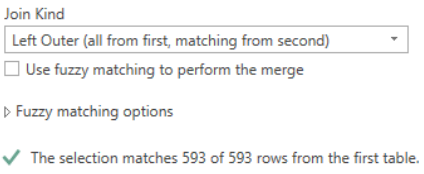


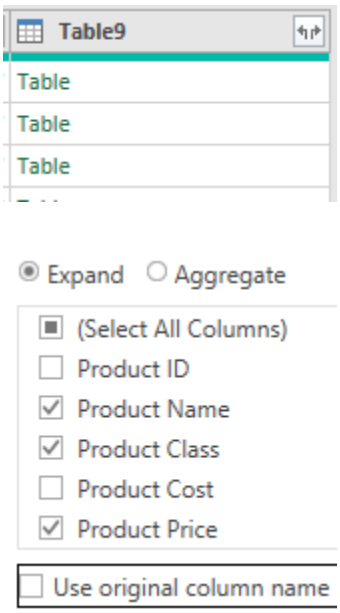
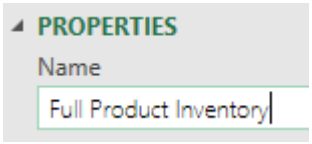
Load the query to the page	Click **Home Tab > Close & Load > Close & Load To...** and choose the appropriate location.

Any queries that Power Query is controlling can be appended, but the structures of the data must match.

Exercise 03: Outer Join

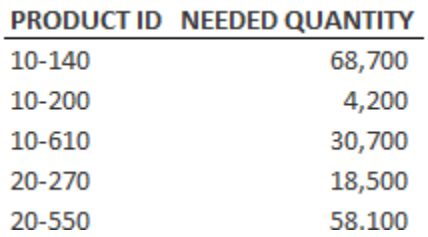
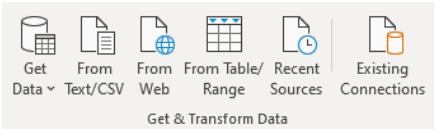
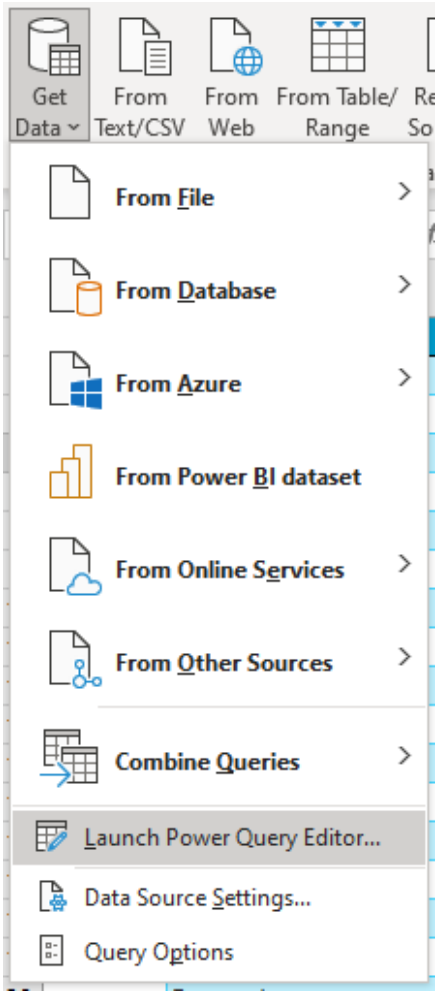
Navigate to the appropriate sheet	Select **L03 Outer Joins**	
Query the two tables	Click cell **B4** (Product Inventory list), and choose **Data Tab > From Table/Range**. Once that data is queried, load the data as a **Connection**, and repeat for cell **G4** (Product information list).	 
Return to the Power Query Editor	Click **Data Tab > Get Data > Launch Power Query Editor…**	

Join the tables	Select the inventory table and click **Home Tab > Merge Queries > Merge Queries as New**.	
Establish the relationship	From the first table, select the **Product ID** column. From the second table, select the **Product ID** column**.**	Table8 **Product ID** | **Location ID** BC-260 | 13-132-76845 AA-230 | 24-145-52082 PL-210 | 42-293-15403 AA-190 | 24-191-64747 RT-110 | 42-138-60915 Table9 **Product ID** | **Product Name** AA-100 | Single-Fan 4V AA-120 | Single-Fan 6V AA-150 | Single-Fan 8 V AA-170 | Single-Fan 2x 4V AA-190 | Single-Fan 2x 6V
Identify the success and type of join	Note that Join Kind has been set to **Left Outer**. This is the equivalent of VLOOKUP – finding matches for the first table in the second table, and using those matches to blend them into a more detailed single table. Note that Power Query informs you that it has found matches for all 593 entries.	

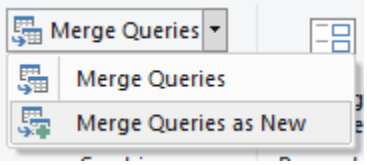
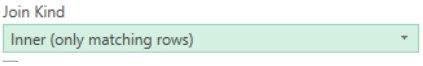
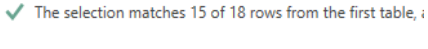
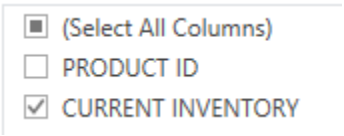
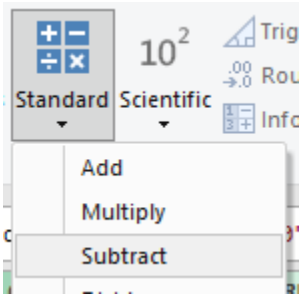
Complete the join	Click **OK**. Then, note that the second table has been added as a column to the first table. Click the arrow button to expand and include the **Product Name**, **Product Class**, and **Product Price**.	
Rename the query	Call the new query **Full Product Inventory**.	
Load the query to the page	Click **Home Tab > Close & Load > Close & Load To...** and choose the appropriate location.	

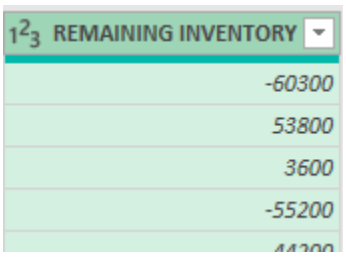
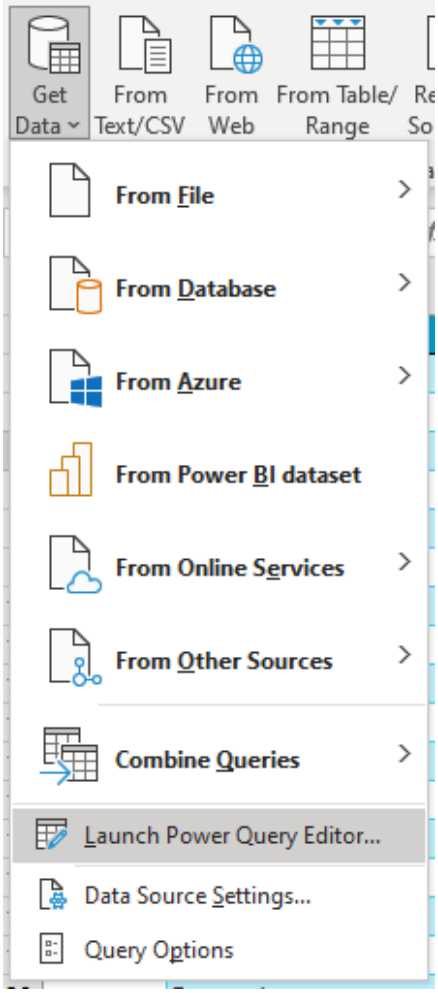
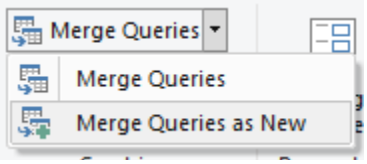
While VLOOKUP can be difficult to maintain and guarantee accuracy, Power Query joins are automatically updated and guaranteed.

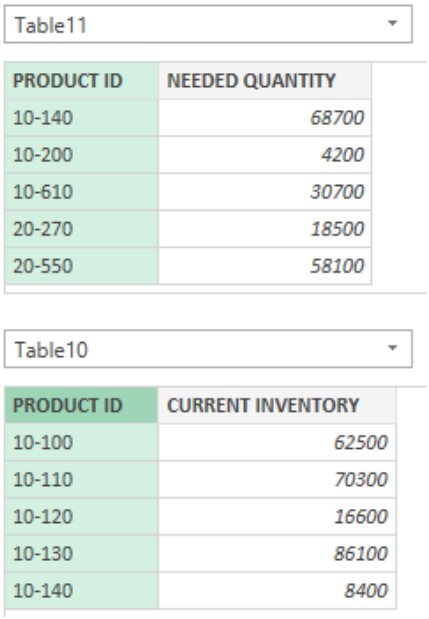
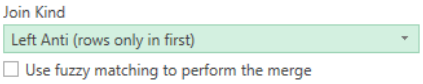
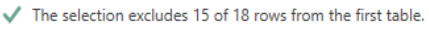
Exercise 04: Inner Joins

Navigate to the appropriate sheet	Find **L04 Inner and Anti**.	
Assess the scenario	We have an inventory of each product, and quantities needed. There are three possible situations: 1. There is a listing with adequate inventory for the need. 2. There is a listing with *inadequate* inventory for the need. 3. There is no listing for the needed product	
Query both datasets	Click **B4** and choose **Data Tab > From Table/Range**. Then load to a **Connection**. Repeat for the dataset in **F4**.	
Return to the Power Query Editor	Click **Data Tab > Get Data > Launch Power Query Editor...**	

PRODUCT ID	NEEDED QUANTITY
10-140	68,700
10-200	4,200
10-610	30,700
20-270	18,500
20-550	58.100

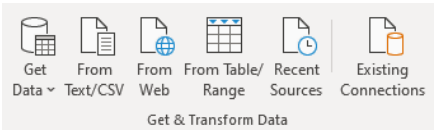
Create an inner join	Select the table with the needs in the Power Query editor. Click **Home Tab > Merge Queries > Merge Queries as New**.	
Establish Relationship	In the first table, select **Product ID** as the column to match, and choose the same column in the second table.	Table11 PRODUCT ID / NEEDED QUANTITY 10-140 68700 10-200 4200 10-610 30700 20-270 18500 20-550 58100 Table10 PRODUCT ID / CURRENT INVENTORY 10-100 62500 10-110 70300 10-120 16600 10-130 86100 10-140 8400
Switch the join	Change the Join Kind to **Inner**, which will only keep those entries that match between the two tables.	
Note the effect	This should inform you that there are matches for 15 of 18 entries – the other three will be excluded from the results.	
Complete the merge	Click **OK**. Then, from the new column, expand to include the **CURRENT INVENTORY**.	
Create a calculated column	Select the **CURRENT INVENTORY** column, then hold **[CTRL]** and select the **NEEDED QUANTITY**. Choose **Add Column Tab > Standard > Subtract**.	

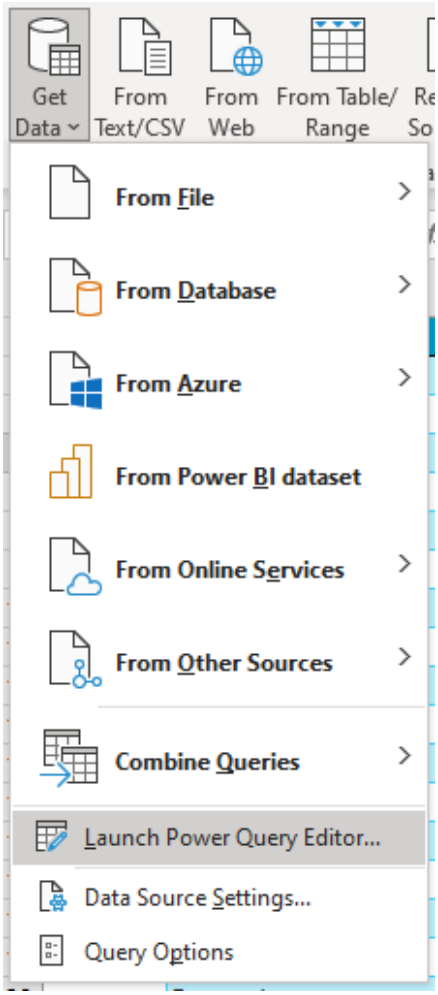
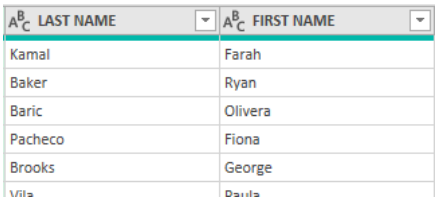
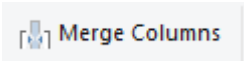
Rename column	Name the Subtraction column **Remaining Inventory**.	
Load the query to the page	Click **Home Tab > Close & Load > Close & Load To...** and choose the appropriate location.	
Return to the Power Query Editor	Click **Data Tab > Get Data > Launch Power Query Editor...**	
Create an Anti Join	Select the table with the needs in the Power Query editor. Click **Home Tab > Merge Queries > Merge Queries as New**.	

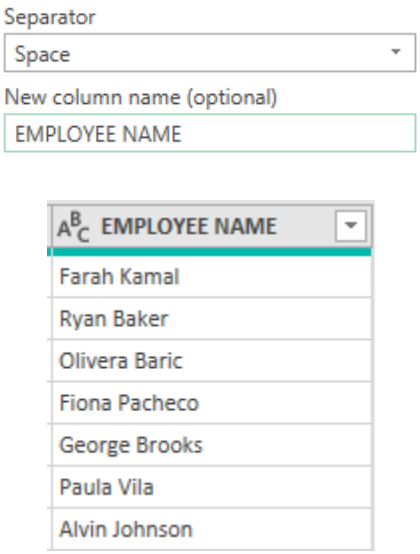
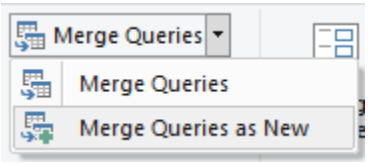
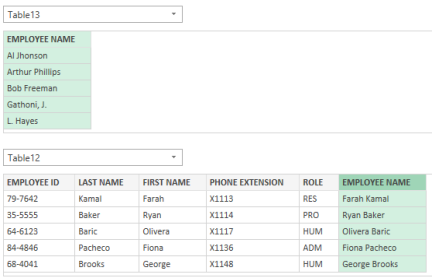
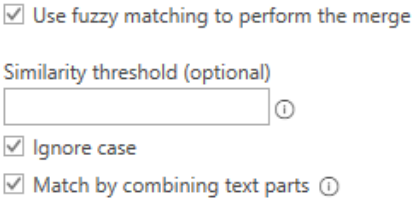
Establish Relationship	In the first table, select **Product ID** as the column to match, and choose the same column in the second table.	
Switch the join	Change the join requested to **Left Anti** (keeping only unmatched entries from the main table).	
Note the effect	Power Query should inform you that only 3 of the 18 lines remain (others have been excluded).	
Load the query to the page	Click **Home Tab > Close & Load > Close & Load To...** and choose the appropriate location.	

Inner and Anti Joins help you see the relationship between tables by either showing the matches or the unmatched entries.

Exercise 05: Fuzzy Joins

Navigate to the appropriate sheet	Find **L05 Fuzzy Joins**	
Note the scenario	We have sloppy data here. Employee names have been entered in every conceivable configuration, none of which include the easier-to-match Employee ID. In fact, some of these names are even misspelled. This would typically require quite a lot of re-sorting and scrolling to try to find all relevant matches.	
Query the two tables	Click cell **B4** (Employee list), and choose **Data Tab > From Table/Range**. Once that data is queried, load the data as a **Connection**, and repeat for cell I**4** (shorter list of employees).	

Return to the Power Query Editor	Click **Data Tab > Get Data > Launch Power Query Editor…**	
Choose the master table	Select the table that includes the full Employee List.	
Create a useful column	Select the **FIRST NAME** and **LAST NAME** columns, then choose **Add Column Tab > Merge Columns**.	

Establish the new column	Choose a **space** between the first and last names, and create a column called **Employee Name**.	
Merge the queries	Select the shorter employee list. Click **Home Tab > Merge Queries > Merge Queries as New**.	
Establish the relationship	Select the **EMPLOYEE NAME** columns from both tables to establish the match.	
Set the fuzzy match	Click the checkbox **Use fuzzy matching to perform the merge**. Also, set the Similarity threshold to **.5**.	

Expand the match	On the new column, click the expand button and include the new data from the second table.