4. **Change Data Types**:
    - Click the icon in the column header
    - Or: select column → Transform tab → Any Column → Data Type
    - Common types: Text, Number, Date, True/False

Data Cleaning Operations

1. **Replace Values**:
    - Select column → Transform tab → Replace Values
    - Enter value to find and replacement value
    - Example: Replace "N/A" with null
2. **Remove Errors**:
    - Select column → Home tab → Reduce Rows → Remove Errors
    - Or: filter column to exclude errors
3. **Remove Duplicates**:
    - Select column(s) → Home tab → Reduce Rows → Remove Duplicates
    - Can be based on one or multiple columns
4. **Fill Down/Up**:
    - Select column → Transform tab → Fill → Down/Up
    - Fills empty cells with value from above/below

Structural Transformations

1. **Pivot Column**:
    - Transform tab → Pivot Column
    - Converts rows to columns
    - Great for transforming survey data or cross-tabulations
2. **Unpivot Columns**:
    - Select columns → Transform tab → Unpivot Columns
    - Converts columns to rows
    - Excellent for fixing "wide" data formats
3. **Group By**:
    - Home tab → Group By
    - Aggregates data by categories
    - Similar to creating a pivot table in Excel
4. **Add Custom Column**:

- Add Column tab → Custom Column
- Create calculated fields using M formula language

Understanding Applied Steps

Each transformation you perform creates a "step" in the Query Settings pane. This is crucial to understand because:

- Steps run in sequence from top to bottom
- You can select any step to see how your data looked at that point
- You can rename steps for clarity (right-click → Rename)
- You can delete steps you no longer need
- You can insert new steps between existing ones
- You can disable steps temporarily (right-click → Enable/Disable)

This step-by-step approach makes it easy to track changes and fix issues.

M Formula Language

Power Query uses a language called "M" to define transformations. While you don't need to know M to use Power Query, understanding the basics helps:

- Each step generates M code automatically
- You can view the code in the formula bar
- You can edit the code directly for advanced transformations
- M is case-sensitive and uses "#" for comments

Simple M example for a custom column that concatenates first and last names:

```
= [FirstName] & " " & [LastName]
```

Real-World Example

Imagine you receive a monthly sales report export with these issues:

- Column names have extra spaces
- Date field is in text format (MM/DD/YYYY)
- Product codes contain a mix of uppercase and lowercase
- Some rows have "NULL" as text instead of actual null values
- The data has multiple header rows

In Power Query, you can clean this by:

1. Remove extra header rows (Home → Reduce Rows → Remove Rows → Remove Top Rows)
2. Use first row as headers (Home → Transform → Use First Row as Headers)
3. Rename and trim column names (right-click → Rename)
4. Convert date column to Date type (click column data type icon → Date)
5. Standardize product codes (select column → Transform → Format → UPPERCASE)
6. Replace "NULL" text with actual null values (Transform → Replace Values)

Now every month when you get new data, simply refresh and all these cleaning steps happen automatically!

Step-by-Step: Basic Data Cleaning

Let's clean a typical sales dataset:

1. **Open Power Query Editor**:
   - Home tab → Transform Data
2. **Fix Column Names**:
   - Right-click the column header → Rename
   - Or: Transform tab → Transform → Format → Capitalize Each Word
3. **Change Data Types**:
   - Set date columns to Date type
   - Set numerical columns to appropriate number types (whole number, decimal, etc.)
   - Set text columns to Text type
4. **Remove Unnecessary Columns**:
   - Select columns you don't need → right-click → Remove
   - Or: Home tab → Manage Columns → Remove Columns
5. **Check and Apply Changes**:
   - Review the preview to make sure data looks correct
   - Click "Close & Apply" in the Home tab to apply changes and return to Power BI

Practice Exercise: Power Query Basics

Objective: Use Power Query Editor to clean and transform a dataset.

Prerequisites: Import a new file called "MessySalesData.xlsx" (your instructor will provide this file with intentional errors).

Tasks:

1. **Open Power Query Editor**:

- After importing the file, click "Transform Data" in the Navigator
- Or if already imported, select Home → Transform Data

2. **Clean Column Names**:
   - Rename columns to follow a consistent naming convention
   - Remove any unnecessary columns
3. **Fix Data Types**:
   - Set appropriate data types for each column
   - Pay special attention to dates, numbers, and currency fields
4. **Clean Data Values**:
   - Find and replace any text versions of NULL or N/A with actual null values
   - Fix any obvious errors in text fields (misspellings, inconsistent capitalization)
5. **Document Your Process**:
   - Rename each step in the Applied Steps pane with descriptive names
   - Take notes on what issues you found and how you fixed them

Bonus Challenge: Create a custom column that combines information from two or more existing columns (like a full name from first and last names).

Removing Errors, Filtering, Splitting Columns

Advanced Data Cleaning Techniques

While basic transformations help with simple issues, real-world data often requires more advanced cleaning techniques. In this module, we'll focus on three powerful methods: handling errors, filtering data, and splitting columns.

Identifying and Handling Errors

Errors in Power Query appear as red text with the word "Error" in cells. They occur when:

- Data can't be converted to the assigned type
- Calculations encounter problems (like division by zero)
- Functions receive invalid parameters

Ways to Handle Errors

1. **Remove Errors**:
   - Select the column with errors
   - Home tab → Reduce Rows → Remove Errors
   - Best when errors represent invalid data you want to exclude

2. **Replace Errors**:
   - Select the column with errors
   - Transform tab → Replace Errors
   - Enter a replacement value (like 0, null, or "Unknown")
   - Useful when you need to keep the rows but fix the problematic values
3. **Convert to Null**:
   - Transform tab → Replace Errors → (leave replacement value empty)
   - Good middle-ground approach
4. **Error Inspection**:
   - Filter column to show only errors
   - Try to understand the pattern causing errors
   - Fix the root cause rather than just the symptoms

Filtering Data

Filtering lets you focus on specific portions of your data or remove unwanted records.

Filter Types in Power Query

1. **Basic Filters**:
   - Click the drop-down arrow in column header
   - Select/deselect values to include/exclude
   - Great for categorical data with few unique values
2. **Text Filters**:
   - Select "Text Filters" from dropdown
   - Options include:
     - Begins With/Ends With
     - Contains/Does Not Contain
     - Equals/Does Not Equal
     - Custom Filter (combine multiple conditions)
3. **Number Filters**:
   - Select "Number Filters" from dropdown
   - Options include:
     - Equals/Does Not Equal
     - Greater Than/Less Than
     - Between

        - Top N/Bottom N

4. **Date Filters**:
    - Select "Date Filters" from dropdown
    - Options include:
        - Before/After/Between
        - In the Past/In the Next
        - Date range options (This Month, Last Year, etc.)

5. **Advanced Filter**:
    - Add Column tab → Custom Column
    - Create a formula that returns true/false
    - Then filter to show only "true" values

Filter Locations

- **Column Header Filters**: Apply to individual columns
- **Keep/Remove Rows**: Home tab → Reduce Rows
- **Filter Rows**: Select rows → right-click → Keep/Remove Rows

Splitting Columns

Splitting columns is essential when dealing with data that combines multiple pieces of information in a single field.

Common Split Scenarios

- **Full names** into first and last names
- **Addresses** into street, city, state, and zip
- **Product codes** that combine category and number
- **Date-times** that need separate date and time components

Splitting Methods

1. **Split by Delimiter**:
    - Select column → Transform tab → Split Column → By Delimiter
    - Choose the character that separates values:
        - Common delimiters: comma, space, tab, semicolon
        - Custom delimiter option for other characters
    - Choose split options:
        - Left-most delimiter (splits at first occurrence)

    - Right-most delimiter (splits at last occurrence)
    - Each occurrence (creates multiple columns)

2. **Split by Number of Characters**:
    - Select column → Transform tab → Split Column → By Number of Characters
    - Specify how many characters per column
    - Useful for fixed-width data formats

3. **Split by Positions**:
    - Select column → Transform tab → Split Column → By Positions
    - Specify exact positions where splits should occur
    - Good for complex or irregular patterns

4. **Advanced Splits using Custom Columns**:
    - Add Column tab → Custom Column
    - Use Text.Start, Text.Middle, Text.End functions
    - Example: `Text.Start([FullName], 1)` for first initial

Handling Split Results

After splitting columns:

- Rename the resulting columns with meaningful names
- Set appropriate data types
- Consider removing the original column if no longer needed

Real-World Example

Imagine you have customer data with these issues:

1. **Combined Name Column**:
    - Original: "Smith, John A."
    - Needed: Last Name = "Smith", First Name = "John", Middle Initial = "A"

2. **Invalid Phone Numbers**:
    - Some entries have letters or are too short
    - These appear as errors when converted to phone number format

3. **Need Only US Customers**:
    - Data includes international customers you want to exclude

Using Power Query, you could:

1. **Split the Name**:

- Split by delimiter (comma) to separate last name
- Split the remainder by the last space to separate middle initial
- Rename columns appropriately

2. **Handle Phone Errors**:
   - Convert column to text type
   - Replace errors with "Invalid"
   - Or filter to remove rows with invalid phone numbers

3. **Filter for US Customers**:
   - Apply a filter on Country column to show only "USA" or "US"
   - Or use a contains filter on the Postal Code to match US format

Step-by-Step: Advanced Data Cleaning

Let's walk through cleaning a customer dataset:

1. **Handle Name Column**:
   - Select "FullName" column
   - Transform tab → Split Column → By Delimiter
   - Choose space as delimiter and split at the first occurrence
   - Rename resulting columns to "FirstName" and "LastName"

2. **Clean Phone Numbers**:
   - Select "Phone" column
   - Transform tab → Replace Values
   - Replace "-", "(", ")" and spaces with nothing to standardize format
   - Filter to show only values with 10+ characters (valid phone length)

3. **Extract Domain from Email**:
   - Select "Email" column
   - Transform tab → Split Column → By Delimiter
   - Choose "@" as delimiter
   - Rename second column to "EmailDomain"
   - You can now analyze which email providers are most common

4. **Fix Date Format Issues**:
   - Select "JoinDate" column
   - If errors appear when converting to Date type
   - Transform tab → Replace Errors → null

- Or create formula to handle different date formats

Practice Exercise: Advanced Data Cleaning

Objective: Apply advanced cleaning techniques to prepare data for analysis.

Prerequisites: Import a file called "CustomerData.xlsx" (your instructor will provide this).

Tasks:

1. **Handle Error Values**:
   - Identify columns with errors
   - For each error column, decide whether to:
     - Remove rows with errors
     - Replace errors with null or another value
     - Fix the root cause of the errors
2. **Apply Filters**:
   - Filter the data to include only:
     - Active customers (Status = "Active")
     - Customers who joined within the last 2 years
     - Customers with complete contact information
3. **Split Complex Columns**:
   - Find the "Address" column which contains full address in one field
   - Split it into components (Street, City, State, ZIP)
   - Set appropriate data types for each new column
4. **Create Clean Customer Dataset**:
   - Ensure all columns have appropriate names and data types
   - Remove any unnecessary columns
   - Document your cleaning process

Bonus Challenge: Create a custom column that categorizes customers based on their email domain (e.g., "Gmail User", "Business User", "Other").

Combining Data (Merge & Append)

The Power of Data Combination

In real-world scenarios, the data you need for analysis often lives in multiple files or tables. Power Query excels at combining data through two key operations:

1. **Merge**: Joins tables horizontally by matching related rows (like SQL JOIN)

2. **Append**: Combines tables vertically by stacking rows (like SQL UNION)

Understanding when and how to use these operations is essential for comprehensive data analysis.

Merging Tables

Merging combines two tables based on a common field or key, bringing columns from the second table into the first.

When to Use Merge

- Combine product details with sales data
- Add customer information to order history
- Link employee data to performance metrics
- Connect lookup tables with transaction data

Merge Types Explained

Power Query offers several merge types that determine which rows appear in the result:

1. **Left Outer (most common)**:
   - Keeps all rows from first table
   - Adds matching data from second table
   - Creates null values where no match exists
   - Use when: You need all records from primary table regardless of matches
2. **Right Outer**:
   - Keeps all rows from second table
   - Adds matching data from first table
   - Creates null values where no match exists
   - Use when: You need all records from secondary table regardless of matches
3. **Full Outer**:
   - Keeps all rows from both tables
   - Creates null values where no match exists
   - Use when: You need all records from both tables
4. **Inner**:
   - Keeps only rows that have matches in both tables
   - Use when: You only want complete records with data from both tables
5. **Left Anti**:
   - Keeps only rows from first table that don't match second table

- Use when: Finding exceptions or missing relationships

6. **Right Anti**:
   - Keeps only rows from second table that don't match first table
   - Use when: Finding exceptions or missing relationships

Step-by-Step Merge Process

Let's merge Sales data with Product details:

1. **Start the Merge**:
   - In Power Query Editor, select the table to merge into (e.g., Sales)
   - Home tab → Combine → Merge Queries
   - Select "Merge Queries" for in-place merge or "Merge Queries as New" to create a new query
2. **Select Tables and Join Columns**:
   - Your first table is pre-selected
   - Choose the second table from dropdown (e.g., Products)
   - Click column(s) that match in both tables (e.g., ProductID)
   - Select join type (e.g., Left Outer)
   - Click OK
3. **Work with Merged Data**:
   - A new column appears containing related rows from second table
   - Click the expand button (double arrow icon) in column header
   - Select which columns to bring in from second table
   - Uncheck "Use original column name as prefix" for cleaner naming
   - Click OK
4. **Finalize the Merge**:
   - Rename columns if needed
   - Set proper data types
   - Apply any additional transformations

Common Merge Issues and Solutions

1. **Data Type Mismatches**:
   - Issue: Join columns have different data types (e.g., text vs. number)
   - Solution: Make sure join columns have matching types before merging
2. **Duplicate Keys**:
   - Issue: Multiple matching rows create duplicate or expanded results

- Solution: Remove duplicates from key columns or use aggregation before merging

3. **Missing Matches**:
   - Issue: Expected matches are missing
   - Solution: Check for spaces, case differences, or formatting issues in join columns
4. **Performance**:
   - Issue: Large merges are slow
   - Solution: Filter tables before merging to include only necessary data

Appending Tables

Appending combines tables with similar structures by adding rows from one table to another.

When to Use Append

- Combine monthly data files into a single table
- Merge historical and current data with the same structure
- Integrate data from different sources with compatible columns
- Combine regional data into a company-wide view

Append Process

Let's append quarterly sales tables:

1. **Start the Append**:
   - In Power Query Editor, select one of the tables to append
   - Home tab → Combine → Append Queries
   - Select "Append Queries" for in-place append or "Append Queries as New" to create a new query
2. **Select Tables to Append**:
   - Choose append mode:
     - "Two tables" to append one table to another
     - "Three or more tables" to append multiple tables at once
   - Select tables to append from the dropdown(s)
   - Click OK
3. **Handle Column Differences**:
   - If tables have different columns:
     - Matching columns will align automatically
     - Non-matching columns will get null values where needed
   - Consider adding a source identifier (see next section)

4. **Finalize the Append**:
    - Set proper data types
    - Remove any duplicate rows if needed
    - Apply additional transformations

Tracking Data Sources

When appending data, it's often useful to track which source each row came from:

1. **Before Appending**:
    - For each source table, add a custom column
    - Add Column tab → Custom Column
    - Name it "Source" or "DataSource"
    - Enter a static value that identifies the table (e.g., "Q1 Sales", "2022 Data")
2. **After Appending**:
    - All source identifiers will be preserved in the combined table
    - You can filter or analyze by source as needed

Real-World Example

Imagine you're a business analyst who receives:

1. Monthly sales files (12 separate Excel files with identical structure)
2. A product catalog with details like cost, category, and supplier
3. A store locations file with address and region information

To create a comprehensive sales analysis:

1. **Append** all 12 monthly sales files into a single sales table
    - Add a "Month" column to each before appending
2. **Merge** the combined sales with product catalog
    - Left outer join on ProductID
    - Bring in Category, Cost, Supplier
3. **Merge** again with store locations
    - Left outer join on StoreID
    - Bring in Region, City, StoreSize

Now you have a complete dataset for analysis with:

- Sales transactions for the entire year
- Product details for each sale

- Store information for regional analysis

Step-by-Step: Combining Multiple Data Sources

Let's walk through combining regional sales data:

1. **Append Regional Sales Tables**:
   - Home tab → Combine → Append Queries as New
   - Select "Three or more tables"
   - Add all regional sales tables (North, South, East, West)
   - Name the new query "Combined Sales"
2. **Add Source Identification**:
   - Go back to each regional query
   - Add Column tab → Custom Column
   - Name: "Region"
   - Formula: "North" (adjust for each table)
   - Go back to Combined Sales and refresh
3. **Merge with Product Information**:
   - With Combined Sales selected
   - Home tab → Combine → Merge Queries
   - Select Products table
   - Join on ProductID columns
   - Choose Left Outer join
   - Expand to bring in needed product columns

Practice Exercise: Combining Data

Objective: Combine multiple related datasets to create a comprehensive analysis table.

Prerequisites: Import the following files (your instructor will provide these):

- "SalesQ1.xlsx", "SalesQ2.xlsx", "SalesQ3.xlsx", "SalesQ4.xlsx"
- "ProductCatalog.xlsx"
- "CustomerList.xlsx"

Tasks:

1. **Append Quarterly Sales Data**:
   - Create a unified sales table from all four quarterly files
   - Add a quarter identifier to each table before appending

- Ensure data types are consistent across all tables

2. **Merge with Product Catalog**:
   - Combine the sales data with product details
   - Use appropriate join type (consider what should happen with unknown products)
   - Bring in relevant product fields (description, category, cost)
3. **Merge with Customer Information**:
   - Add customer details to your combined table
   - Use appropriate join type
   - Bring in relevant customer fields (name, segment, region)
4. **Create Analysis-Ready Table**:
   - Ensure all columns have descriptive names
   - Set appropriate data types
   - Create any additional calculated columns needed for analysis
   - Document the data combination process

Bonus Challenge: Calculate how many sales transactions had missing product information or customer details, and determine if there's a pattern to these gaps.

Relationships and Model View

Understanding Data Relationships

In Power BI, relationships connect tables together, enabling cross-filtering between visuals based on different tables. A well-structured data model with proper relationships is the foundation of an effective Power BI report.

What is a Relationship?

A relationship in Power BI is a connection between two tables based on related columns. This tells Power BI how data in one table corresponds to data in another table.

For example:

- A Products table might have a ProductID column
- A Sales table also has a ProductID column
- A relationship connects these tables through their ProductID columns
- This lets you analyze Sales data by Product attributes

The Importance of Relationships

Without relationships:

- Tables exist in isolation
- You can't create visuals combining fields from different tables
- Filters from one table won't affect visuals based on another table

With proper relationships:

- Cross-filtering works automatically
- You can use fields from multiple tables in a single visual
- Your reports become truly interactive

Understanding Cardinality

Cardinality describes how records in one table relate to records in another. Power BI supports several relationship types:

1. **One-to-Many** (most common):
   - One record in the first table can relate to many in the second
   - Example: One Product can have Many Sales
   - The "one" side contains unique values (like a primary key)
   - The "many" side can contain duplicate values (like a foreign key)
2. **Many-to-One**:
   - Same as one-to-many but with table order reversed
   - Direction matters for filter flow in Power BI
3. **One-to-One**:
   - One record in the first table relates to exactly one in the second
   - Both columns contain unique values
   - Example: EmployeeID in both Employee and EmployeeDetails tables
4. **Many-to-Many**:
   - Multiple records in first table relate to multiple in second
   - Requires special handling in Power BI
   - Often implemented using an intermediate table

Model View in Power BI

Model view provides a diagram of your data model, showing tables and their relationships.

Accessing Model View

- Click the "Model view" icon in the left navigation bar (third icon)
- Or go to View tab → Model view

Model View Interface Elements

- **Tables**: Shown as boxes containing field lists
- **Relationships**: Shown as lines connecting tables
- **Cardinality**: Indicated by 1 or * (asterisk) symbols on relationship lines
- **Filter direction**: Shown by arrows on relationship lines

Creating and Managing Relationships

Automatic Relationship Detection

When you load multiple tables, Power BI tries to detect relationships automatically based on:

- Matching column names
- Compatible data types
- Uniqueness of values

However, auto-detection isn't perfect and often needs adjustment.

Creating Relationships Manually

1. **In Model View**:
   - Click and drag from a column in one table to the related column in another table
   - Configure relationship settings in the dialog box
2. **Using Manage Relationships Dialog**:
   - Home tab → Relationships → Manage Relationships
   - Click "New" to create a relationship
   - Select tables and columns to connect
   - Configure cardinality and other settings

Relationship Settings

When creating or editing a relationship, you'll see several options:

1. **Cardinality**:
   - Select the appropriate type (usually "Many to One")
2. **Cross-filter direction**:
   - Single: Filters flow from the "one" side to the "many" side only (default)
   - Both: Filters flow in both directions
   - Both is useful in specific scenarios but can cause performance issues
3. **Make this relationship active**:

- Only one active relationship can exist between two tables
- Inactive relationships can be used in specific DAX calculations

Common Relationship Scenarios

Fact-to-Dimension Relationships

The most common pattern in analytical models:

- **Fact table**: Contains business events or measurements (sales, orders)
- **Dimension tables**: Contain descriptive attributes (products, customers)
- **Relationship**: Many-to-one from fact to dimension tables
- Example: Sales (fact) related to Products, Customers, and Dates (dimensions)

Date Table Relationships

Almost every model needs a date table:

- Date tables contain calendar attributes (year, month, quarter)
- Create relationships between transaction dates and the date table
- This enables time intelligence functions and date-based filtering

Role-Playing Dimensions

When the same dimension serves multiple roles:

- Example: A Date table related to both OrderDate and ShipDate in an Orders table
- Solution: Create multiple relationships but only one active
- Use inactive relationships in specific measures with USERELATIONSHIP function

Troubleshooting Relationship Issues

Common Problems and Solutions

1. **"The relationship couldn't be created"**:
   - Check for data type mismatches between columns
   - Verify the "one" side contains unique values
   - Look for empty or null values in key columns
2. **Ambiguous paths between tables**:
   - Problem: Multiple paths between tables cause filter confusion
   - Solution: Make some relationships inactive or restructure your model
3. **Circular dependencies**:
   - Problem: Filter paths create a circular loop

- Solution: Break the circle by making one relationship inactive

4. **Missing relationships**:
   - Visuals combining tables don't show the right data
   - Create necessary relationships between tables

Real-World Example

Imagine you're analyzing a retail business with these tables:

1. **Sales**: Contains transaction details (ProductID, CustomerID, StoreID, Date, Quantity, Revenue)
2. **Products**: Contains product information (ProductID, Name, Category, Cost)
3. **Customers**: Contains customer data (CustomerID, Name, City, Segment)
4. **Stores**: Contains store information (StoreID, Location, Size)
5. **Calendar**: Contains date attributes (Date, Day, Month, Quarter, Year)

Your relationship model would have:

- Sales to Products: Many-to-One on ProductID
- Sales to Customers: Many-to-One on CustomerID
- Sales to Stores: Many-to-One on StoreID
- Sales to Calendar: Many-to-One on Date

With this model:

- You can analyze sales by product category
- You can filter all visuals by customer segment
- You can compare performance across store locations
- You can view trends over time periods

Step-by-Step: Building a Relational Model

Let's create a basic sales analysis model:

1. **Examine Your Tables**:
   - Switch to Data view to understand each table's structure
   - Identify primary keys (unique identifiers) in each table
   - Identify foreign keys (references to other tables)
2. **Switch to Model View**:
   - Click the Model view icon on the left navigation bar
3. **Create Basic Relationships**:
   - Click and drag from Sales[ProductID] to Products[ProductID]

- Click and drag from Sales[CustomerID] to Customers[CustomerID]
- Configure each as Many-to-One with single filter direction

4. **Check Relationship Status**:
   - Verify relationships appear with correct cardinality symbols
   - Test relationships by creating a visual with fields from different tables
5. **Add Date Relationship**:
   - Create or import a date table
   - Create relationship between Sales[Date] and Calendar[Date]

Practice Exercise: Building a Data Model

Objective: Create an effective data model with proper relationships between tables.

Prerequisites: Import the following tables (your instructor will provide these):

- "Orders" (contains OrderID, CustomerID, ProductID, Date, Quantity, Amount)
- "Customers" (contains CustomerID, Name, City, Country)
- "Products" (contains ProductID, ProductName, Category, UnitPrice)
- "Calendar" (contains Date and date attributes like Year, Month, Quarter)

Tasks:

1. **Examine Table Structure**:
   - Review each table in Data view
   - Identify primary keys and foreign keys
   - Check data types of key columns
2. **Create Relationships**:
   - Switch to Model view
   - Create appropriate relationships between:
     - Orders and Customers
     - Orders and Products
     - Orders and Calendar
   - Set proper cardinality for each relationship
3. **Test Your Model**:
   - Switch to Report view
   - Create a visual showing Orders Amount by Product Category
   - Create another visual showing Orders by Customer Country

- Verify that clicking items in one visual filters the other

4. **Document Your Model**:
   - Take a screenshot of your model diagram
   - List each relationship you created and its purpose
   - Note any challenges you encountered

Bonus Challenge: Add a calculated column in the Orders table that calculates profit by subtracting product cost from order amount. Then create a visual showing profit by product category.

Star Schema & Lookup Tables

Understanding Data Modeling Concepts

A well-designed data model is the foundation of an effective Power BI solution. The Star Schema is one of the most popular and efficient data modeling approaches for analytics and reporting.

What is a Star Schema?

A Star Schema is a data modeling technique that organizes data into:

- One central **Fact Table** containing business measurements or events
- Multiple **Dimension Tables** containing descriptive attributes
- Relationships that connect dimensions to the fact table

The name "Star Schema" comes from the shape it creates in a diagram, with the fact table in the center and dimension tables arranged around it like points of a star.

Benefits of Star Schema

- **Simplified Queries**: Clear paths between tables
- **Improved Performance**: Optimized for analytical operations
- **Intuitive Structure**: Easy for users to understand
- **Enhanced Reporting Flexibility**: Analyze by any dimension
- **Consistent Results**: Single version of truth

Fact Tables Explained

Fact tables record business events or measurements that you want to analyze.

Characteristics of Fact Tables

- **Contain measures** (numeric values to analyze)
- Typically have **many rows** (thousands to millions)
- Include **foreign keys** to dimension tables

- Are often **date/time-stamped**
- Are usually **narrow** (few columns but many rows)

Examples of Fact Tables

- Sales transactions
- Website visits
- Manufacturing output
- Survey responses
- Financial transactions

Common Measures in Fact Tables

- Quantities (units sold, number of visits)
- Monetary values (revenue, cost, profit)
- Duration (call time, processing time)
- Rates and ratios (conversion rate, error rate)

Dimension Tables Explained

Dimension tables provide context for the measurements in fact tables.

Characteristics of Dimension Tables

- Contain **descriptive attributes**
- Typically have **fewer rows** than fact tables
- Include a **primary key** (unique identifier)
- Are usually **wider** (more columns but fewer rows)
- Provide **hierarchical relationships** within dimensions

Examples of Dimension Tables

- Products (with attributes like category, brand, size)
- Customers (with attributes like name, segment, location)
- Dates (with attributes like year, quarter, month, day)
- Locations (with attributes like city, region, country)
- Employees (with attributes like department, job title, hire date)

Creating Lookup Tables

Lookup tables are a type of dimension table that normalize your data model by removing redundant information.

When to Create Lookup Tables

- When a column contains repeating values
- When descriptive attributes belong to a distinct entity
- When you need to standardize categorization
- When you want to improve model efficiency

Process for Creating Lookup Tables

1. **Identify Candidates**:
   - Columns with repeating values (like ProductCategory)
   - Columns with multiple related attributes (like ProductCategory, ProductSubcategory)
2. **Extract to a New Table**:
   - Use Power Query's "Remove Duplicates" to create distinct values
   - Add a key column if one doesn't exist
   - Add descriptive columns as needed
3. **Create Relationships**:
   - Connect the lookup table to your fact or main table
   - Ensure proper cardinality (usually one-to-many)

Date Dimensions

A date dimension