DATASOCIETY:

Power BI Intermediate

Day 1

Welcome!

- Hours and breaks
  - 4 days
  - 11 am - 2 pm
  - 1-2 short breaks each class

- Materials
  - PDF slide decks, Datasets, lab and exercise files

Best practices for virtual classes

- Find a quiet place, free of as many distractions as possible. Headphones are recommended.
- Remove or silence alerts from cell phones, e-mail pop-ups, etc.
- Participate in activities and ask questions. This will be interactive!

Who we are

Data Society's mission is to integrate Big Data and machine learning best practices across entire teams and empower professionals to identify new insights.

We provide:

- High-quality data science training programs
- Customized executive workshops
- Custom software solutions and consulting services

Since 2014, we've worked with thousands of professionals to make their data work for them.

Pre-Work for PowerBI

If needed, review the following:

- Overview of Power BI Desktop - 4 minutes
- Getting started with Power BI Desktop - 9 minutes
- Connect to data sources in Power BI Desktop - 8 minutes

Setting Expectations

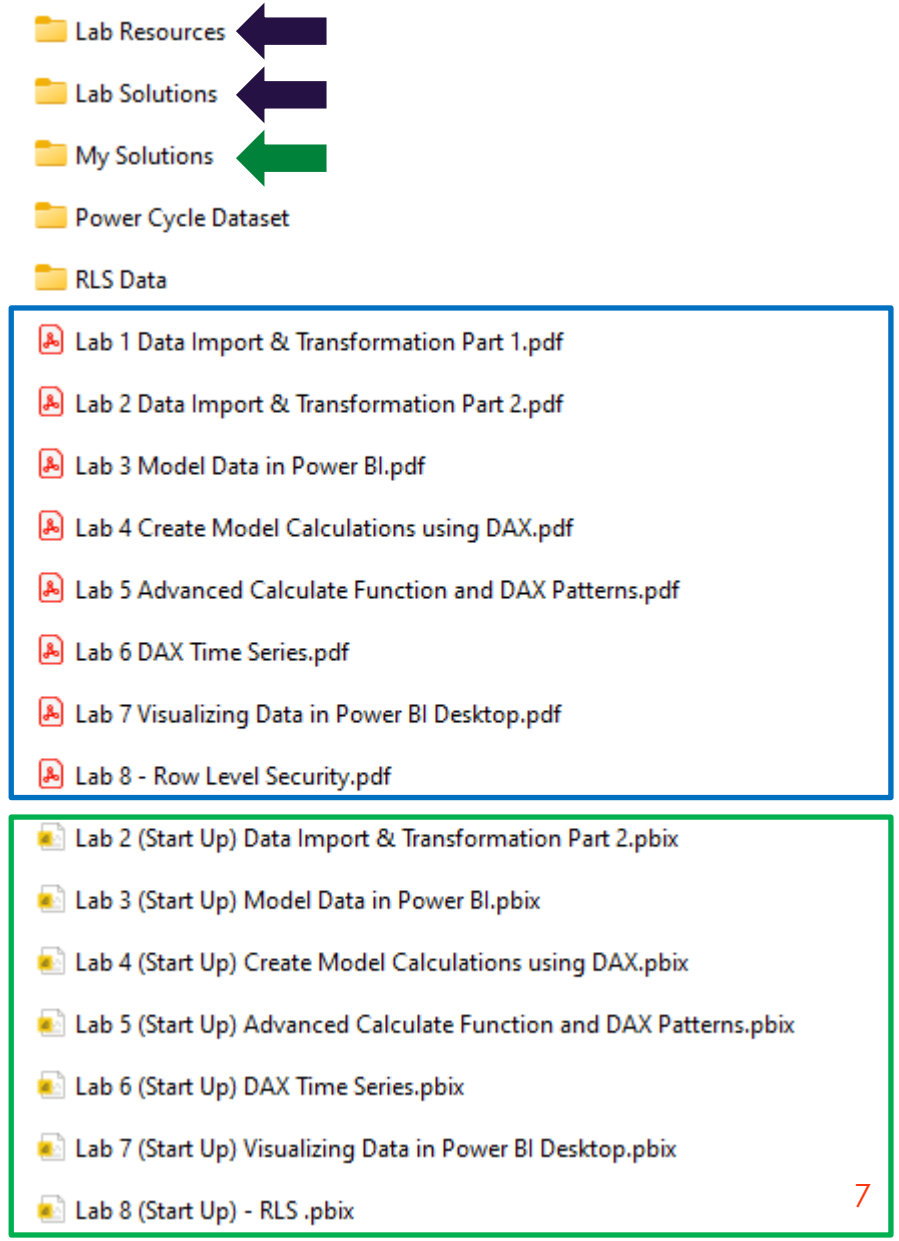
- This course is for users who have a pre-existing working knowledge of Power BI with analytical experience using Power BI Desktop and service
- Each lesson will be accompanied by student exercise and knowledge checks
- This is **not** a DAX-centric course but is designed for Business Intelligence report development
- Students are advised to explore the additional resources to further skill development beyond this course

Introduction to the Data Sets

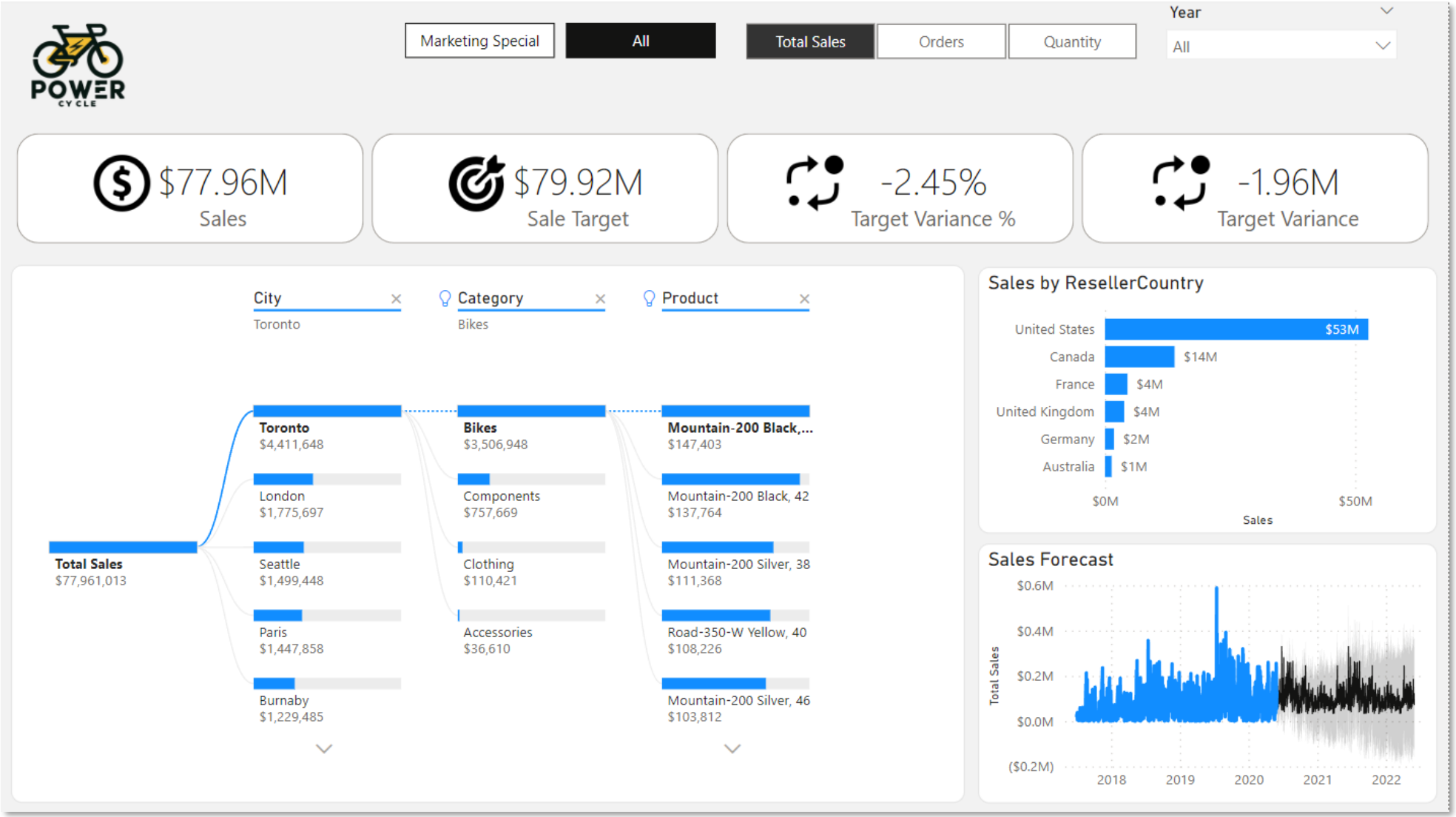
For the lab exercises, we will be using the Power Cycle data set. This is a Global fictitious Cycle Retailer and your Job will be to use the data set to complete the lab based on the instructions provided in the lab resource materials.

Fact Tables (Sales Table)
Australia
Canada
France
Germany
UK
US
Power Cycle Target

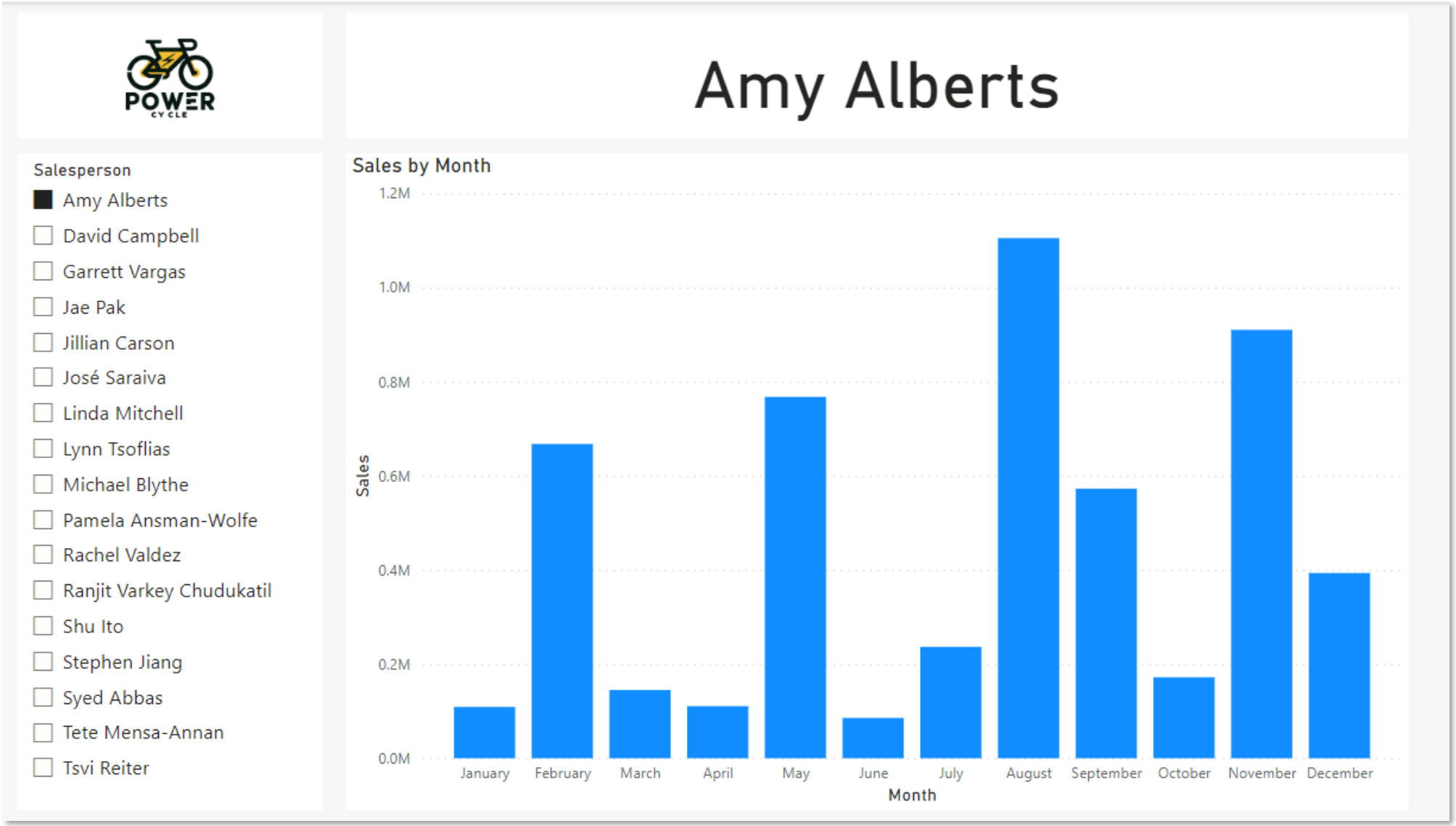
Dimension Tables
Product Category
Product Sub-Category
Product Sales Region
Salesperson



Final Report



Employee RLS Page

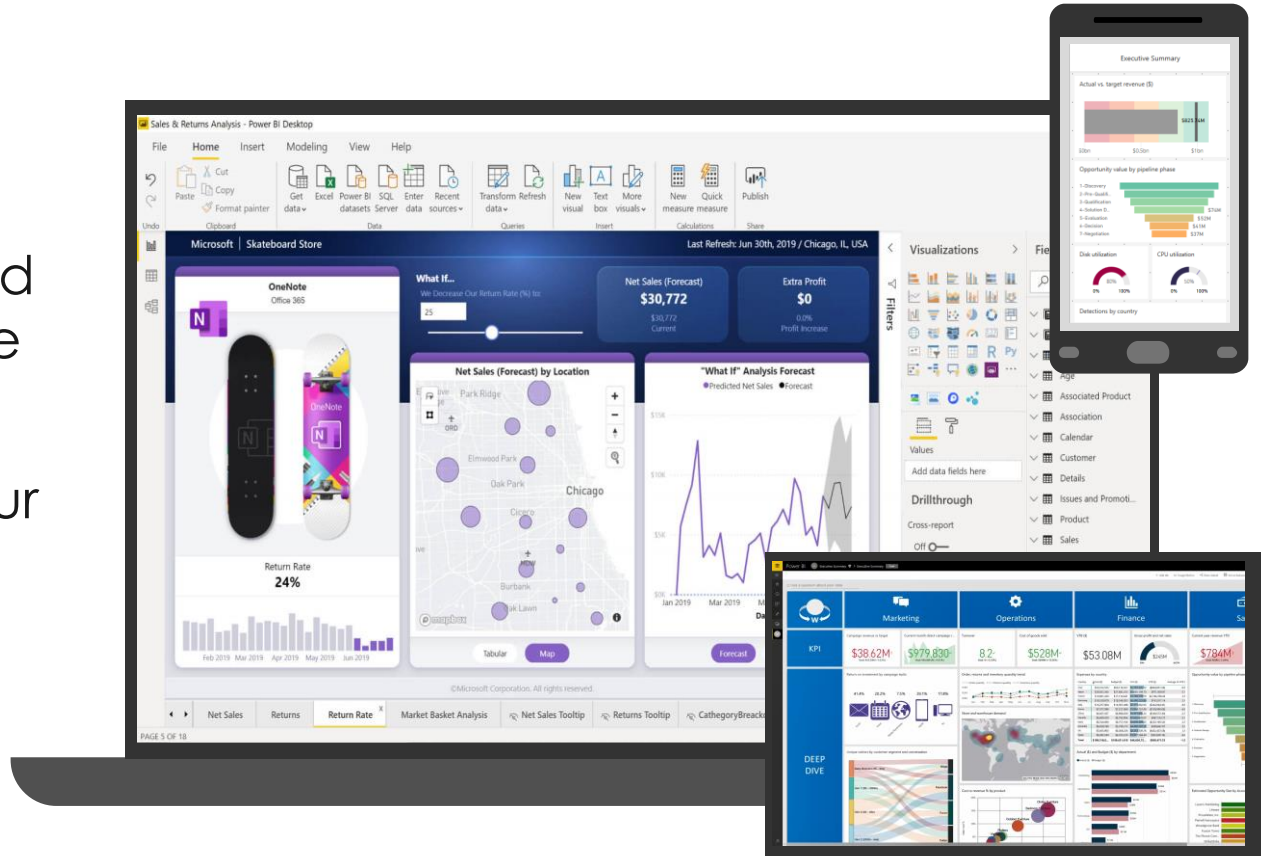


Power BI Overview

Connect to all your data and get a consolidated view across your business through a single pane of glass

Create ad-hoc analysis, live dashboards and interactive reports that are easy to consume on the web and across mobile devices

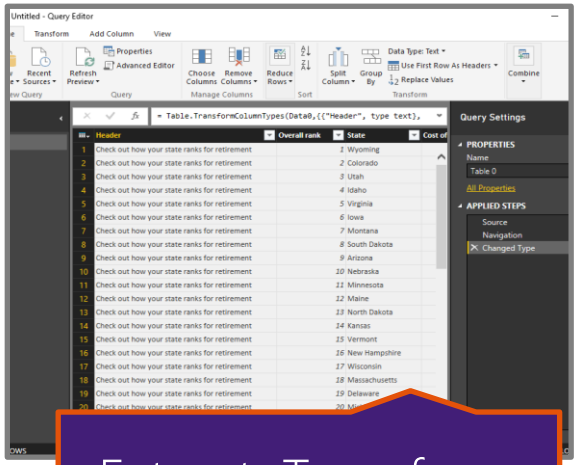
Build smart apps by infusing insights from your data and drive action with the power of Microsoft Power Platform



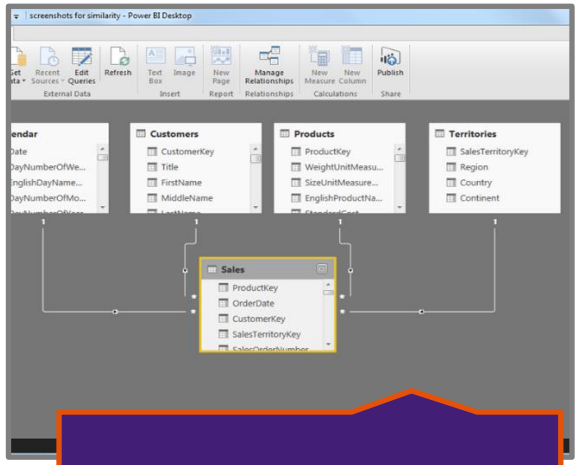
4 Layers of Power BI

Model Developer

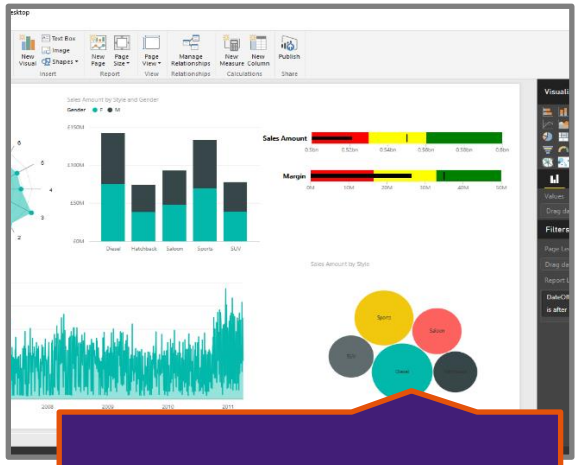
Report Developer



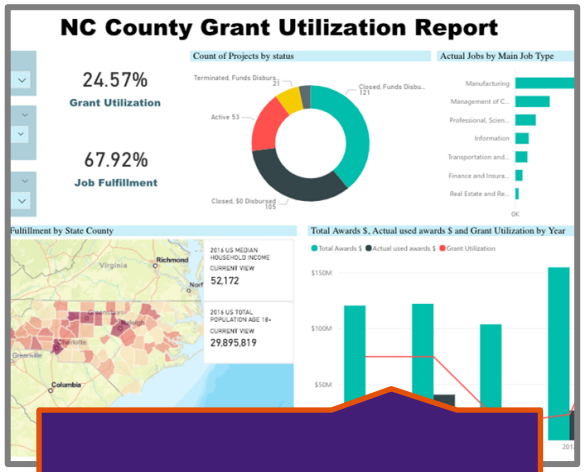
Extract, Transform and Load (ETL)



Data Modeling



Report Design



Web Portal

Introduction to the Lab Exercises

1. The data set is from a fictitious cycling retailer with an international footprint. Your goal as the analyst is to use Power BI and DAX calculations to answer key questions about the business including sales trends, top preforming resellers, actual sales vs. budget and segmentation.

2. There will be a total of 6 Lab to be completed. Each Lab will build upon the previous one

3. All you've been given is a collection of raw csv files containing POS sales records, along with details about the company's products, customers, locations and employees.

**This data is adapted from Microsoft Adventure works data pack and is for informational purposes only . These samples are provided"as is" without warranty of any kind. The example companies, organizations, products, domain names, email addresses, people, places, and events depicted herein are fictitious, and no association with any real company, organization, product, person, place, or event is intended or should be inferred.*

Lesson 1
Getting data from different sources

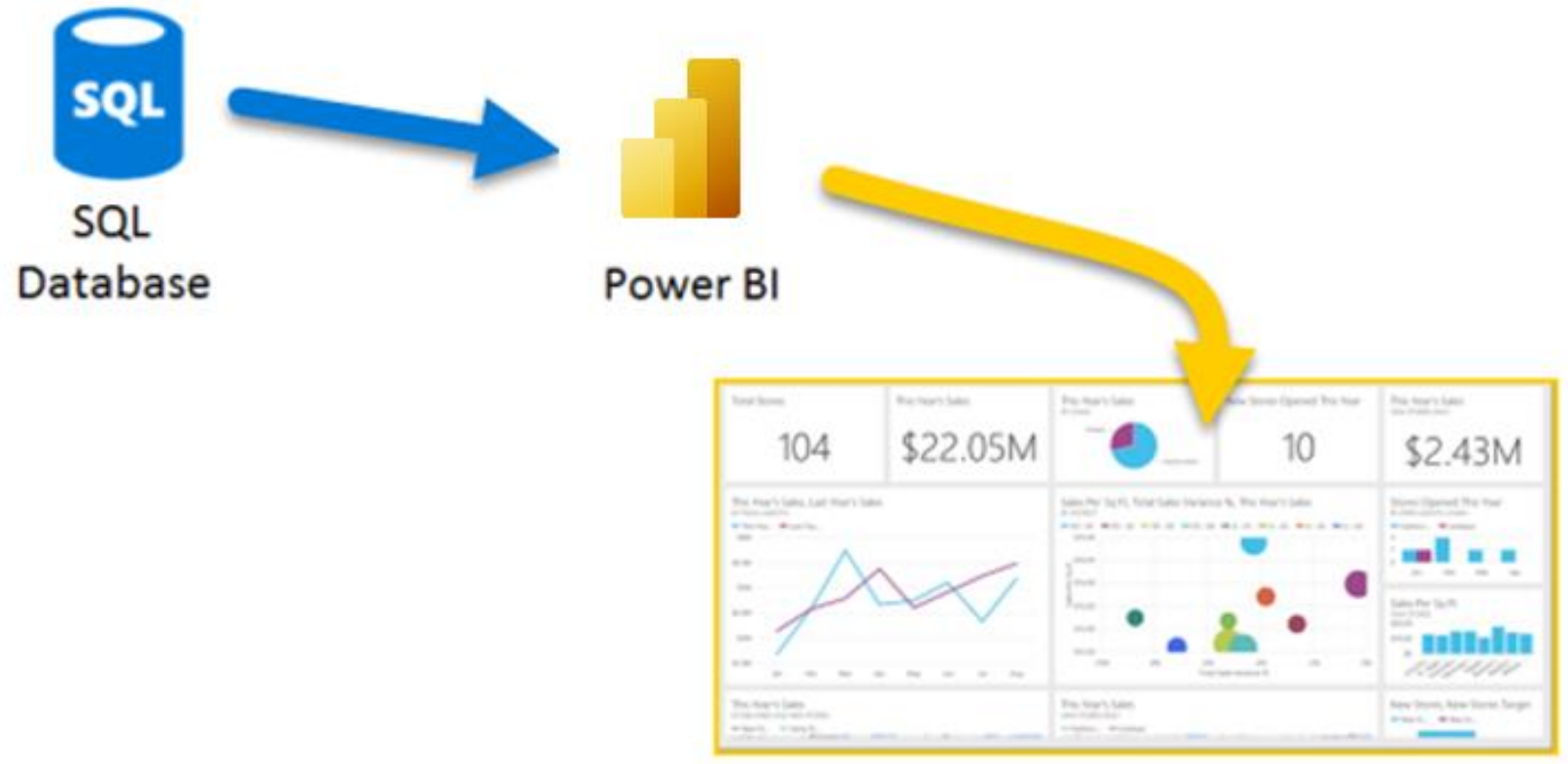
After completing this module, students will be able to:

- Identify and retrieve data from different data sources
- Understand the connection methods and their performance implications
- Connecting to Sharepoint and Power BI services
- Understanding storage modes

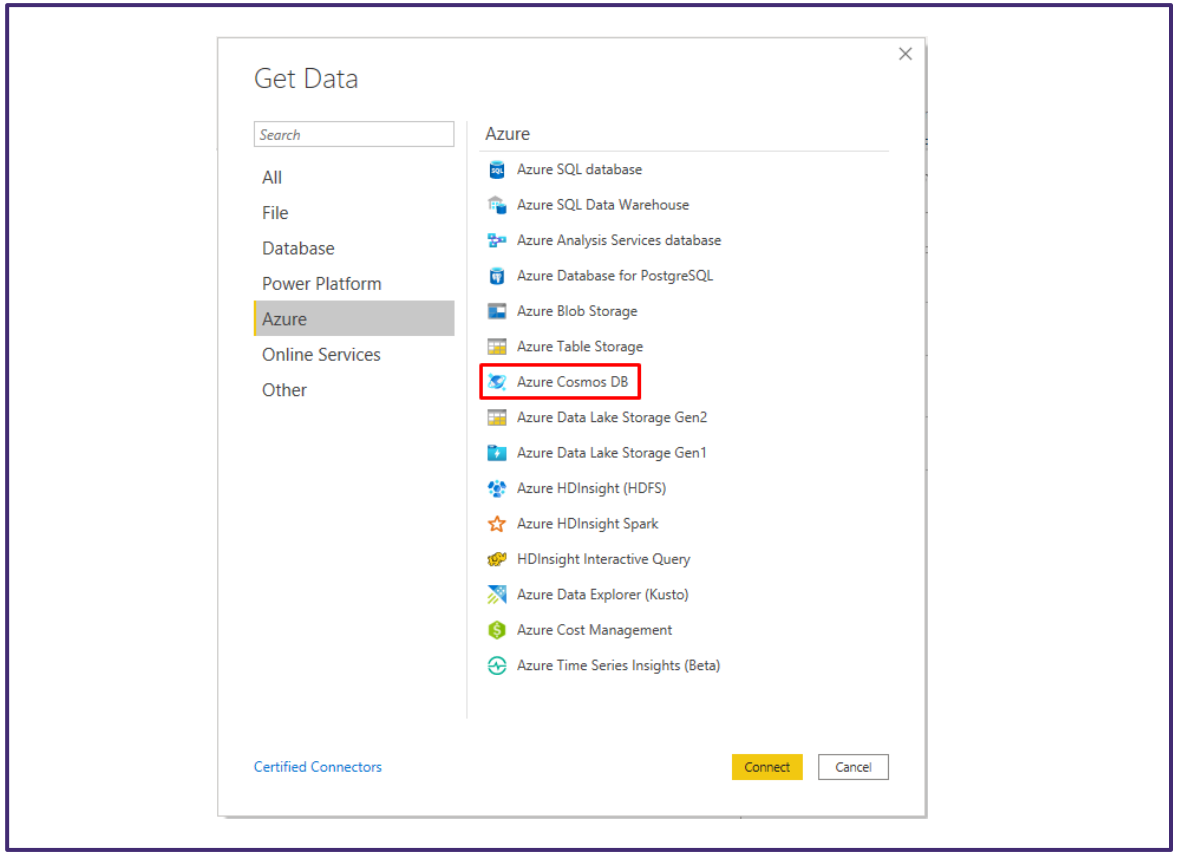
Source Types

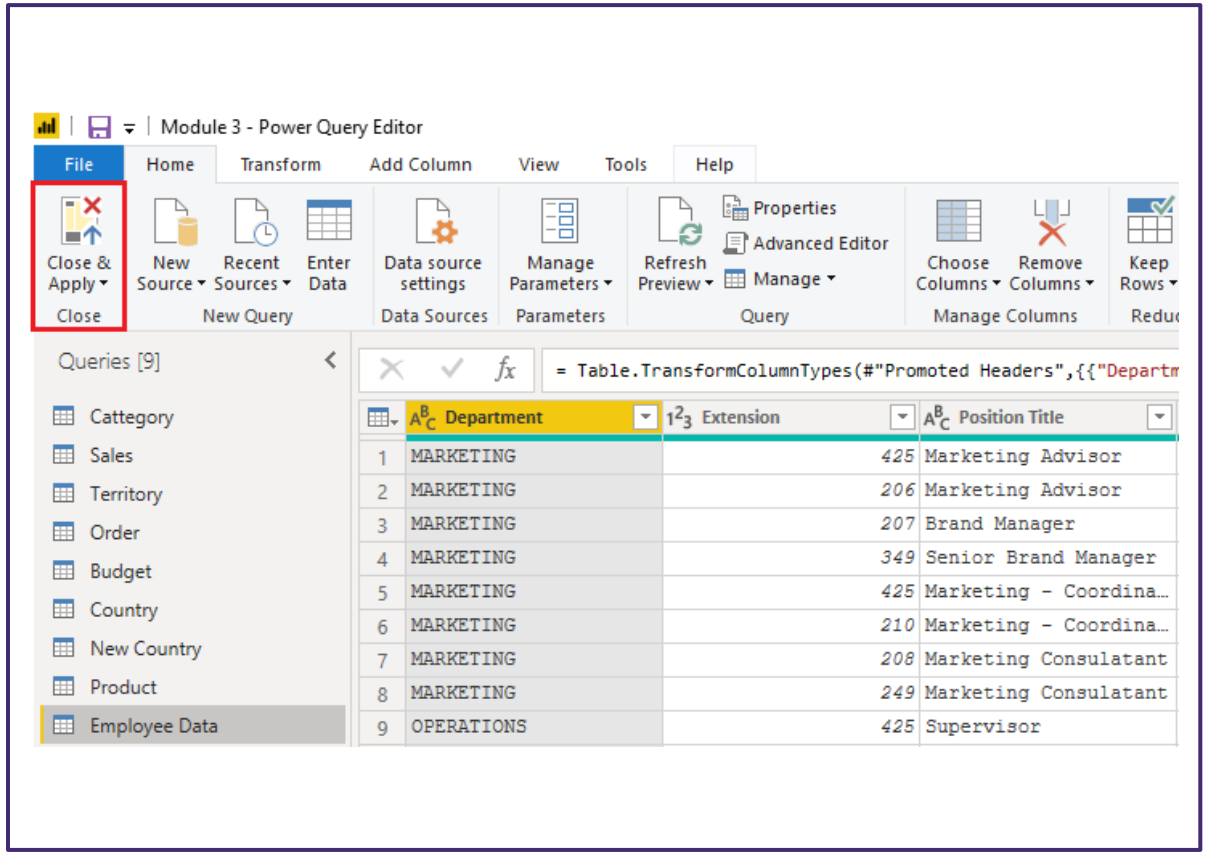
- Flat File
- Relational Data Sources
- Non-SQL Data (unstructured)
- Online Services
- API's

Get data from relational data sources

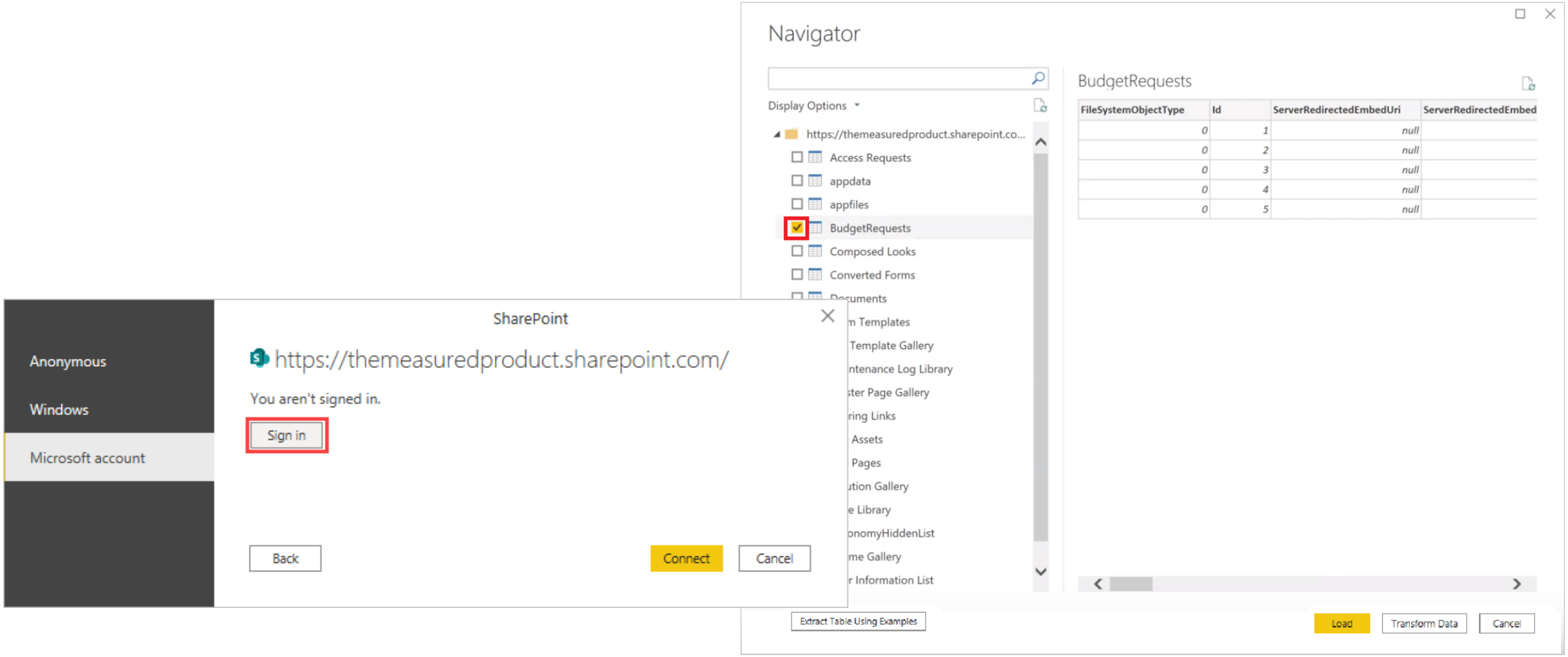


Unstructured Data Source



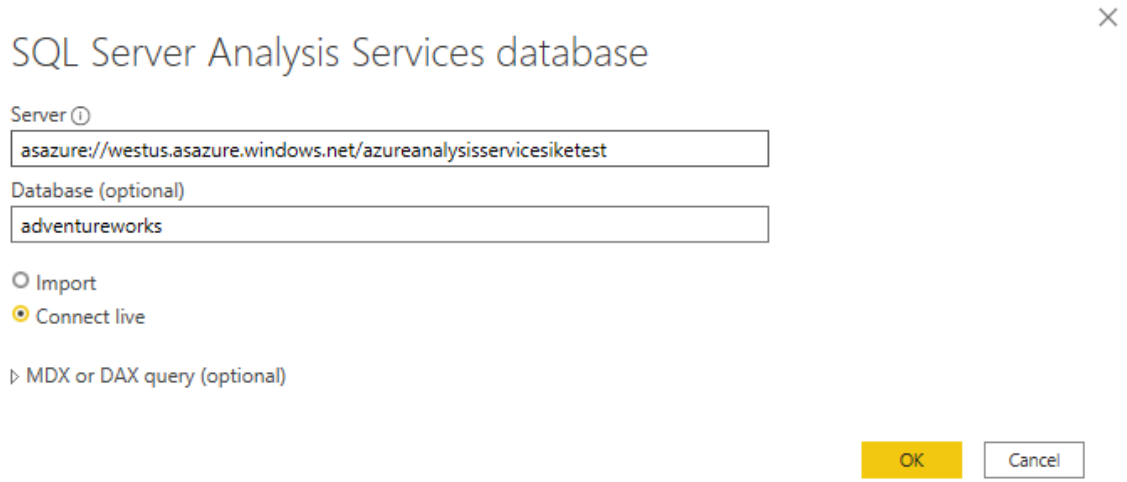


Getting Data From SharePoint

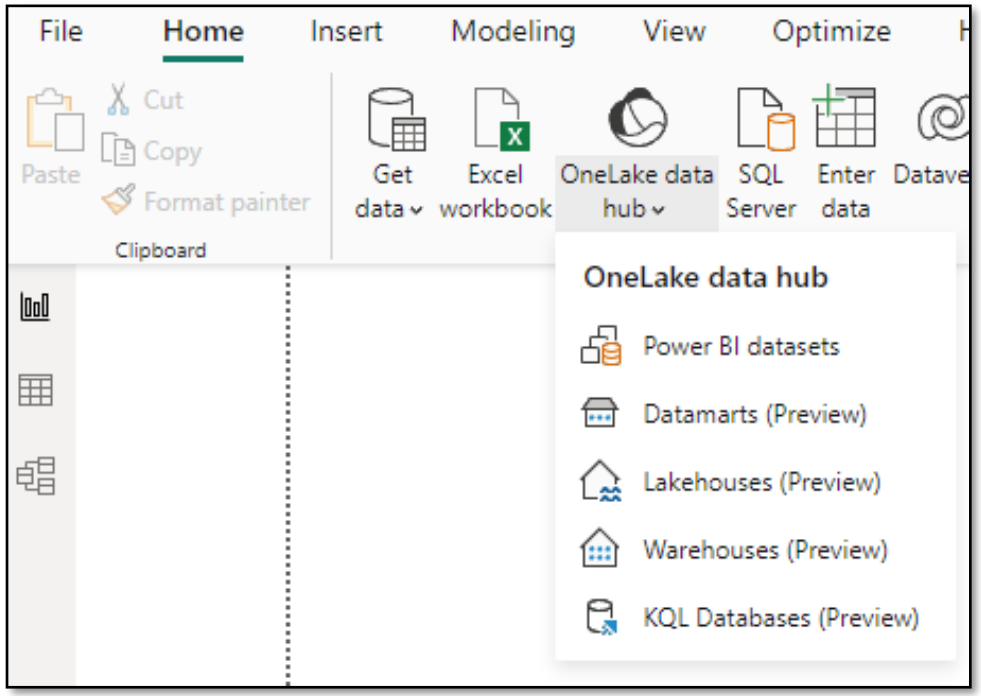


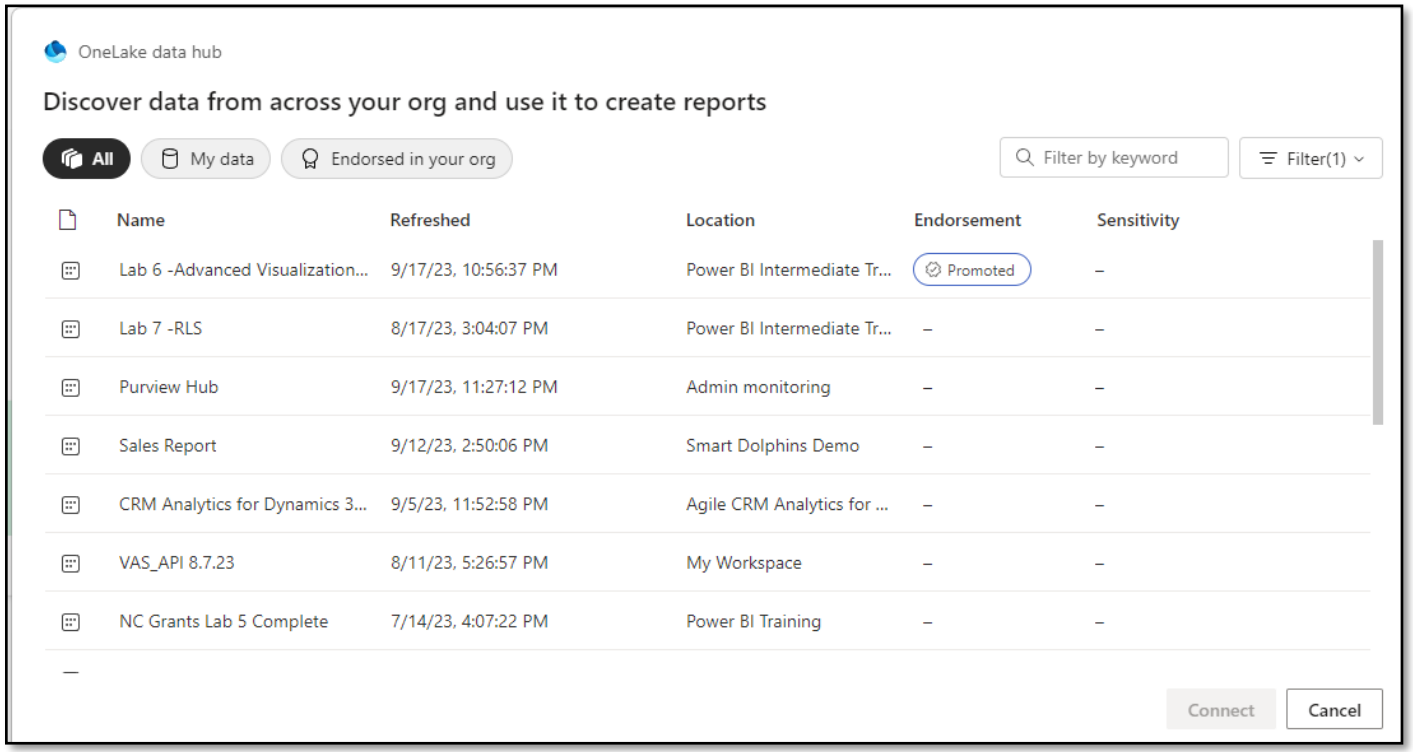
Get data from Analysis Services

An analytical data engine that lets you digest data from multiple data sources and create calculations on the fly.

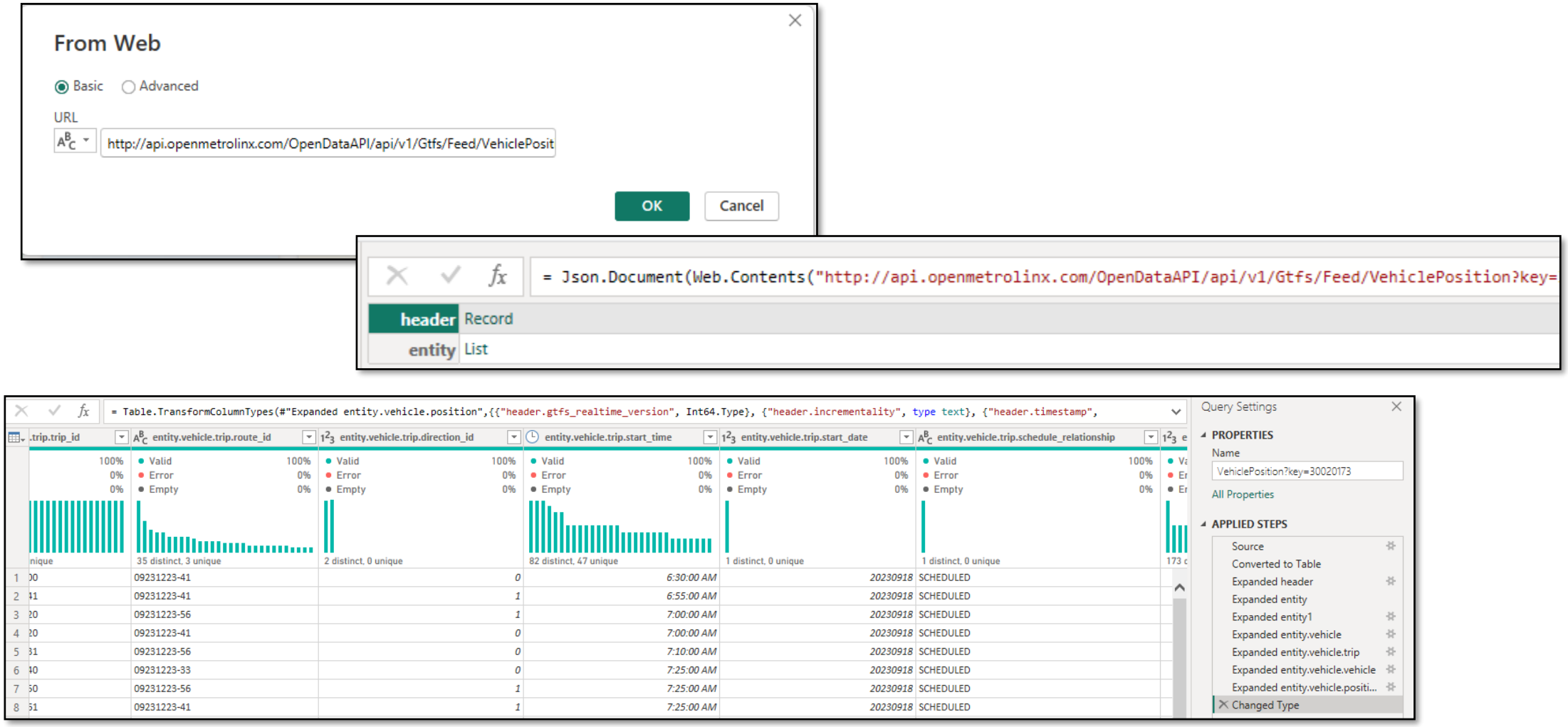


Power BI Published Dataset





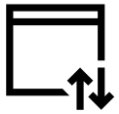
Web Services & API's



Power BI Desktop Data Storage Modes

Import

- Dataset is less than 1GB (after compression) & fast performance
- Source data does not change frequently
- No restrictions on Power Query, data modeling, and DAX functions



DirectQuery

- Dataset is too large to be stored in-memory
- Source data changes frequently and reports must reflect changes
- Company policy states that data can only be accessed from the original source

Dule (Composite Model)

- Boost performance by setting appropriate storage for each table
- Combine a DirectQuery model with additional imported data
- Create a single model from two or more DirectQuery models

Live Connection

- Create one dataset that serves as a central source of truth
- Analyst teams can create different reports from the same source
- Multi-developer teams where one user builds the model and another works on visualization

**Not all sources support all modes*

Implications of using DirectQuery

Benefits

- Frequently changing data
- Need near real-time
- Large data volumes
- Multi-dimensional data

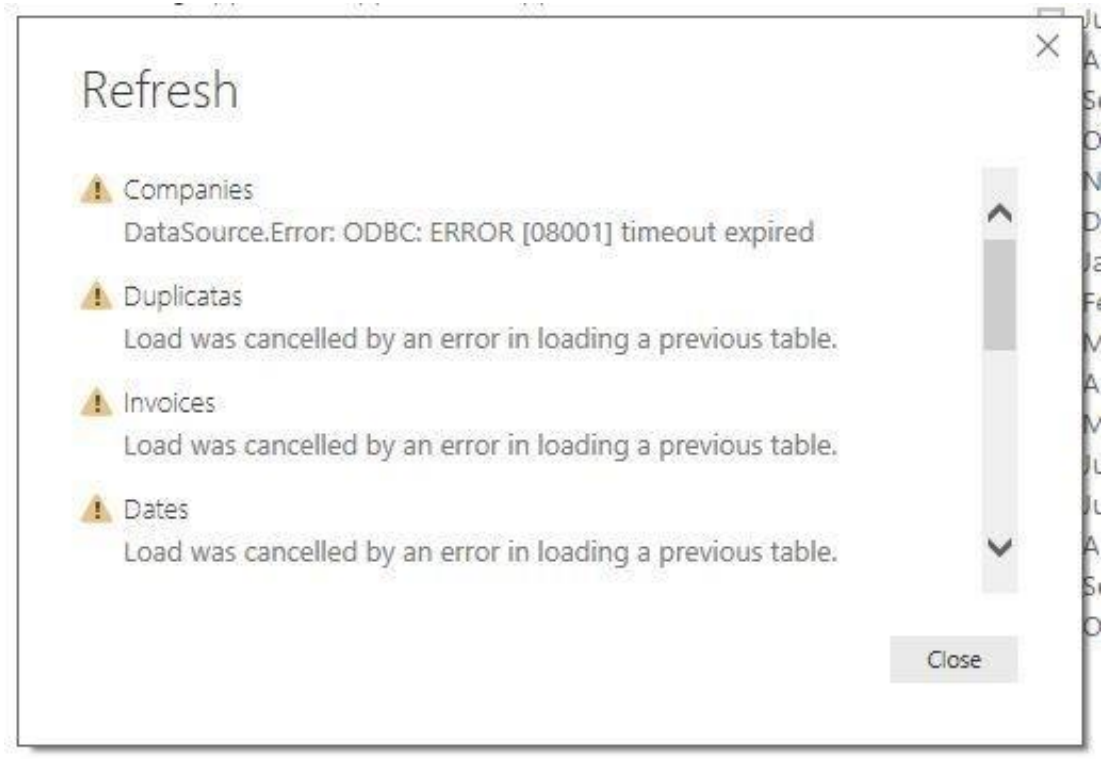
Limitations

- Performance: Dependent on the underlying data source.
- Security: Understand how data moves between source and destination.
- Modeling: Some modeling capabilities are limited or aren't supported.
- Transformation: Some data transformation techniques are limited.

Dealing with Import Errors

When loading your data, you may encounter the following errors:

- *Query Timeout*
- *Couldn't find data formatted as a table*
- *Could not find file*
- *Data type errors*



Chat Question

What are some common data import errors do you encounter when creating reports?

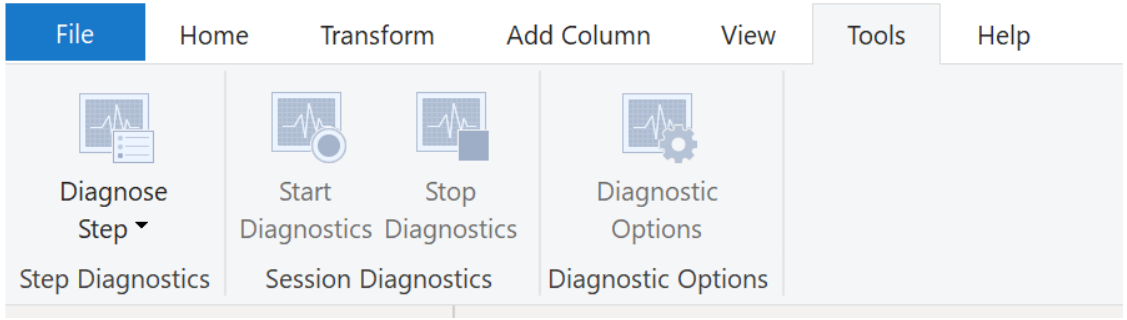
Optimize Query Performance

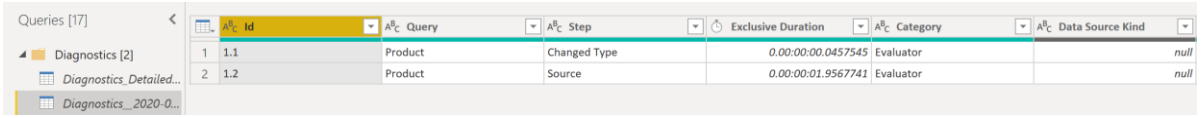
Performance in Power Query depends heavily on the performance at the data source.

Follow performance tuning guidelines of the source product.

Some performance tuning can be done in Power BI.

Fix Performance Issues





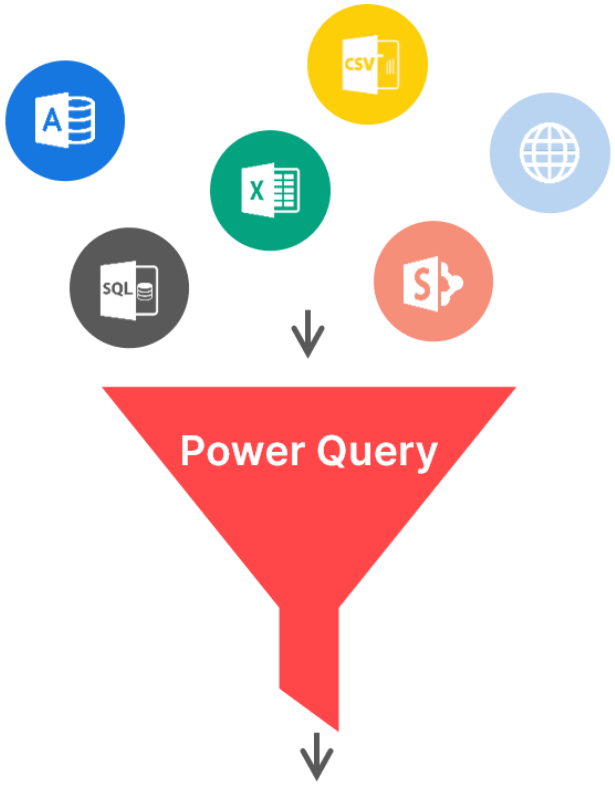
KC 1

Lesson 2
Clean Transform, Load the Data

After completing this module, students will be able to:

- Apply data shape transformations
- Enhance the structure of the data
- Profile and examine the data

Power Query Editor



Clean and **structured data** tables that you can load, share and analyze

Before

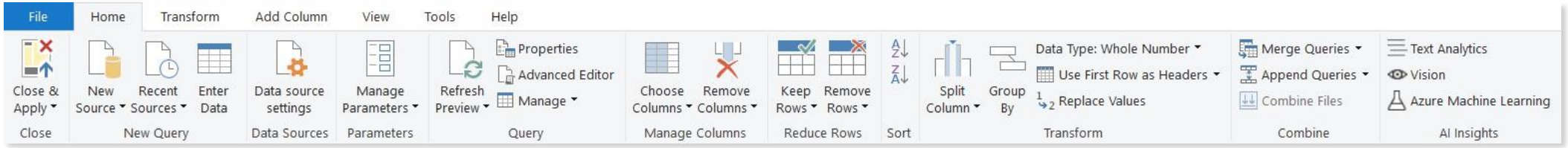
	Date	Customer	Prod 1	Kg 1	Bag 1	Prod 2	Kg 2	Bag 2
1	1/1/2019	A	Rice	10	1	Potato	100	10
2	1/2/2019	A	Potato	20	2	Rice	200	20
3	1/3/2019	B	Tomarto	30	3	Beans	300	30
4	1/4/2019	C	Beans	40	4	Beans	400	40

After

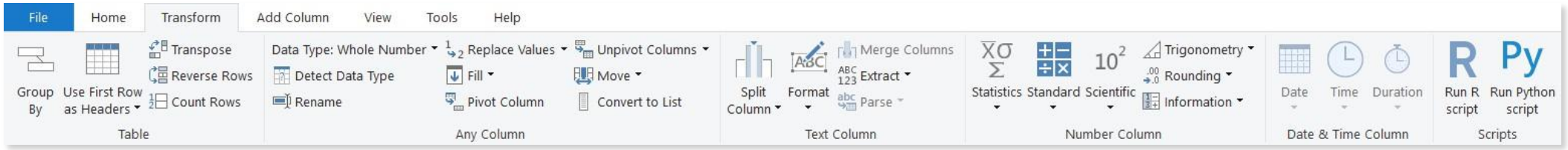
	Date	Customer	Product	1.2 Kg	1.2 Bag
1	2019-01-01	A	Potato	100	10
2	2019-01-01	A	Rice	10	1
3	2019-01-02	A	Potato	20	2
4	2019-01-02	A	Rice	200	20
5	2019-01-03	B	Beans	300	30
6	2019-01-03	B	Tomarto	30	3
7	2019-01-04	C	Beans	440	44

Common transformations

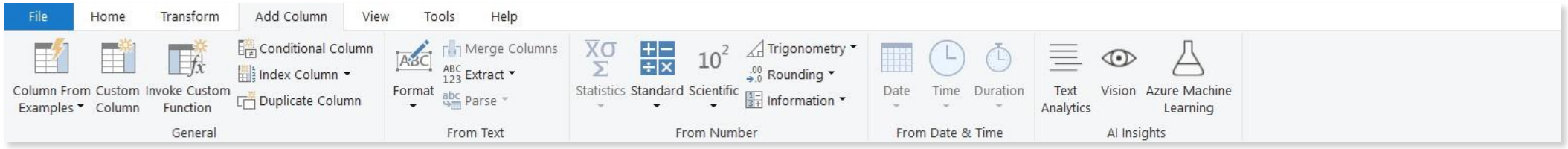
The **HOME** tab includes general settings and common table transformation tools



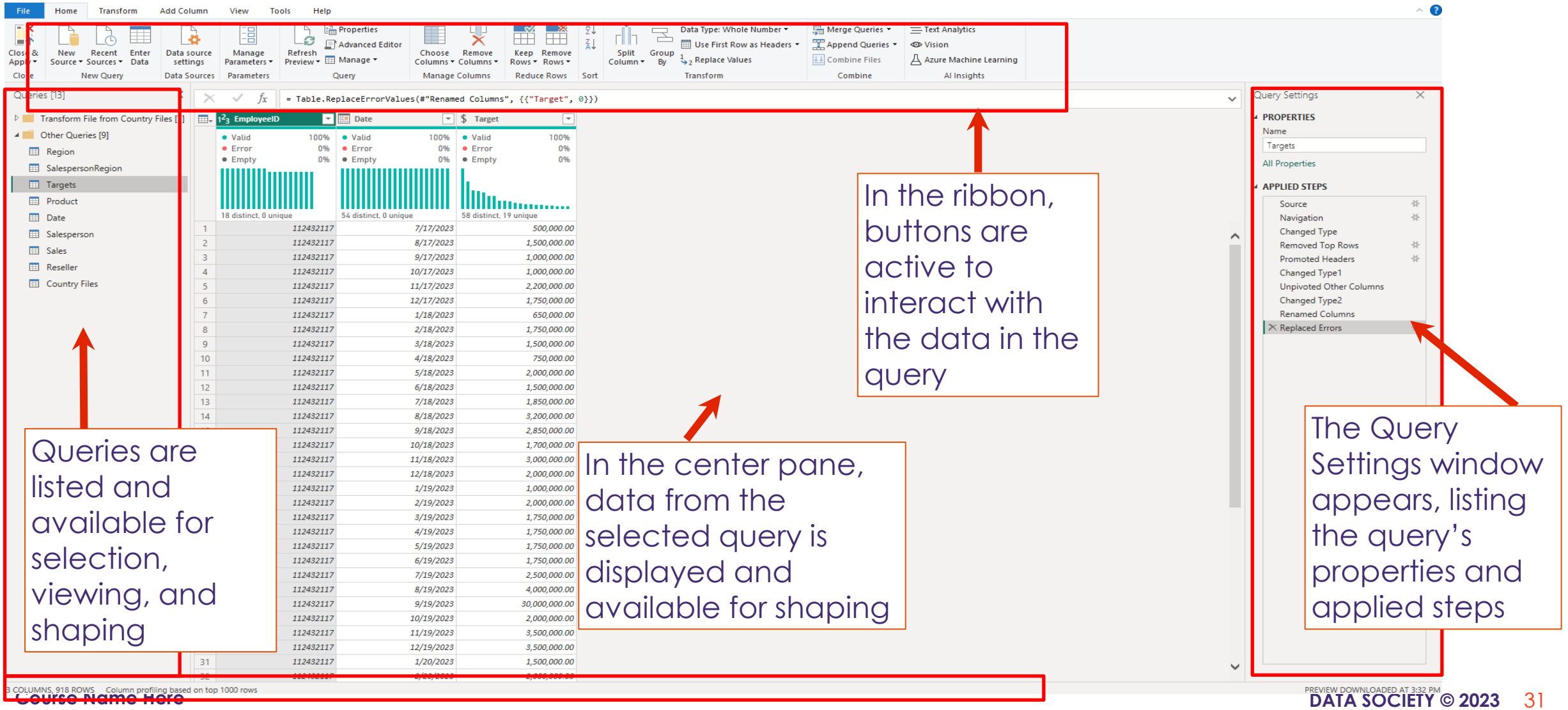
The **TRANSFORM** tab includes tools to modify existing columns (splitting/grouping, transposing, extracting text, etc.)



The **ADD COLUMN** tools create new columns (based on conditional rules, text operations, calculations, dates, etc.)



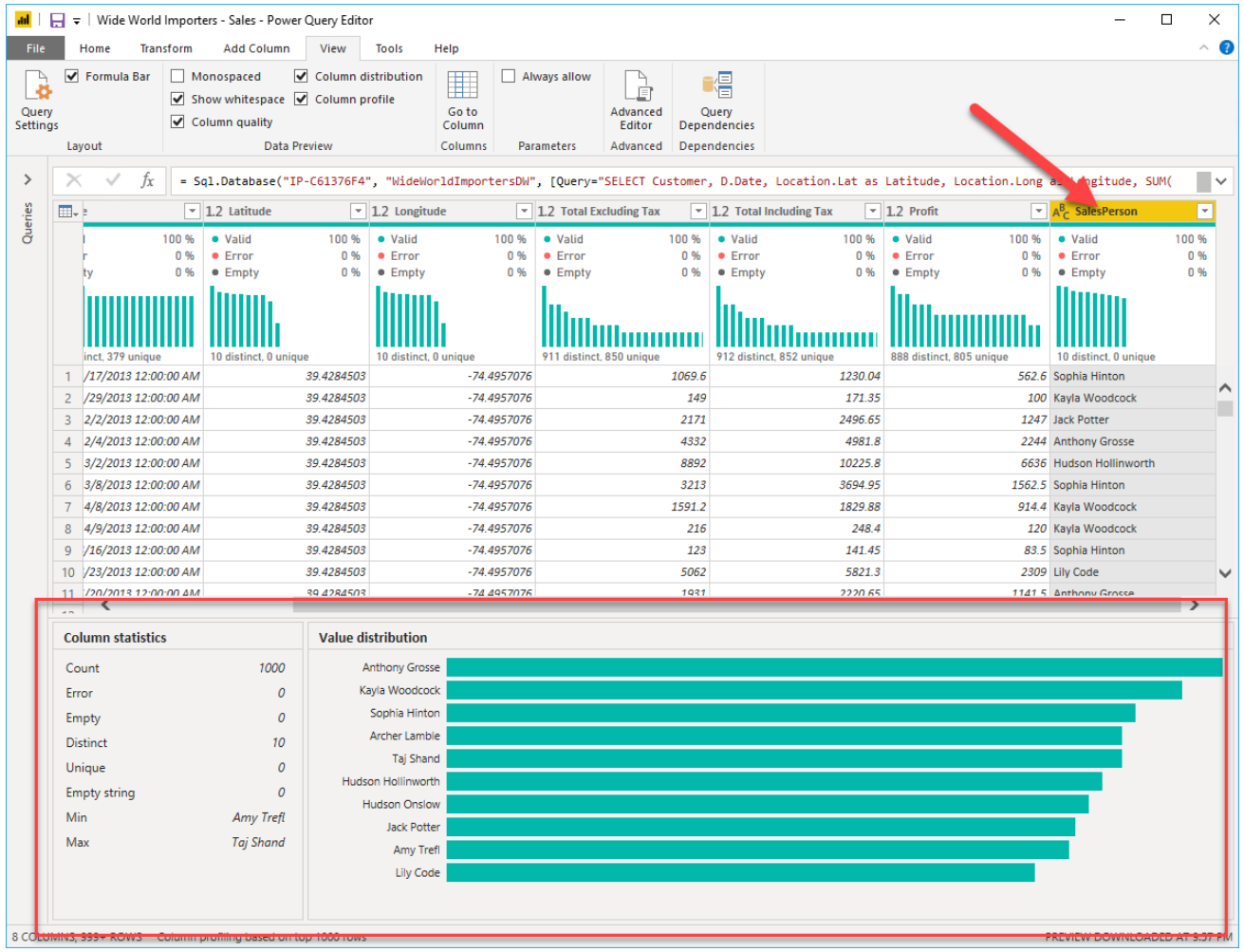
Power Query Interface



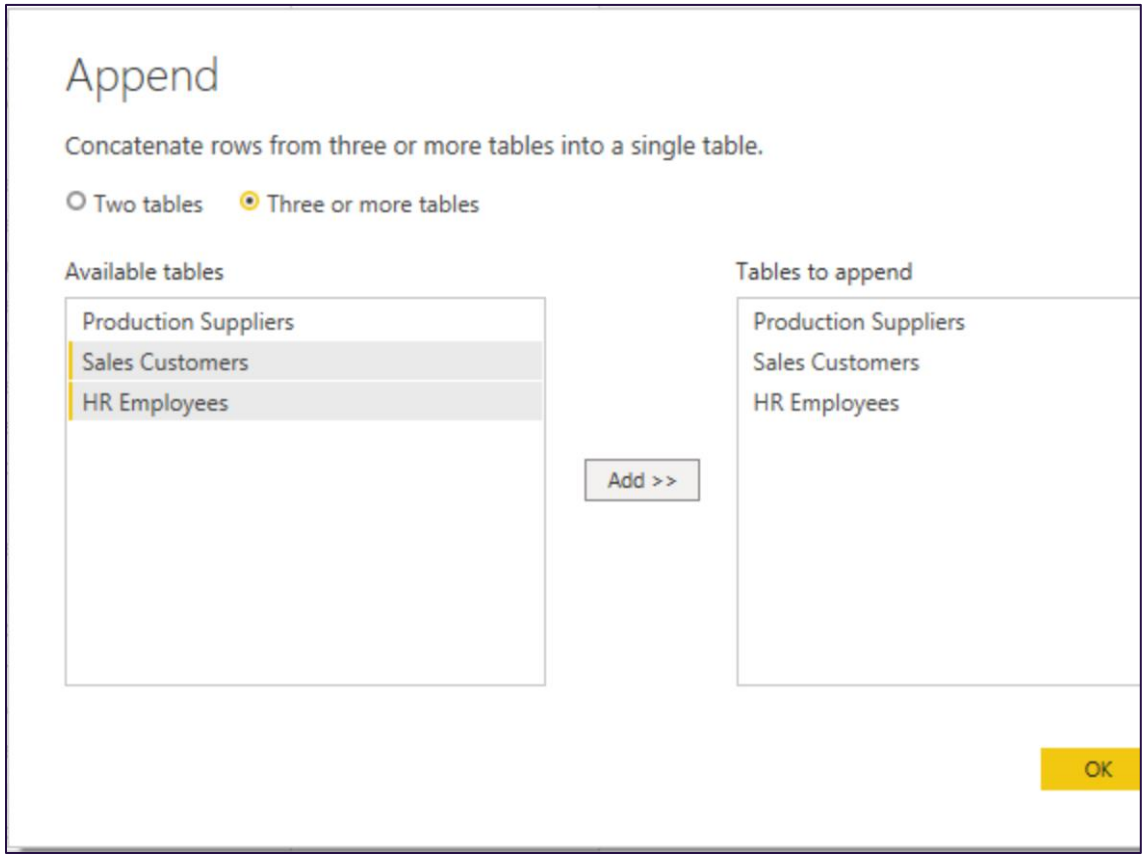
Find data anomalies and data statistics

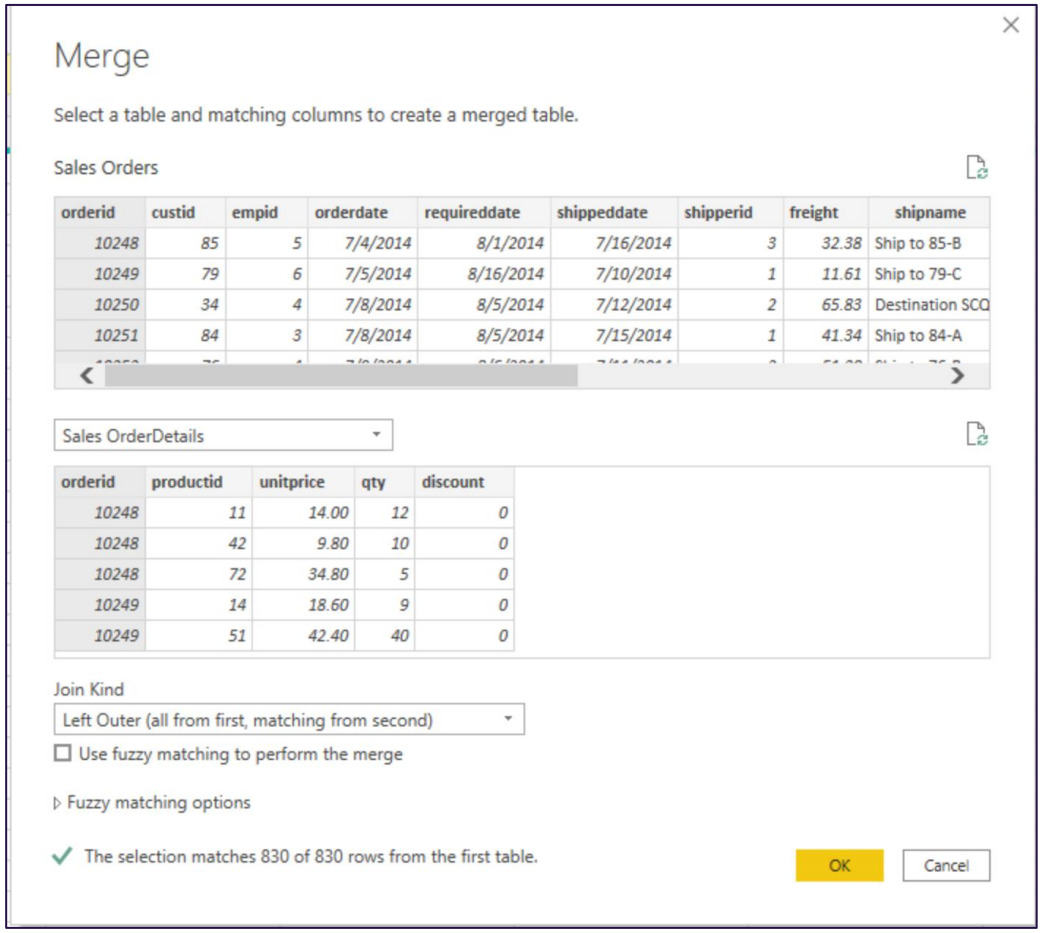
Data profiling entails studying data nuances, detecting anomalies, exploring underlying structures, and querying statistics like row counts and value distributions. This is crucial for simplifying data interaction and streamlining the development of report elements on the frontend.

Note: Data Preview is based on the first 1,000 rows

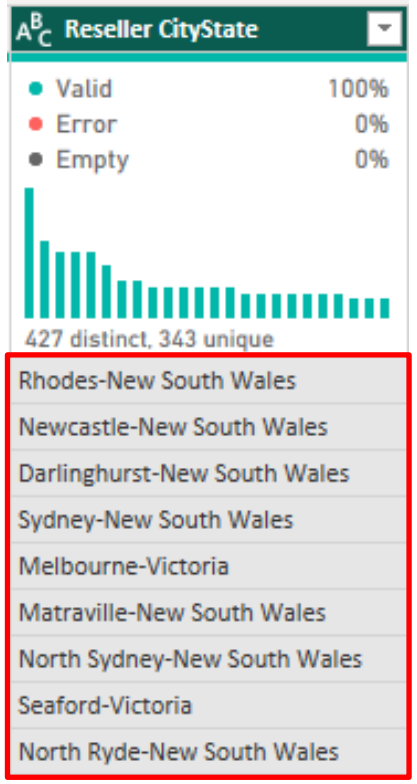


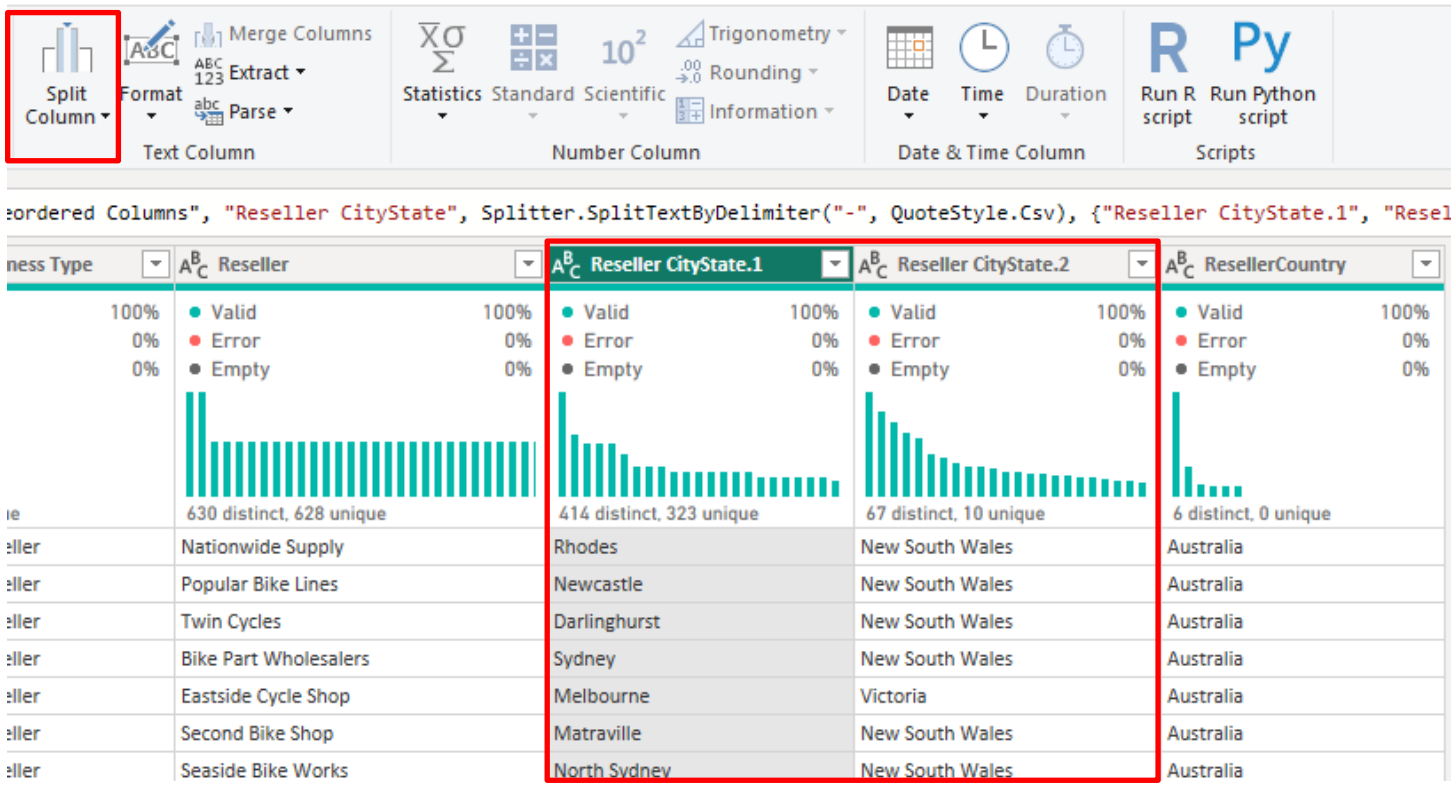
Combine multiple queries into one





Splitting Columns





To split columns in Power Query, first, select the column you want to divide. Then, use the "Split Column" function and specify whether you want to split by a delimiter (like a comma or space) or by a fixed number of characters.

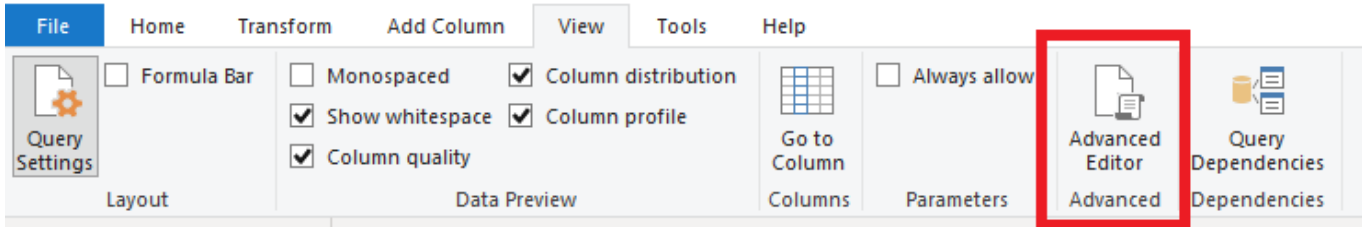
Unpivot columns or pivot

	Category Name	Subcategory Name
1	Bikes	Mountain Bikes
2	Bikes	Road Bikes
3	Bikes	Touring Bikes
4	Clothing	Bib-Shorts
5	Clothing	Caps
6	Clothing	Gloves
7	Clothing	Jerseys
8	Clothing	Shorts
9	Clothing	Socks
10	Clothing	Tights
11	Clothing	Vests
12	Accessories	Bike Racks
13	Accessories	Bike Stands
14	Accessories	Bottles and Cages

	1.2 Bikes	1.2 Components	1.2 Clothing	1.2 Accessories
1	3	14	8	12

Use Advanced Editor to modify M code

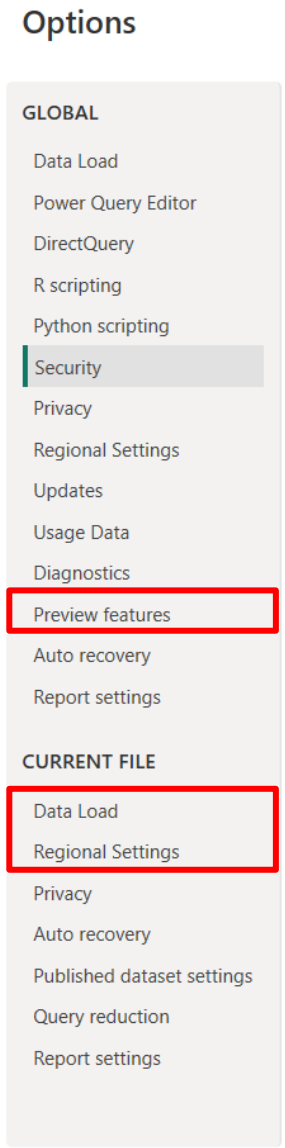
When shaping data in Power Query, each action creates a step in the process that can be reordered, deleted, or modified. While the graphical interface is commonly used, Power Query operates with the underlying M language.

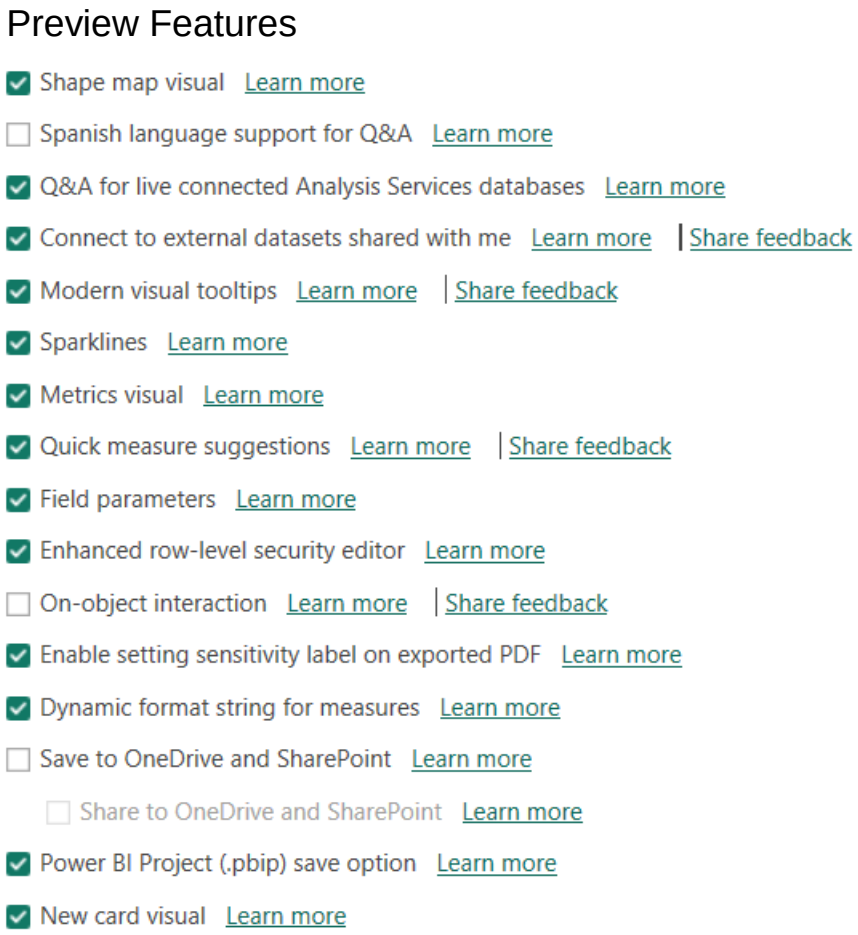


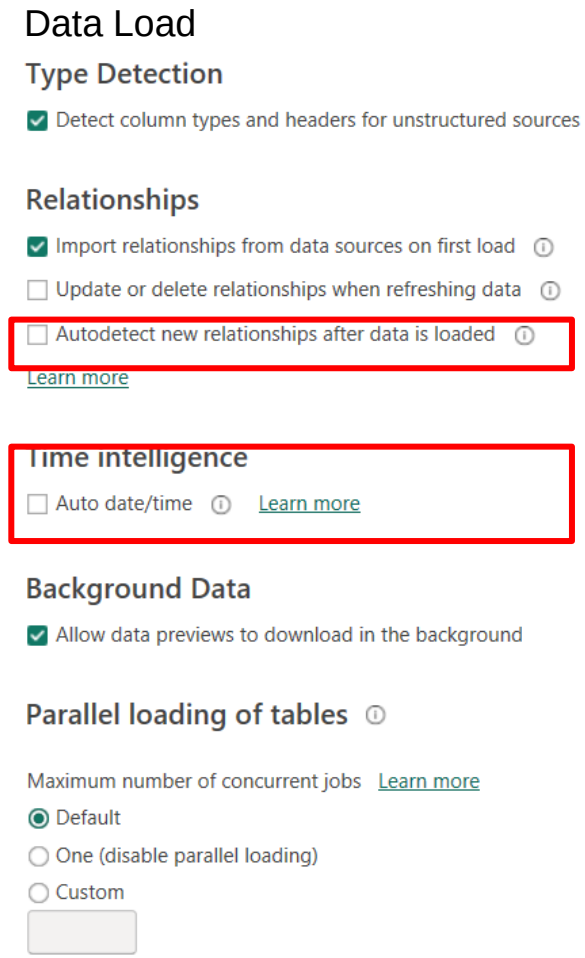
Data reduction best practices

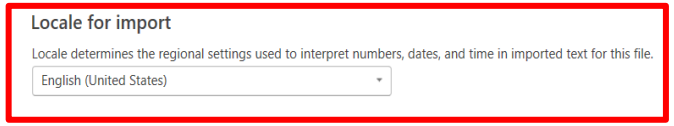
- ✓ Remove unnecessary columns
- ✓ Remove unnecessary rows
- ✓ Group by and summarize
- ✓ Optimize column data types
- ✓ Preference for custom columns
- ✓ Disable Power Query load
- ✓ Disable auto date/time
- ✓ Switch to Mixed mode (Composite model)

Power BI Options & Settings



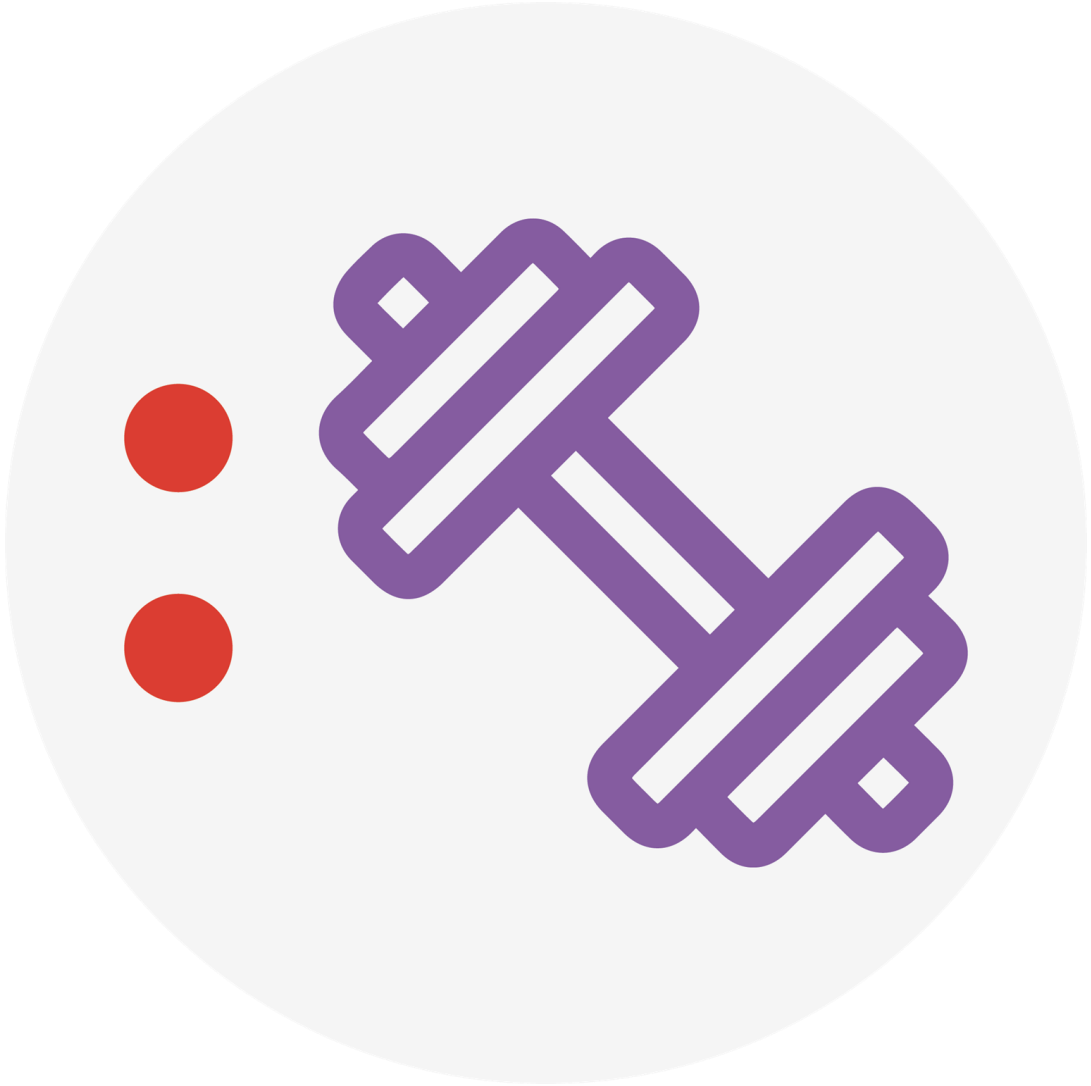






ere

Live Demo



KC2

DATASOCIETY:

Power BI Intermediate
Day 2

Lesson 3
Design a Data Model

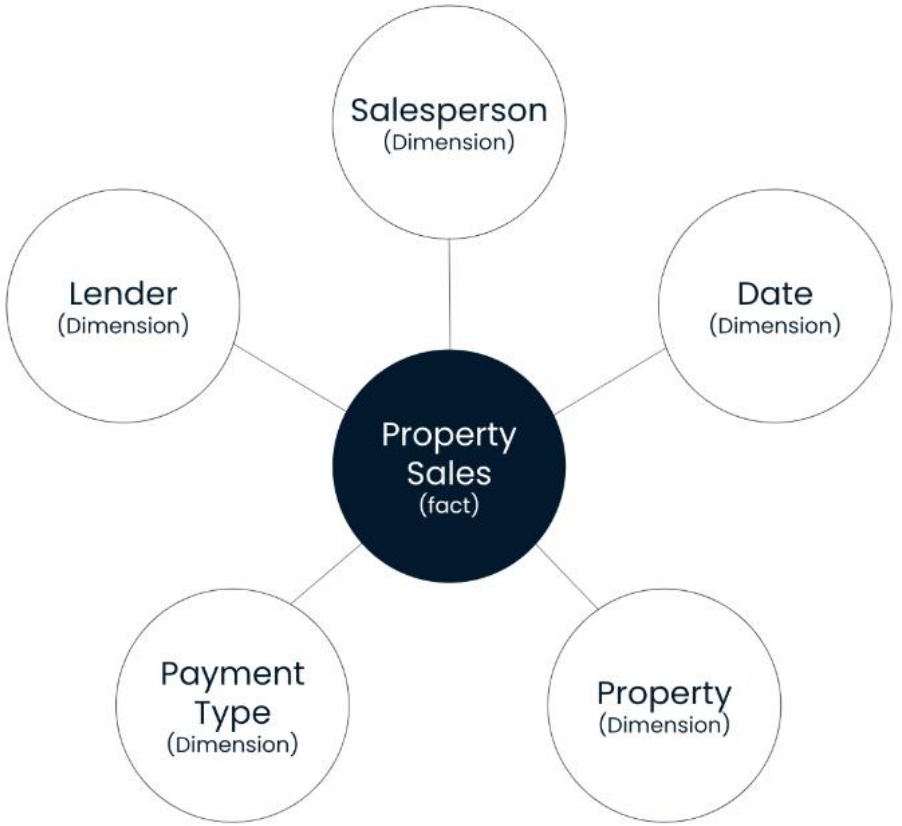
After completing this module, students will be able to:

- Define relationships and their cardinality
- Implement Dimensions and Hierarchies
- Understanding filter propagation

STAR SCHEMA

- **Fact tables** contain observational or event data values: sales orders, product counts, prices, transactional dates and times, and quantities.

- **Dimension tables** contain the details about the data in fact tables: products, locations, employees, and order types. These tables are connected to the fact table through key columns. Dimension tables are used to filter and group the data in fact tables



Snowflake Schema

Snowflake schema allows relationships between dimensions and differs from the star schema in that it has more Hierarchies between tables.

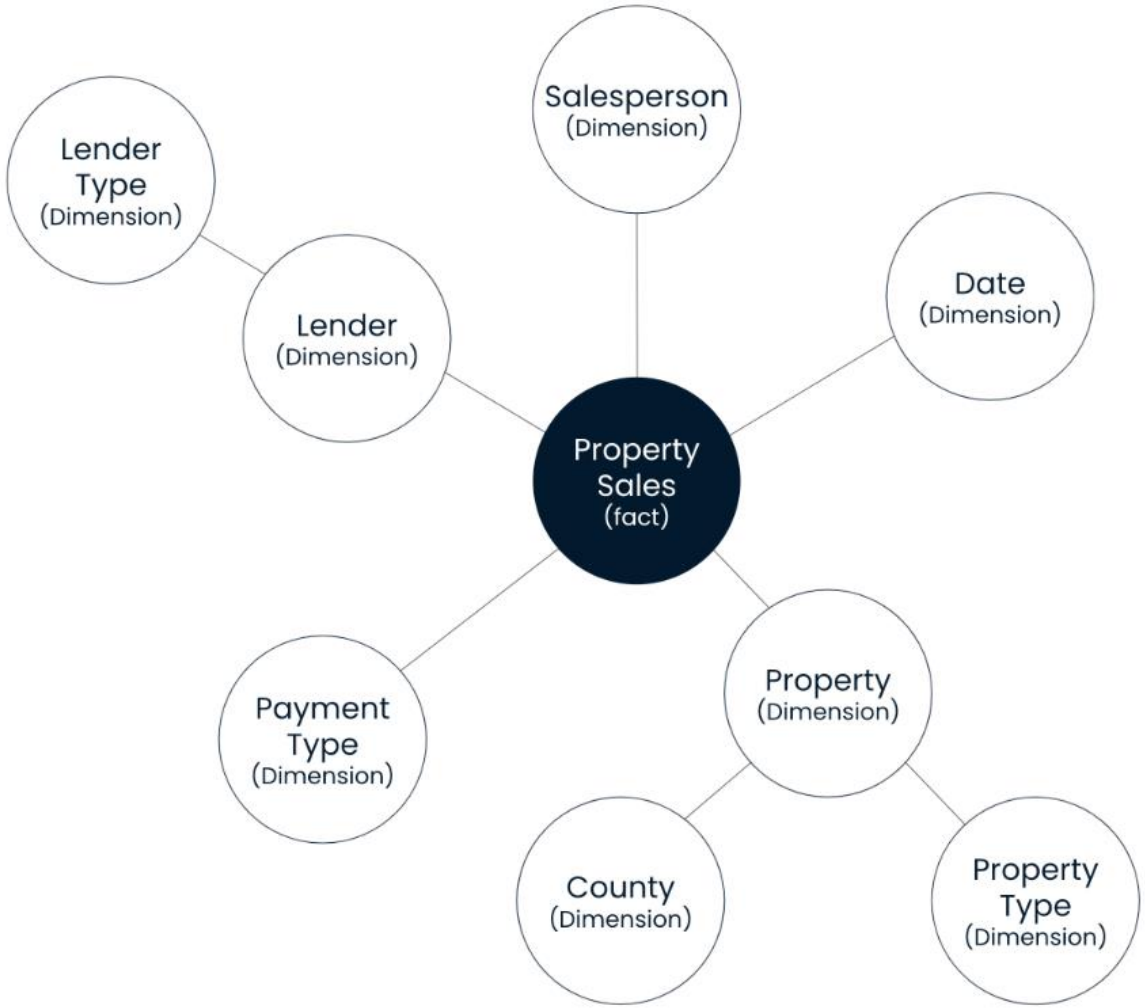


Table Normalization

	SalesOrderNumber	OrderDate	ProductKey	ResellerKey	SalesAmount
1	SO69561	2020-05-31	594	546	226.00
2	SO69560	2020-05-30	513	100	218.45
3	SO69560	2020-05-30	594	100	113.00
4	SO69539	2020-05-28	243	529	858.90
5	SO69539	2020-05-28	378	529	1466.01
6	SO69541	2020-05-28	594	661	113.00
7	SO69542	2020-05-28	243	317	1717.80
8	SO69544	2020-05-28	243	666	3435.60
9	SO69545	2020-05-28	378	436	5864.04
10	SO69532	2020-05-27	594	312	113.00
11	SO69532	2020-05-27	513	312	436.90
12	SO69533	2020-05-27	594	476	226.00

	SalesOrderNumber	OrderDate	ProductKey	Product	Category	Color	Size	ResellerKey	SalesAmount
1	SO69561	2020-05-31	594	Mountain-500 Silver, 48	Bikes	Silver	48	546	226.00
2	SO69560	2020-05-30	513	ML Mountain Frame-W - Silver, 46	Components	Silver	46	100	218.45
3	SO69560	2020-05-30	594	Mountain-500 Silver, 48	Bikes	Silver	48	100	113.00
4	SO69539	2020-05-28	243	HL Road Frame - Red, 44	Components	Red	44	529	858.90
5	SO69539	2020-05-28	378	Road-250 Black, 52	Bikes	Black	52	529	1466.01
6	SO69541	2020-05-28	594	Mountain-500 Silver, 48	Bikes	Silver	48	661	113.00
7	SO69542	2020-05-28	243	HL Road Frame - Red, 44	Components	Red	44	317	1717.80
8	SO69544	2020-05-28	243	HL Road Frame - Red, 44	Components	Red	44	666	3435.60
9	SO69545	2020-05-28	378	Road-250 Black, 52	Bikes	Black	52	436	5864.04
10	SO69532	2020-05-27	594	Mountain-500 Silver, 48	Bikes	Silver	48	312	113.00
11	SO69532	2020-05-27	513	ML Mountain Frame-W - Silver, 46	Components	Silver	46	312	436.90
12	SO69533	2020-05-27	594	Mountain-500 Silver, 48	Bikes	Silver	48	476	226.00

COMPARE FACT AND DIMENSION TABLES

Characteristic	Dimension table	Fact table
Model purpose	Stores business entities	Stores events or observations
Table structure	Includes a key column and descriptive columns for filtering and grouping	Includes dimension key columns and numeric measure columns that can be summarized
Data volume	Typically, contains fewer rows (relative to fact tables)	Can contain numerous rows
Query purpose	To filter and group	To summarize

Relationship & Cardinality

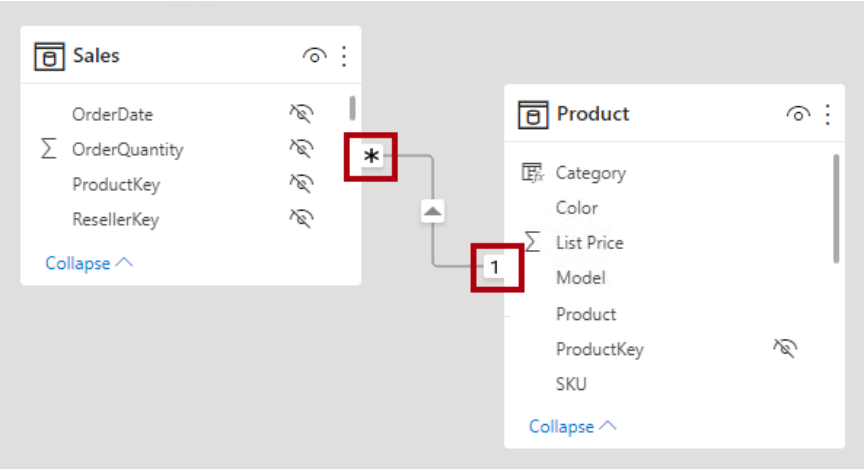
Each model relationship is defined by a cardinality type. There are four cardinality-type options, representing the data characteristics of the "from" and "to" related columns. The "one" side means the column contains unique values; the "many" side means the column can contain duplicate values.

1. One-to-one (1:1)

2. One-to-many (1:*) Many-to-one (*:1)

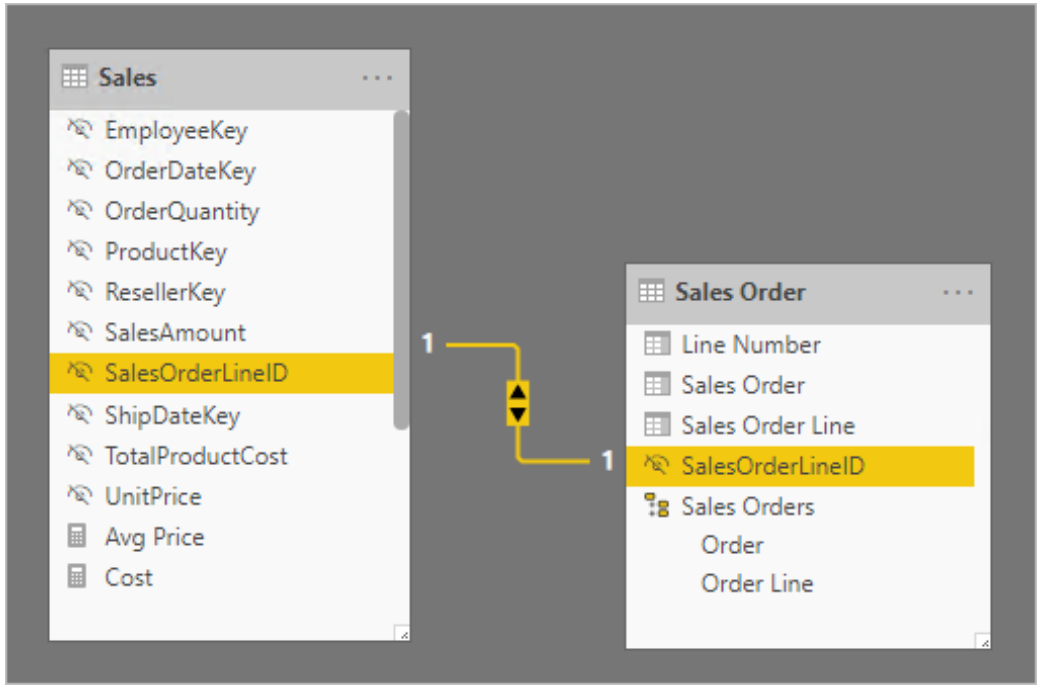
3. Many-to-many (*:*)

One-to-many (1:*) Many-to-one (*:1)



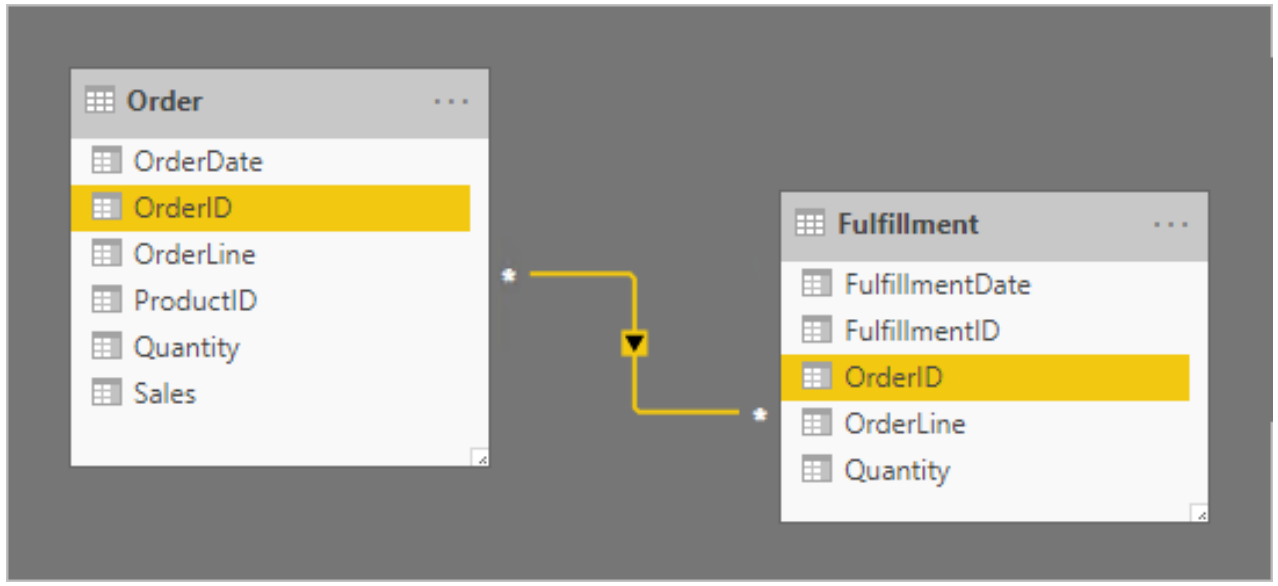
The one-to-many and many-to-one cardinality options are essentially the same, and they're also the most common cardinality types meaning there should be a single instance of each primary key while potentially accommodating multiple instances of each foreign key.

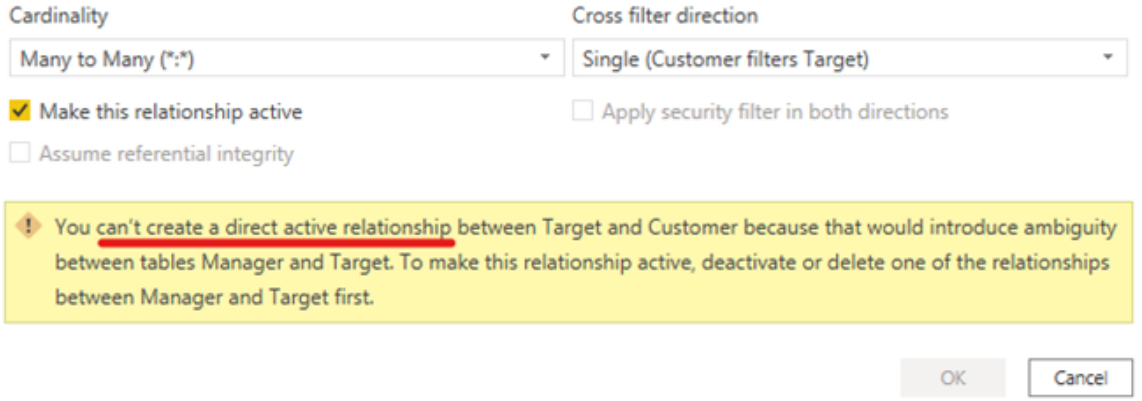
One-to-one (1:1)



Maintaining one-to-one relationships may lead to redundant data storage since the tables involved often contain overlapping information. This redundancy can increase memory usage and make the data model more challenging to manage.

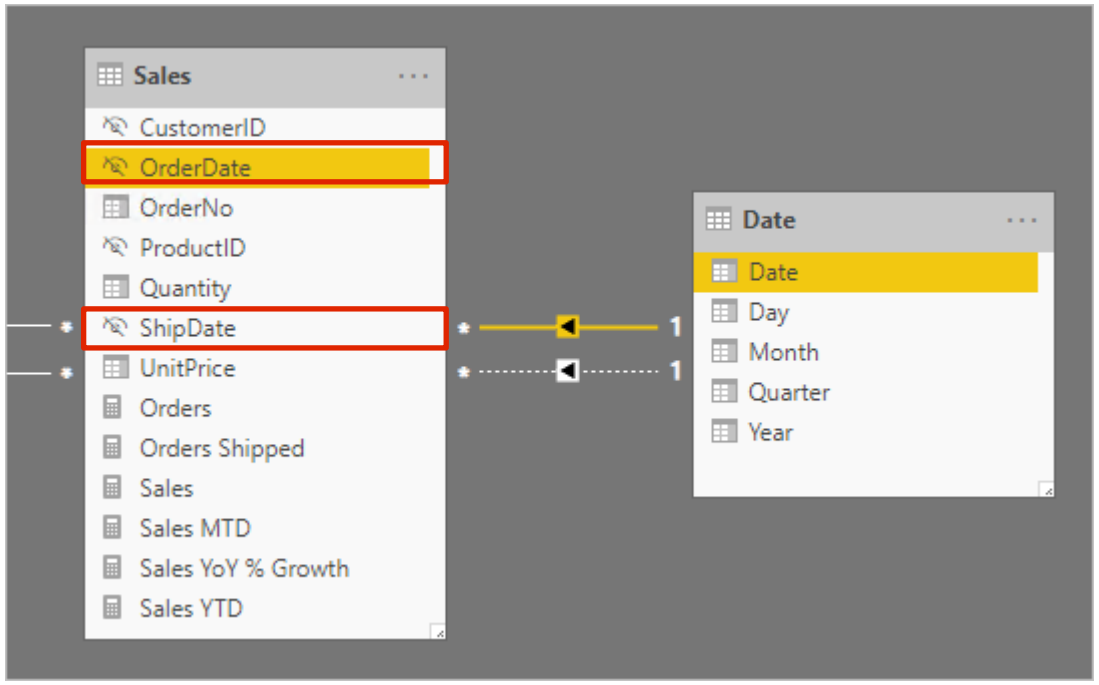
Many-to-many (*:*)





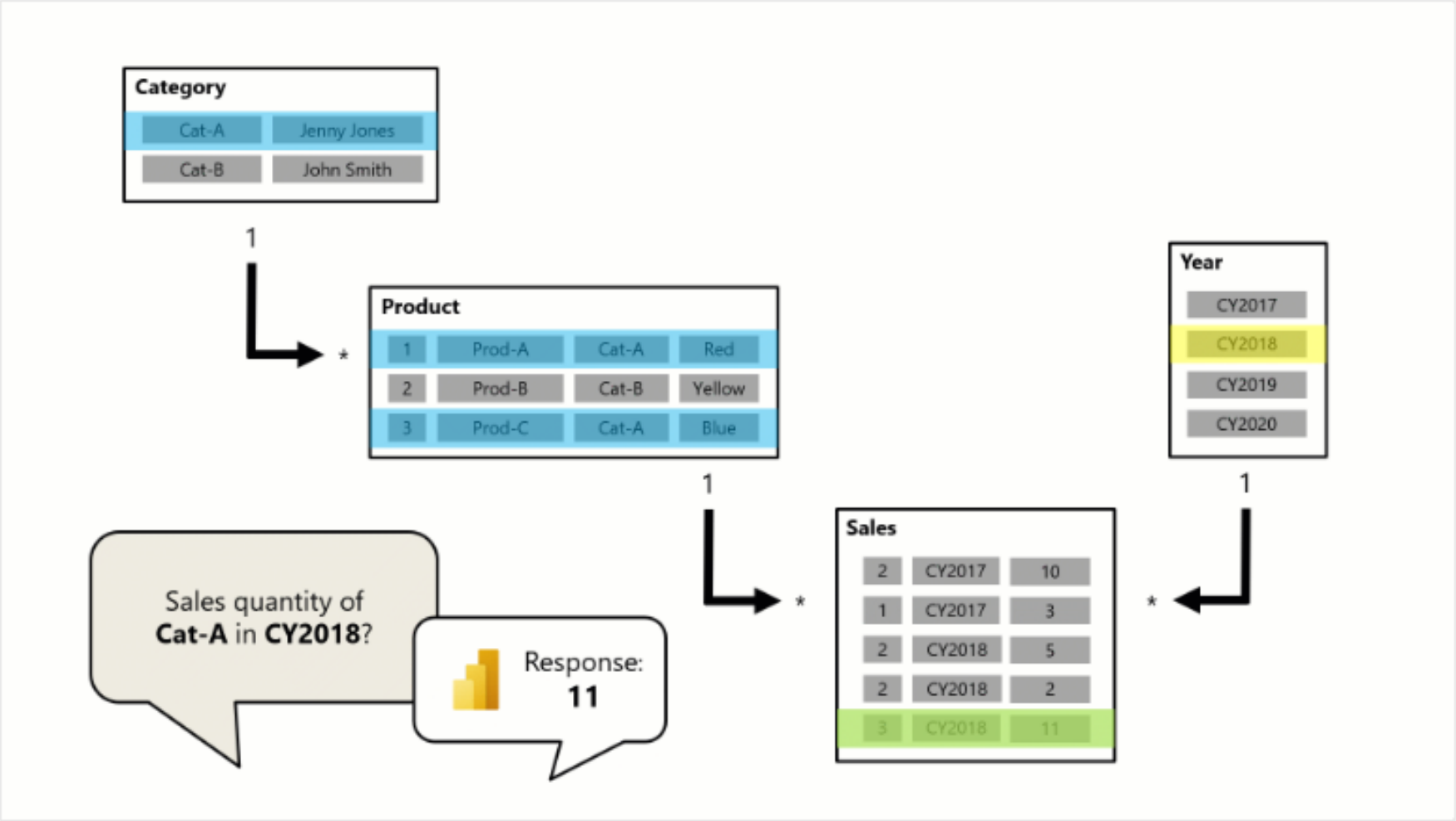
A many-to-many relationship means both columns can contain duplicate values. This cardinality type is infrequently used. It's typically useful when designing complex model requirements.

Inactive Relationships

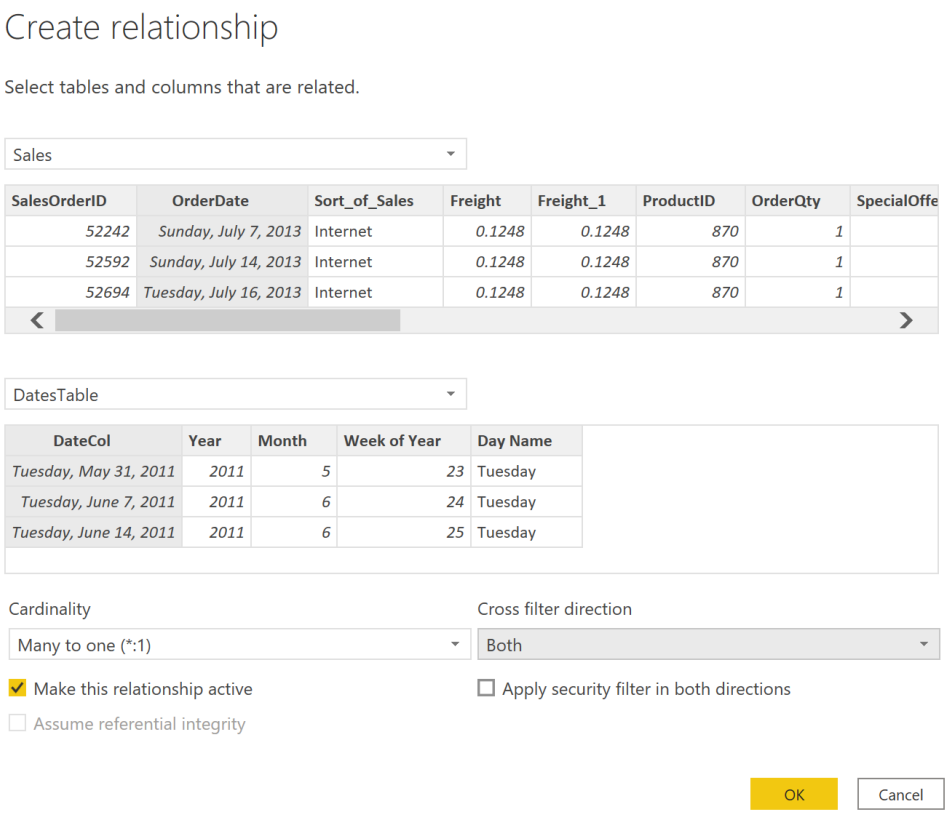


Inactive relationships in Power BI are relationships between tables that are temporarily turned off for filtering and calculations to avoid conflicts in data analysis. They can be activated as needed for specific tasks.

Filter Propagation



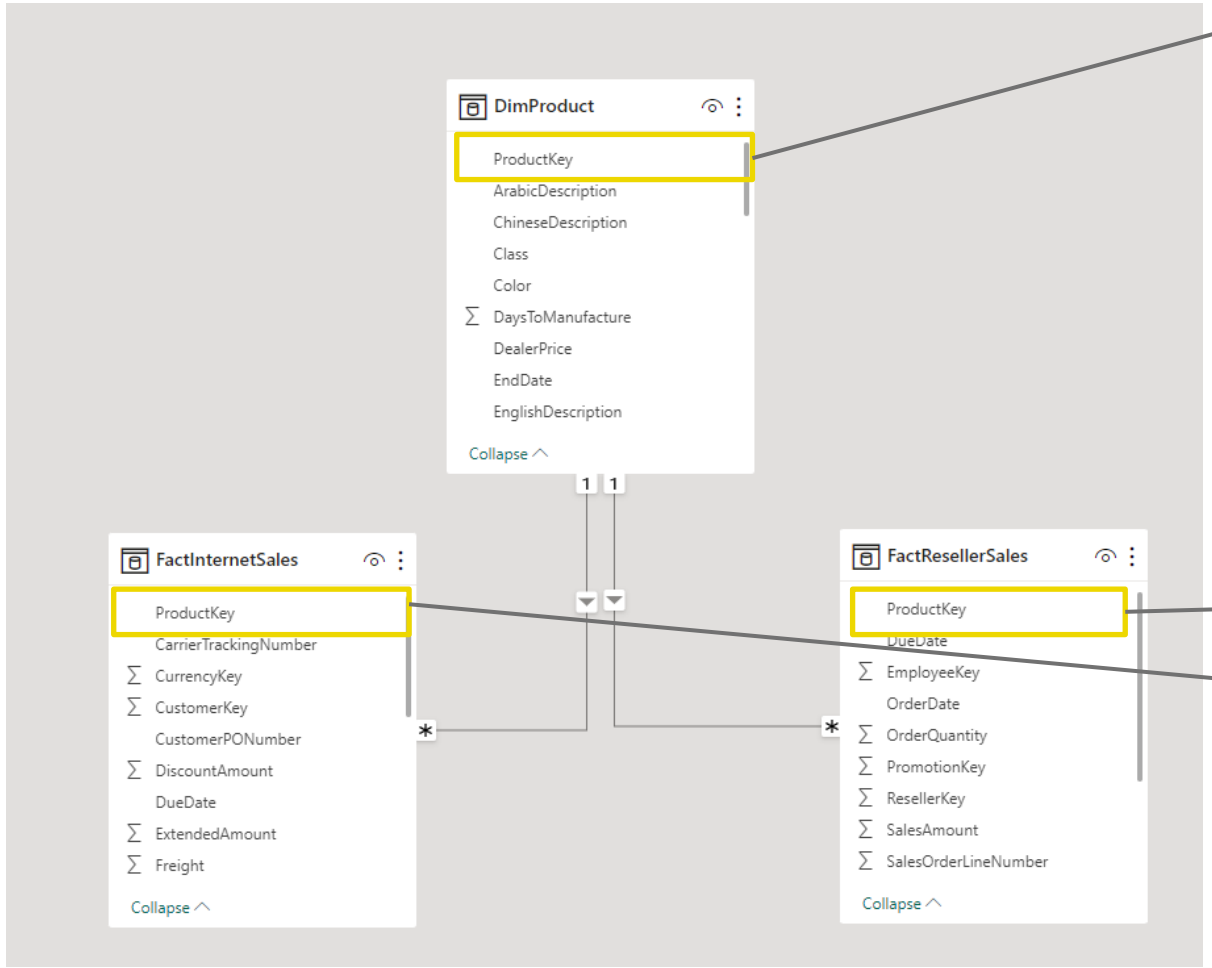
CROSS FILTER DIRECTION



Cardinality type	Cross filter options
One-to-many (or Many-to-one)	Single Both
One-to-one	Both
Many-to-many	Single (Table1 to Table2) Single (Table2 to Table1) Both

Each model relationship is defined with a cross-filter direction. Your setting determines the direction(s) that filters will propagate. The possible cross-filter options are dependent on the cardinality type.

Filter Flow Example



ProductKey	Internat Sales Amount	Reseller Sales Amount
212		11,387.16
213		29,411.29
214	78,027.70	36,475.84
215		12,105.16
216		32,259.43
217	72,954.15	40,440.11
218		6,148.50
219		513.00
220		13,338.11
221		34,246.14
222	74,353.75	40,546.52
223		5,093.95
224		11,792.35
225	19,688.10	13,772.12
228	21,445.71	
Total	**29,358,677.89**	**77,548,570.20**

ProductKey	Internat Sales Amount	Reseller Sales Amount
214	78,027.70	77,548,570.20
217	72,954.15	77,548,570.20
222	74,353.75	77,548,570.20
225	19,688.10	77,548,570.20
228	21,445.71	77,548,570.20
231	22,095.58	77,548,570.20
234	22,595.48	77,548,570.20
237	20,645.87	77,548,570.20
310	1,202,298.72	77,548,570.20
311	1,005,493.87	77,548,570.20
312	1,205,876.99	77,548,570.20
313	1,080,637.54	77,548,570.20
314	1,055,589.65	77,548,570.20
320	13,282.90	77,548,570.20
321	43,064.45	77,548,570.20
Total	**29,358,677.89**	**77,548,570.20**

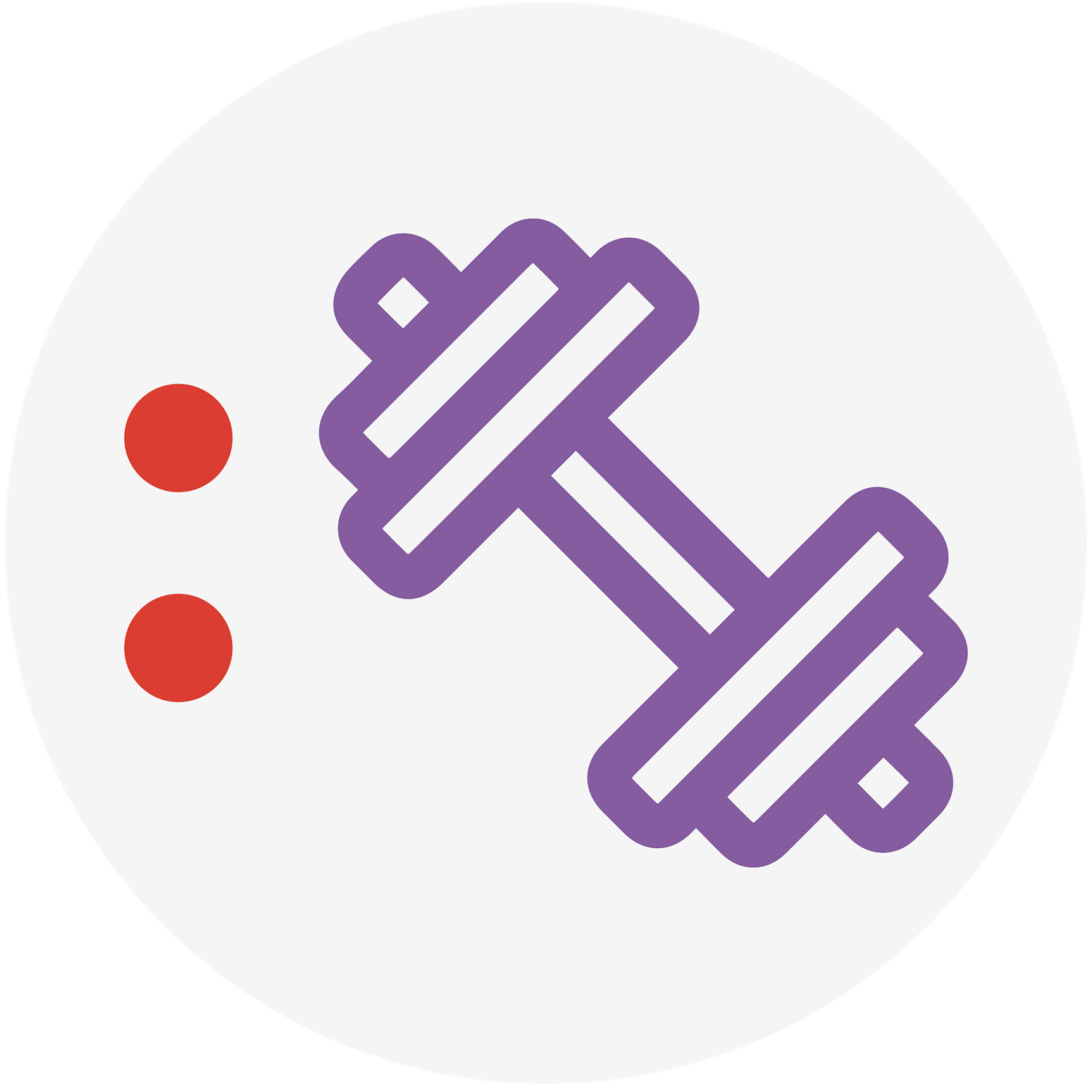
To filter both facts tables correctly the ProductKey from the Dimension table must be used for the correct results

ProductKey	Internat Sales Amount	Reseller Sales Amount
212	29,358,677.89	11,387.16
213	29,358,677.89	29,411.29
214	29,358,677.89	36,475.84
215	29,358,677.89	12,105.16
216	29,358,677.89	32,259.43
217	29,358,677.89	40,440.11
218	29,358,677.89	6,148.50
219	29,358,677.89	513.00
220	29,358,677.89	13,338.11
221	29,358,677.89	34,246.14
222	29,358,677.89	40,546.52
223	29,358,677.89	5,093.95
224	29,358,677.89	11,792.35
225	29,358,677.89	13,772.12
229	29,358,677.89	12,603.08
Total	**29,358,677.89**	**77,548,570.20**

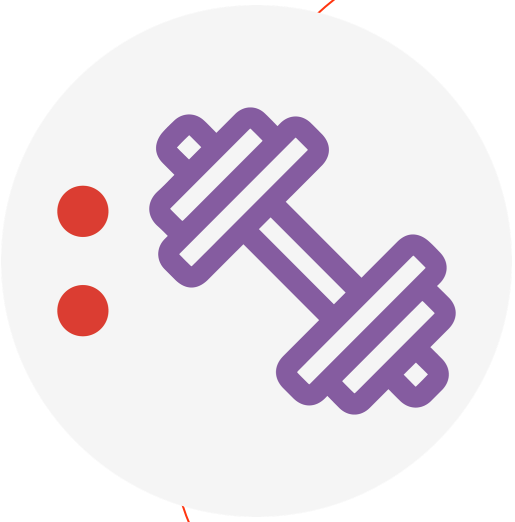
DATA MODEL BEST PRACTICES

- ✓ Use a star schema with
- ✓ Contain relationships with one way filters (vs. bidirectional)
- ✓ Contain tables that each serve a specific purpose, including data (fact) tables and lookup (dim) tables
- ✓ Only include the data you need for analysis (no redundant or unnecessary records or fields)
- ✓ Split out individual date and time components from DateTime fields

Live Demo



Exercise 1



KC3

Lesson 4
Create Model Calculations using DAX

After completing this module, students will be able to:

- Understand DAX functions
- Creating a calculated Data table using DAX
- Use DAX for formulas and expressions
- Create calculated tables and measures
- Build dynamic measures

DAX (Data Analysis Expressions)

DAX can be used to create three types of calculations:

- Calculated table
- Calculated column
- Measure
- DAX can also be used to define row-level security (RLS) rules

Common DAX Functions

Math & Stats

Common Examples:

- SUM
- AVERAGE
- MAX/MIN
- DIVIDE
- COUNT/COUNTA
- COUNTROWS
- DISTINCTCOUNT

- SUMX
- AVERAGEX
- MAXX/MINX
- COUNTX

Logic

Common Examples:

- IF
- IFERROR
- AND
- OR
- NOT
- SWITCH
- TRUE
- FALSE

Text

Common Examples:

- CONCATENATE
- FORMAT
- LEFT/MID/RIGHT
- UPPER/LOWER
- PROPER
- LEN
- SEARCH/FIND
- REPLACE
- REPT
- SUBSTITUTE
- TRIM
- UNICHAR

Filter

Common Examples:

- CALCULATE
- FILTER
- ALL
- ALLEXCEPT
- RELATED
- RELATEDTABLE
- DISTINCT
- VALUES
- EARLIER/EARLIEST
- HASONEVALUE
- HASONFILTER
- ISFILTERED
- USERELATIONSHIP

Time Intelligence

Common Examples:

- TOTALMTD
- TOTALQTD
- TOTALYTD
- SAMEPERIODLASTYEAR
- PARALLELPERIOD
- DATESBETWEEN
- DATEADD
- DATESINPERIOD

Common DAX Operators

Arithmetic Operator	Meaning	Example
+	Addition	2 + 7
-	Subtraction	5 – 3
*	Multiplication	2 * 6
/	Division	4 / 2
^	Exponent	2 ^ 5

Comparison Operator	Meaning	Example
=	Equal to	[City]="Boston"
>	Greater than	[Quantity]>10
<	Less than	[Quantity]<10
>=	Greater than or equal to	[Unit_Price]>=2.5
<=	Less than or equal to	[Unit_Price]<=2.5
<>	Not equal to	[Country]<>"Mexico"

Text/Logical Operator	Meaning	Example
&	Concatenates two values to produce one text string	[City] & " " & [State]
&&	Create an AND condition between two logical expressions	([State]="MA") && ([Quantity]>10)
|| (double pipe)	Create an OR condition between two logical expressions	([State]="MA") || ([State]="CT")
IN	Creates a logical OR condition based on a given list (using curly brackets)	'Store Lookup'[State] IN { "MA", "CT", "NY" }

Standard Aggregation Measures

Typically, numeric columns are summarized using aggregation functions.

Common examples of summarizing:

- Sum
- Count
- Average
- Min and Max

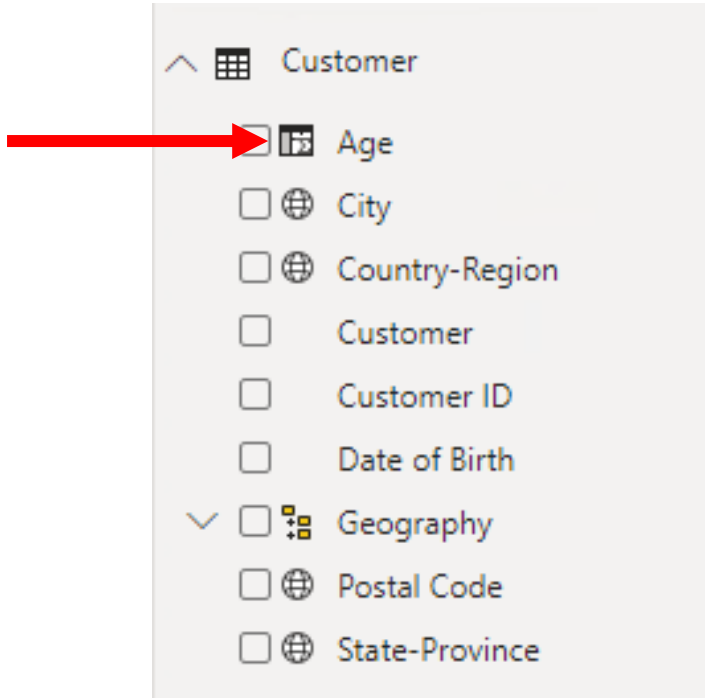
Column references always enclose the column name in square brackets

```
Total Sale = sum('Sales By Country Files’[Sales])
```

```
Sale Quantity = Average('Sales By Country Files'[Quantity])
```


DAX Calculated column

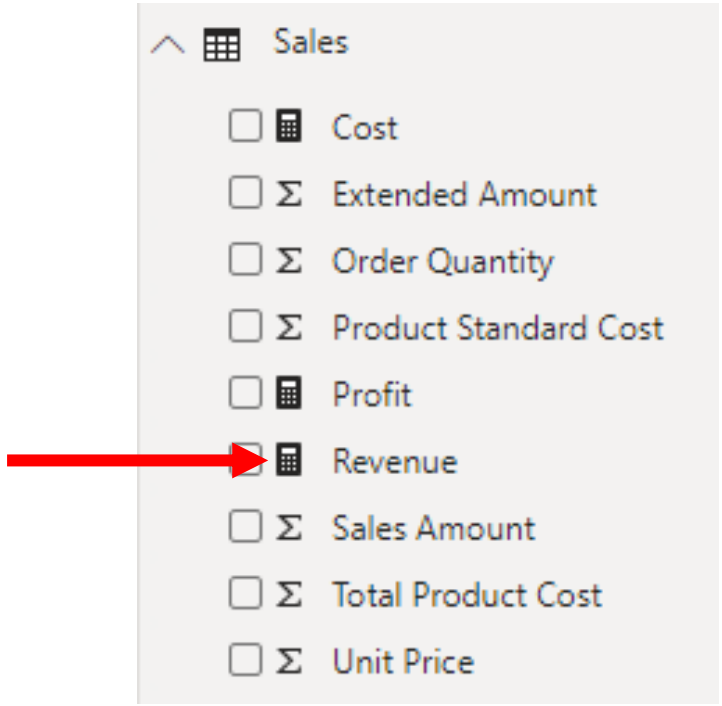
- A **Calculated column** adds a new column to a table
- A formula is evaluated for each row in the table
- The formula must return a single value
- It is only evaluated when the model is refreshed*
- It increases the storage size of the model*



*Except when added to a DirectQuery table

DAX Calculated measures

- A Measure summarizes model data
- The formula must return a single value
- It is evaluated at query-time
- Results are never stored in the model

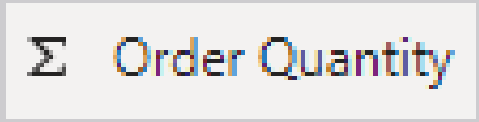


Calculated columns vs. measures

There are differences between calculated column and measure

	Calculated Columns	Measures
Purpose	Extend table	Summarize model data
Evaluation	Row context at data refresh-time	Filter context at query-time
Storage	Stores value in each table row	Never stores value
Visual use	Filter, group, or summarize	Designed to summarize

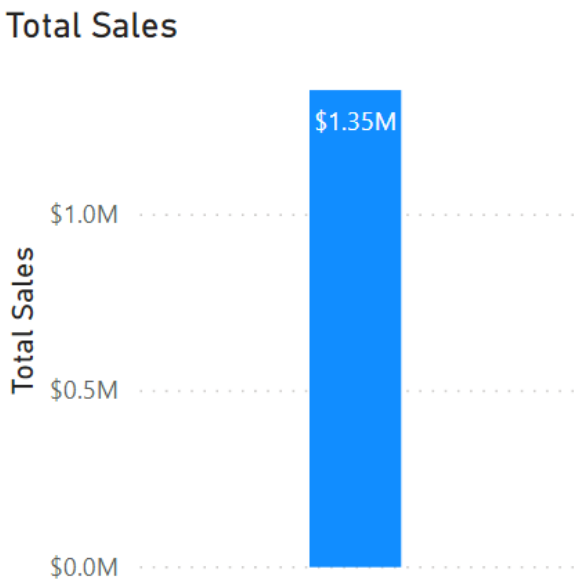
Implicit & Explicit Measures

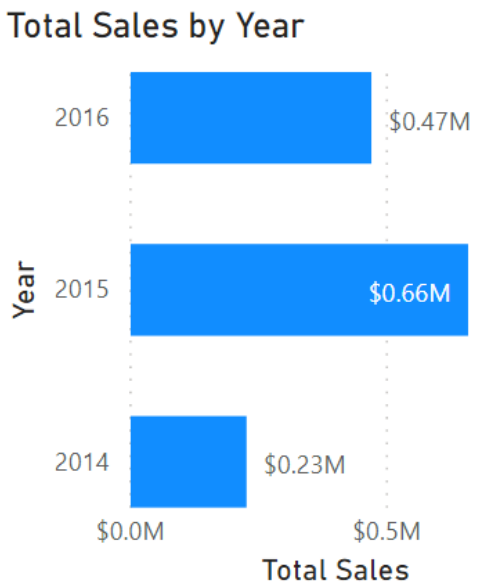
	Benefits	Limitations
Implicit Measures 	• Accessible to non-technical users. • Enables self-service analytics. • Easy to modify and adjust.	• Limited flexibility for complex calculations. • Limited control and customization • May lack granularity and detailed data analysis capabilities.
Explicit Measures 	• Greater flexibility and customization • Requires deeper understanding of DAX and the data model. • Can be used in Analyze with Excel export option	• Requires advanced technical knowledge of DAX. • Time-consuming to create and manage complex calculations. • Excessive use can impact report performance.

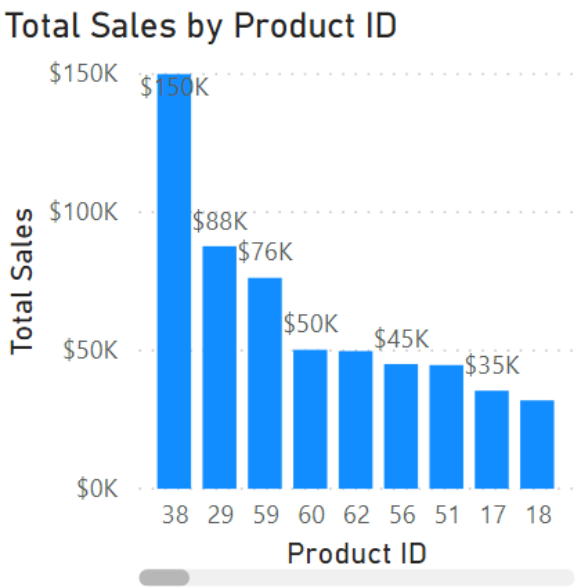
Understand context when using DAX

How context affects DAX measures is a difficult concept to comprehend. The ensuing visuals will demonstrate how context affects DAX measures so you can see how they interact together.

The following three visuals use the exact same DAX measure: Total Sales.







DAX Formatting

From

```
USSales = CALCULATE([Total Sales],FILTER(Reseller,Reseller[ResellerCountry]= "United States“))
```

To

```
USSales =
CALCULATE (
    [Total Sales],
    FILTER ( Reseller, Reseller[ResellerCountry] = "United States" )
)
```

*Use **shift** + **enter** to split out and indent each component of your DAX formulas to make them more readable*

DAX FORMATTER

https://www.daxformatter.com/

Formula Commenting

```
Sales Previous Year % Change =
var CY = [Total Sales] // Current Year Sales
var PY = [Sales Last Year] // Last Years Sales
RETURN
DIVIDE(CY-PY,PY)
```

Comment Type	Marker
Single Line Comment	-- or //
Multi Line Comment	/* … */

Adding comments to your code can assist fellow users in understanding your script, and this practice proves especially beneficial for intricate queries featuring numerous lines and nested functions.

Using Variables

Variables can be defined in an expression to make writing DAX easier

Main benefits:

- Improves readability of a formula
- Improves performance when an expression is used multiple times
- Allows testing portions of a complex formula, by returning only a variable for review

```
VAR = <name> RETURN <result_expression>

Example:
VAR RevenuePriorYear = CALCULATE( [Revenue],
SAMEPERIODLASTYEAR('Date'[Date]) )

RETURN

DIVIDE( [Revenue Measure] -RevenuePriorYear,
RevenuePriorYear )
```

DAX Quick Measures

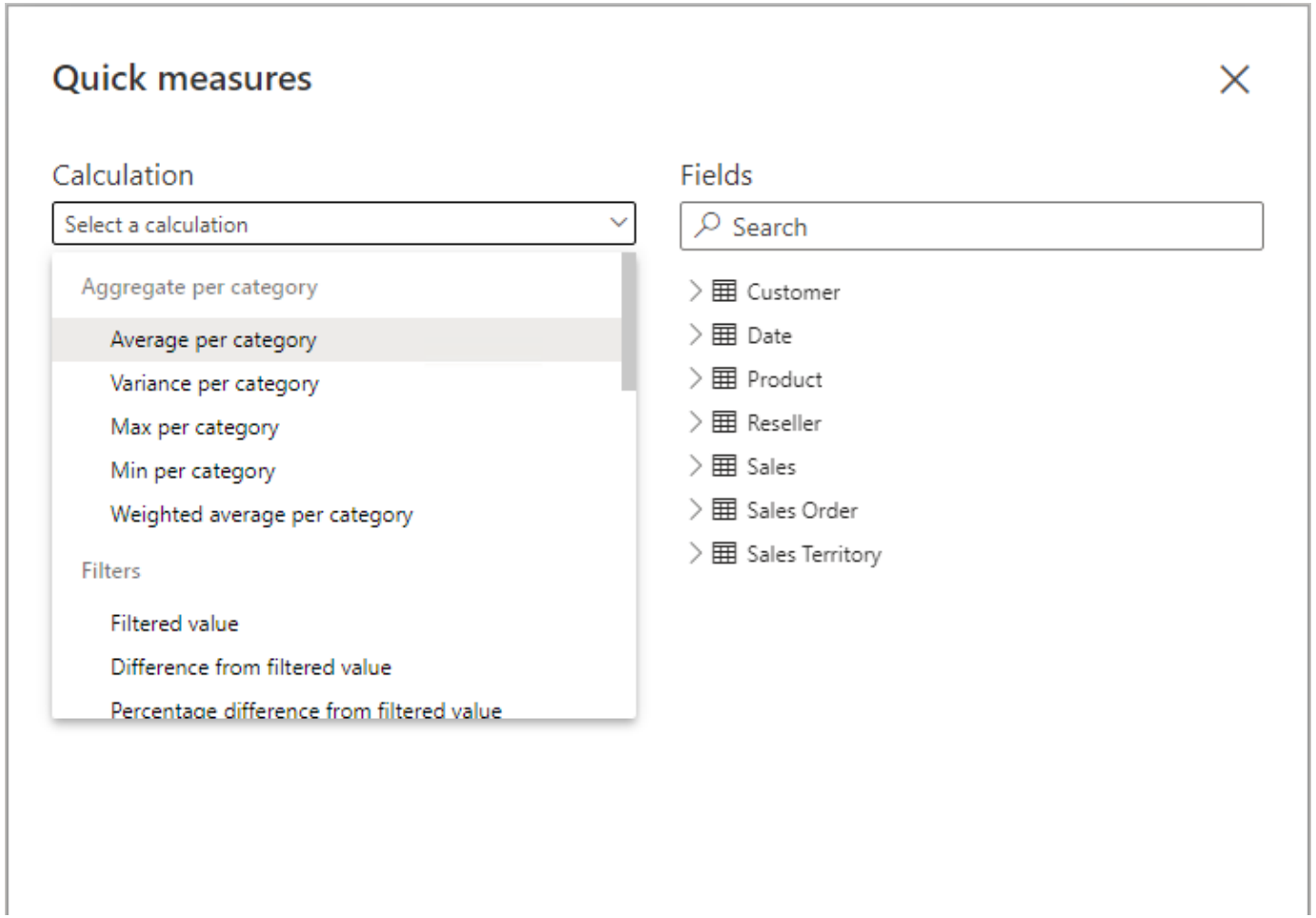
Power BI Desktop supports creating
Quick Measures

Benefits:

- Quick and easy
- Common calculation templates are available
- Automatically creates the DAX expression
- No need to understand any DAX

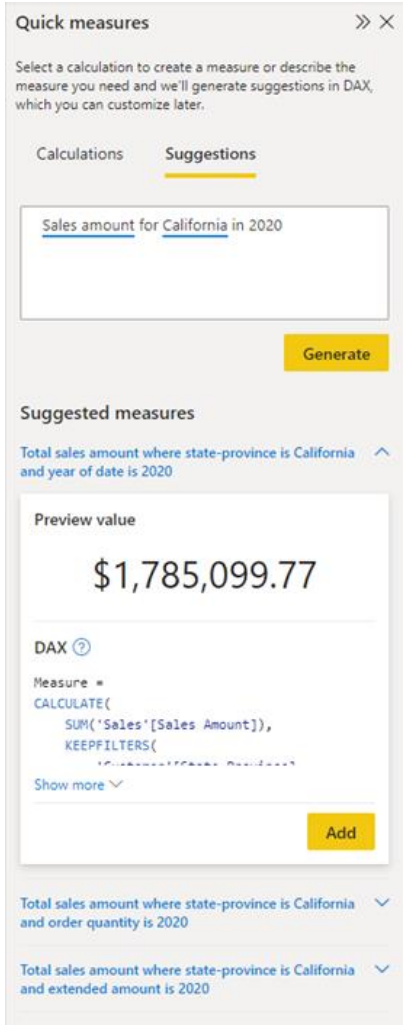
Once created, they are like any other explicit measure

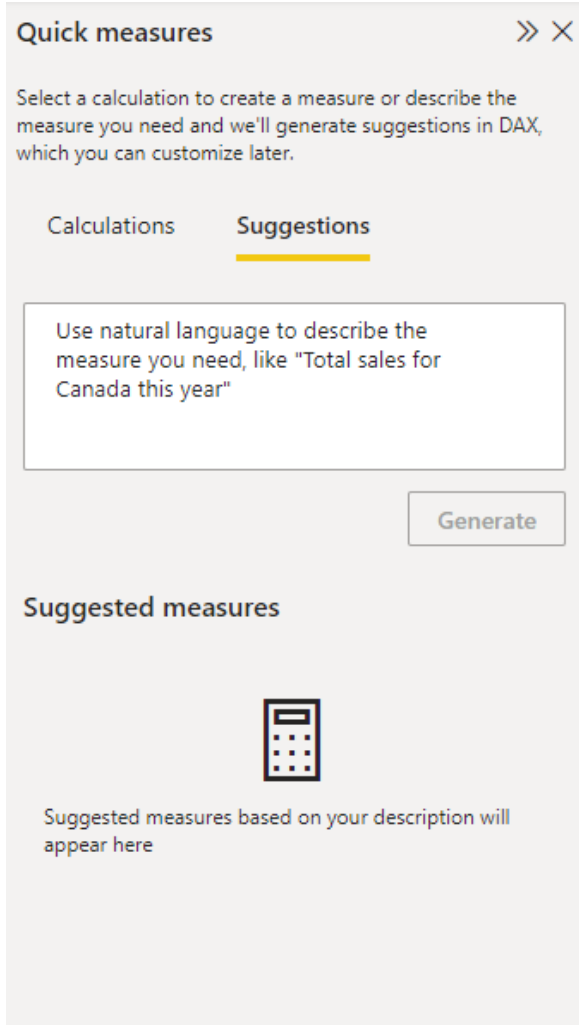
- To modify the measure, simply edit the formula



DAX Quick Suggestion AI generation (New)

Quick measure suggestions assist in creating DAX measures using natural language instead of templates or writing DAX from scratch.





Iterator functions

Iterator functions enumerate all rows of a given table and evaluate a given expression for each row

- They provide flexibility over how calculations summarize data
- All iterator functions require Table and Expression
- The expression must return scalar or single value
- Iterator functions are easily identified by the appended "X"

For example:

- Common aggregation: SUMX, AVERAGEX, COUNTX, MINX, and MAXX
- Special: CONCATENATEX, and RANKX

Iterator functions

Example

```
Measure =
SUMX(
        <Table>,
        <Expression>
)
```

```
Revenue =
SUMX(
        Sales,
        Sales[Order Quantity] * Sales[Unit Price]
)
```

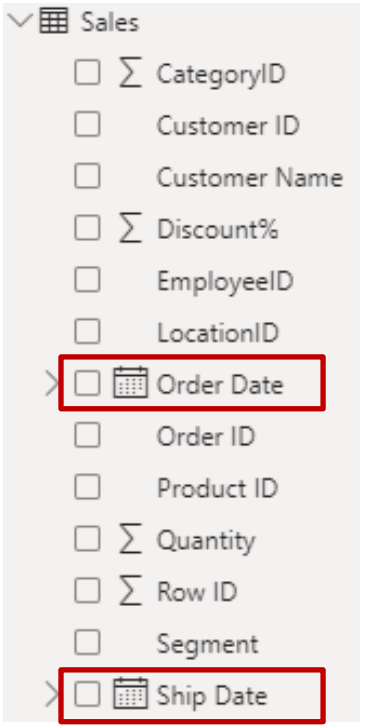
Iterator functions example

yKey	Order Quantity	Unit Price	
5	2	$5.19	10.38
5	4	$20.19	80.76
5	1	$419.46	419.46
5	1	$874.79	874.79
6	1	$809.76	809.76
6	1	$714.70	714.70
6	2	$714.70	1,429.40
6	4	$5.19	...

∑ **208,202.17**

Automatic Date Table

By default, Power BI automatically generates a concealed date table for any table that includes a Date or **DateTime** column on one side of a relationship. These automatically generated calendars encompass all dates up to the end of the year, irrespective of the actual date range within the table.



Date	Day	MonthNo	Month	QuarterNo	Quarter	Year
1/01/2017 00:00:00	1	1	January	1	Q1	2017
1/02/2017 00:00:00	2	1	January	1	Q1	2017
1/03/2017 00:00:00	3	1	January	1	Q1	2017
1/04/2017 00:00:00	4	1	January	1	Q1	2017
1/05/2017 00:00:00	5	1	January	1	Q1	2017
1/06/2017 00:00:00	6	1	January	1	Q1	2017
1/07/2017 00:00:00	7	1	January	1	Q1	2017
1/08/2017 00:00:00	8	1	January	1	Q1	2017
1/09/2017 00:00:00	9	1	January	1	Q1	2017
1/10/2017 00:00:00	10	1	January	1	Q1	2017
1/11/2017 00:00:00	11	1	January	1	Q1	2017
1/12/2017 00:00:00	12	1	January	1	Q1	2017

Automatically created a hidden date table in the background containing all these columns

PROS & CONS AUTOMATIC DATE TABLES

PROS

- Automatically generated
- No code or additional table required
- Built-in date Hierarchies aren't automatically generated

Cons

- Hidden background table that not be modified
- Inflates the Power BI file size due to every date field in your table having a auto-date table
- Not a centralized date dimension table

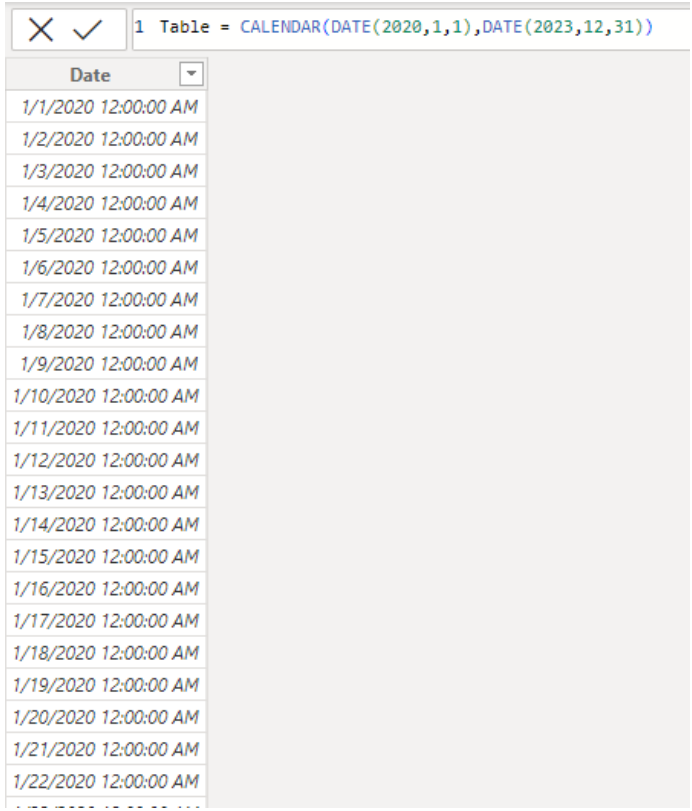
Date Dimension Table Requirements Checklist

- ✓ It should encompass all days corresponding to the years presented in your fact tables.
- ✓ It is necessary to designate at least one field as a Date or DateTime data type.
- ✓ Duplicate dates or datetime values are not permissible.
- ✓ If utilizing a time component within a date column, all times must be uniform (e.g., 12:00 PM).
- ✓ While not mandatory, it is recommended to label it as a date table for optimal use.

CALENDAR Function

- Returns a table with one column of all dates between start and end date
- Dates can also be referenced from date fields in other tables as seen in example 2

```
=CALENDAR(StartDate,EndDate)
```



Example 1: CALENDAR DATE(2020,01,01), DATE (2023,12,31)

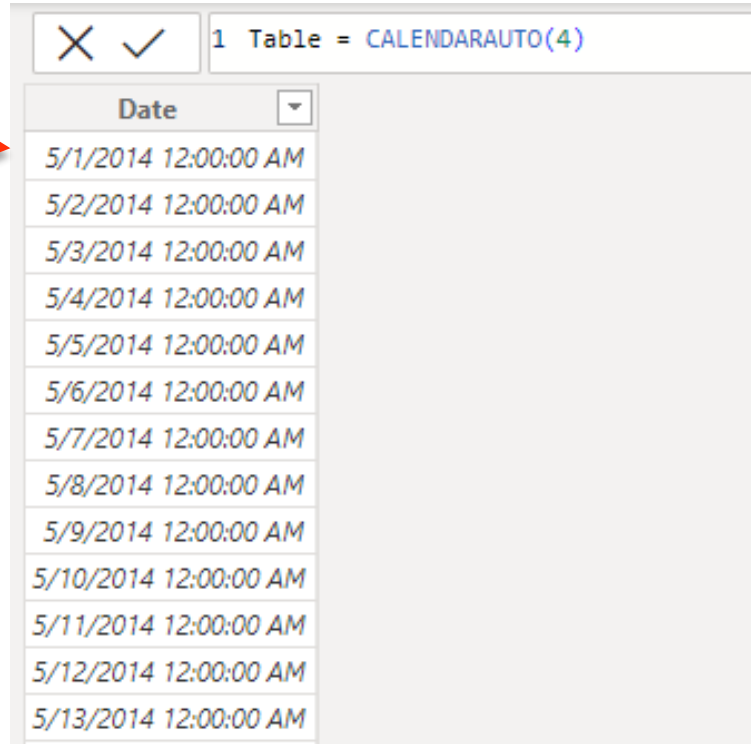
Example 2:

```
CALENDAR
DATE( YEAR ( MIN (Sales [Order Date] )), 1,1)

DATE( YEAR ( MAX (Sales [Order Date] )), 12,31)
```

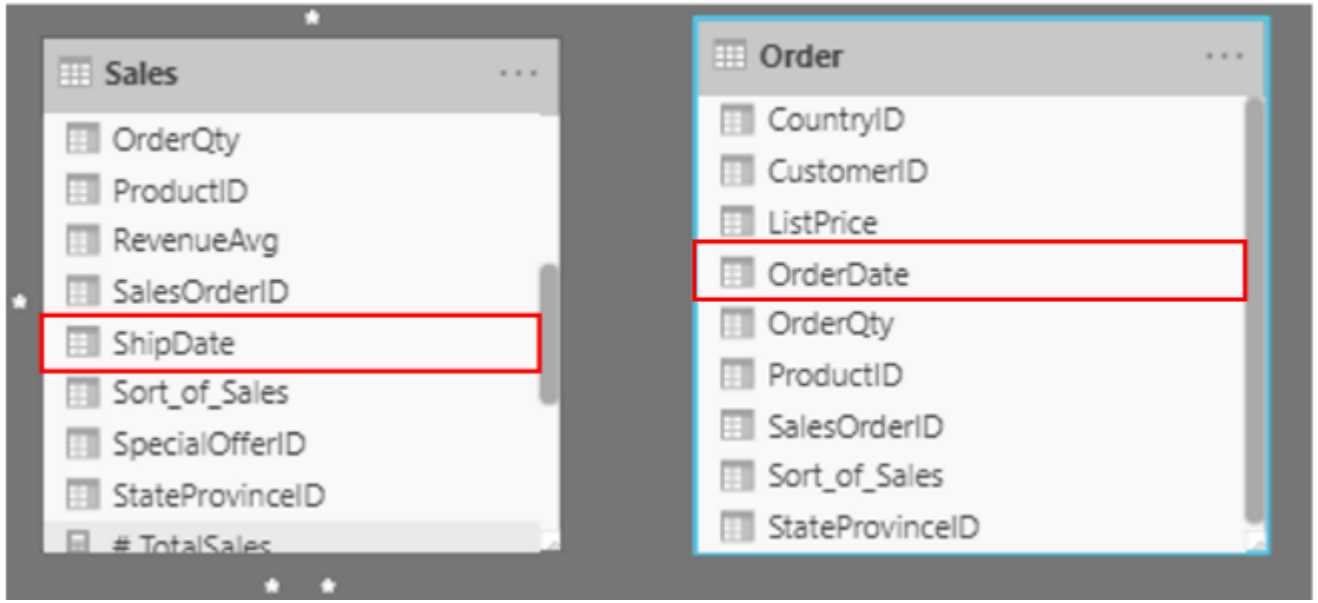

CALENDARAUTO Function

Returns a table with one column of dates based on a fiscal year end month. The Range of dates is calculated automatically based on data in the model

```
Example 1: CALENDARAUTO(4)


Example 2:

Calendar Table =

VAR MinYear = YEAR( MIN( 'Country'[Total Sales]))
VAR MaxYear= YEAR( MAX('Country'[Total Sales]))

RETURN

FILTER(CALENDARAUTO(),
YEAR( [Date] ) >= MinYear&&YEAR( [Date] ) <= MaxYear)
```



The Date Dimension Table



Date	Year	Month	Month Short Name	Month Long Name	Quater	Day of the week	WeekNum
7/1/2020	2020	7	Jul	July	Q3	Wed	27
7/2/2020	2020	7	Jul	July	Q3	Thu	27
7/3/2020	2020	7	Jul	July	Q3	Fri	27
7/4/2020	2020	7	Jul	July	Q3	Sat	27
7/5/2020	2020	7	Jul	July	Q3	Sun	27
7/6/2020	2020	7	Jul	July	Q3	Mon	28
7/7/2020	2020	7	Jul	July	Q3	Tue	28
7/8/2020	2020	7	Jul	July	Q3	Wed	28

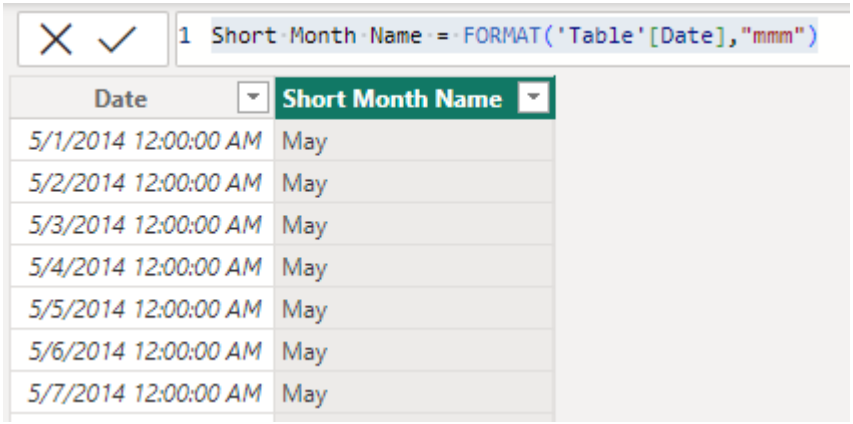
Column Name	Fonction
DAX Defined Date	Dates = CALENDAR(DATE(2011, 5, 31), DATE(2022, 12, 31))
DAX Auto Date	Dates = CALENDARAUTO()
DAX Year	Year = YEAR(Dates[Date])
DAX Month	MonthNum = MONTH(Dates[Date])
DAX Week number	WeekNum = WEEKNUM(Dates[Date])
DAX Day of the week	DayoftheWeek = FORMAT(Dates[Date], "DDDD")

Formatting your date

By using the DAX **FORMAT** function to specify date/time formatting you can specify the format output you would like for your fields.

```
FORMAT(<value>, <format_string>[, <locale_name>])
```

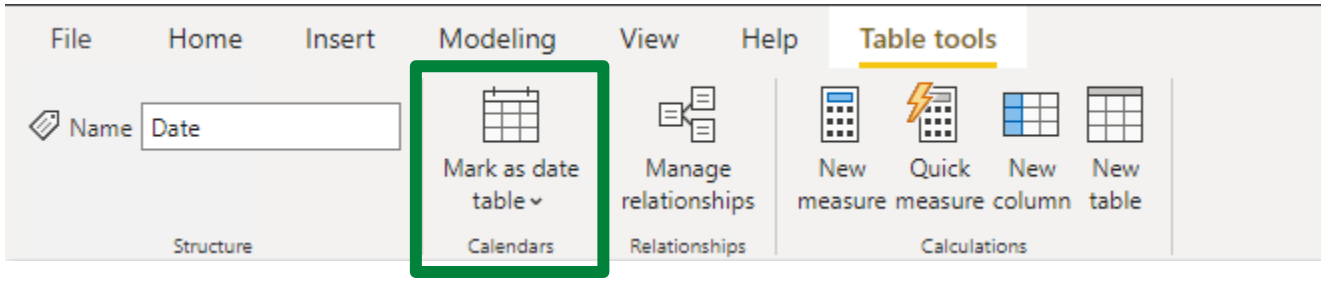
```
Short Month Name = FORMAT('Table'[Date],"mmm")
```



Date	Short Month Name
5/1/2014 12:00:00 AM	May
5/2/2014 12:00:00 AM	May
5/3/2014 12:00:00 AM	May
5/4/2014 12:00:00 AM	May
5/5/2014 12:00:00 AM	May
5/6/2014 12:00:00 AM	May
5/7/2014 12:00:00 AM	May

Code	Description
dddd	Display the full day name
ddd	Display the short day name
mmmm	Show full month name
mmm	Show abbreviated month name
mm	Display the month as a two digit number
q	Display the quarter of the year
yyyy	Show the year as a four digit number

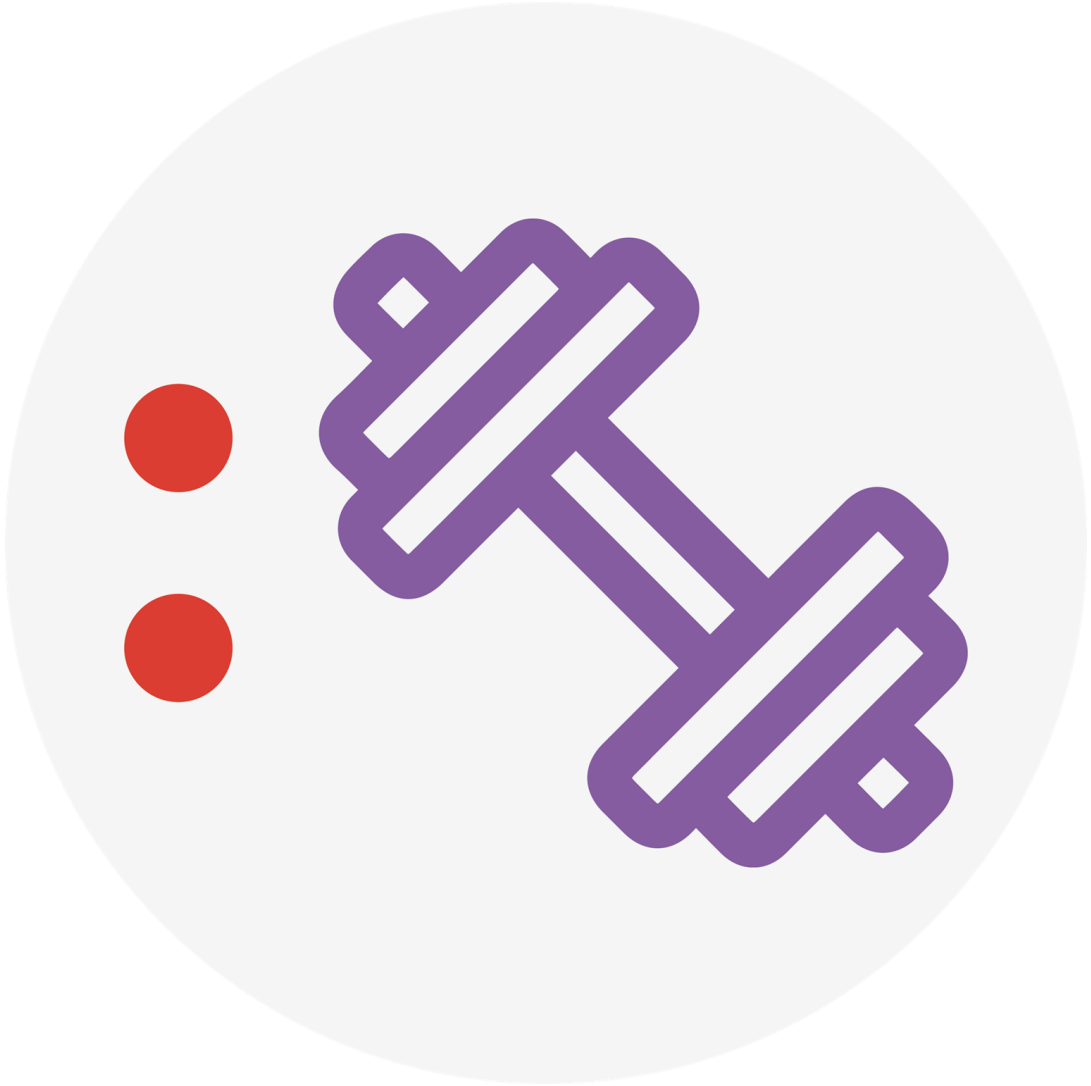
Marking Date Dimension Table



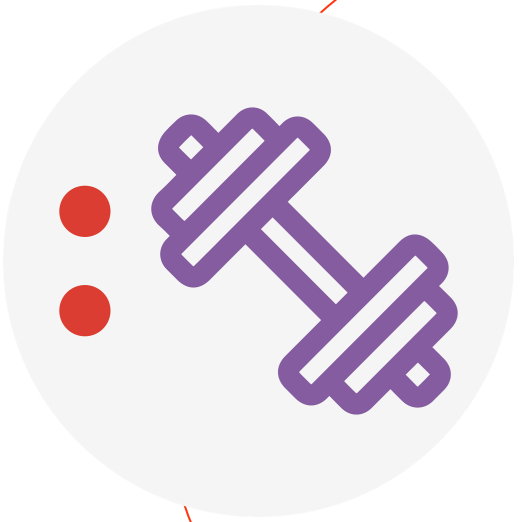
- To use the Time Intelligence functions, a table must be marked as a date table
  - It also allows the model tables to use a non-date type key
  - The date key can be in ISO format, for example, *20210630*

- Multiple model tables can be marked as date tables

Live Demo



Exercise 2



KC4

DATASOCIETY:

Power BI Intermediate

Day 3

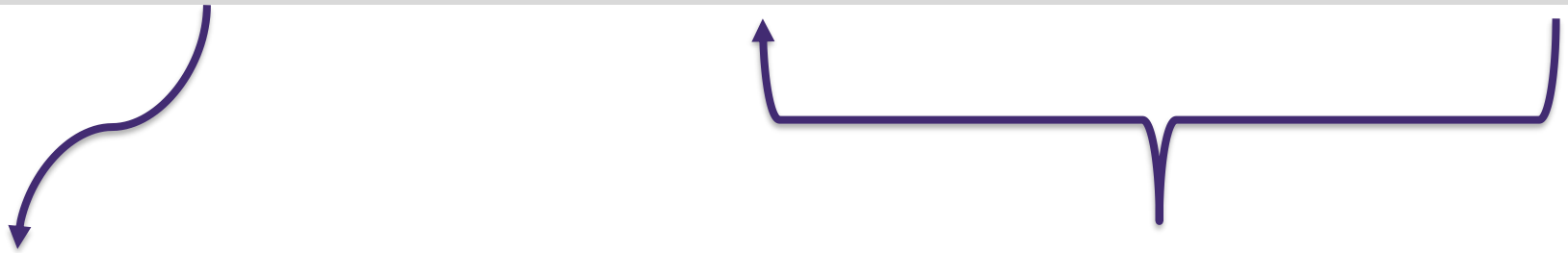
Lesson 5
CALCULATE Function & Modifiers

After completing this module, students will be able to:

- Use the Calculate function
- Use filter modifier functions within Calculate
- Examine filter context

DAX Calculate Function

```
CALCULATE ( <expression> , [[<filter1>], <filter2>]…)
```



Name of an existing measure, or a DAX formula for a valid measure

Example

[Total Sales] existing measure

SUM(Regionsales[Sales]) a DAX formula

List of simple Boolean (True/False) filter expressions.
Note you cannot create filters based on a measure

Example

Product[color]="Blue"

Calendar[Year]>2021

DAX Calculate Function

Boolean expression filters must evaluate as TRUE or FALSE
Each filter:

- Can reference only a single column
- Can not reference measures
- Can not use functions that scan or return table

1

```
Revenue Red =
CALCULATE([Revenue], 'Product'[Color] = "Red")
```

2

```
Revenue Red =
CALCULATE(
          [Revenue],
          FILTER(
                    'Product',
                    'Product'[Color] = "Red"
          )
)
```


Table Expression Filters

Table expression filters apply a table object as a filter
Each filter:

- Can be a reference to model table
- Can be a DAX function that returns a table object
- If columns are not in filter context, new filters are added to filter context
- If columns are already in filter context, existing filters are overwritten

```
Revenue High Margin Products =
CALCULATE(
        [Revenue],
        FILTER(
                'Product',
                'Product'[List Price] > 'Product'[Standard Cost] * 2
        )
)
```

CALCULATE Filter Modifiers

When using CALCULATE, it is possible to pass in filter modification functions
Filter modifier functions include:

- **REMOVEFILTERS** – Remove filters from all tables, a single table or column(s)
- **ALL** – Remove filters from a single table or column(s)
- **ALLEXCEPT** – Remove filters from all columns of a single table, except those explicitly passed in
- **ALLNONBLANKROW** – From the parent table of a relationship, returns all rows but the blank row, or all distinct values of a column but the blank row, and disregards any context filters that might exist

CALCULATE Filter Modifier example

New filter added

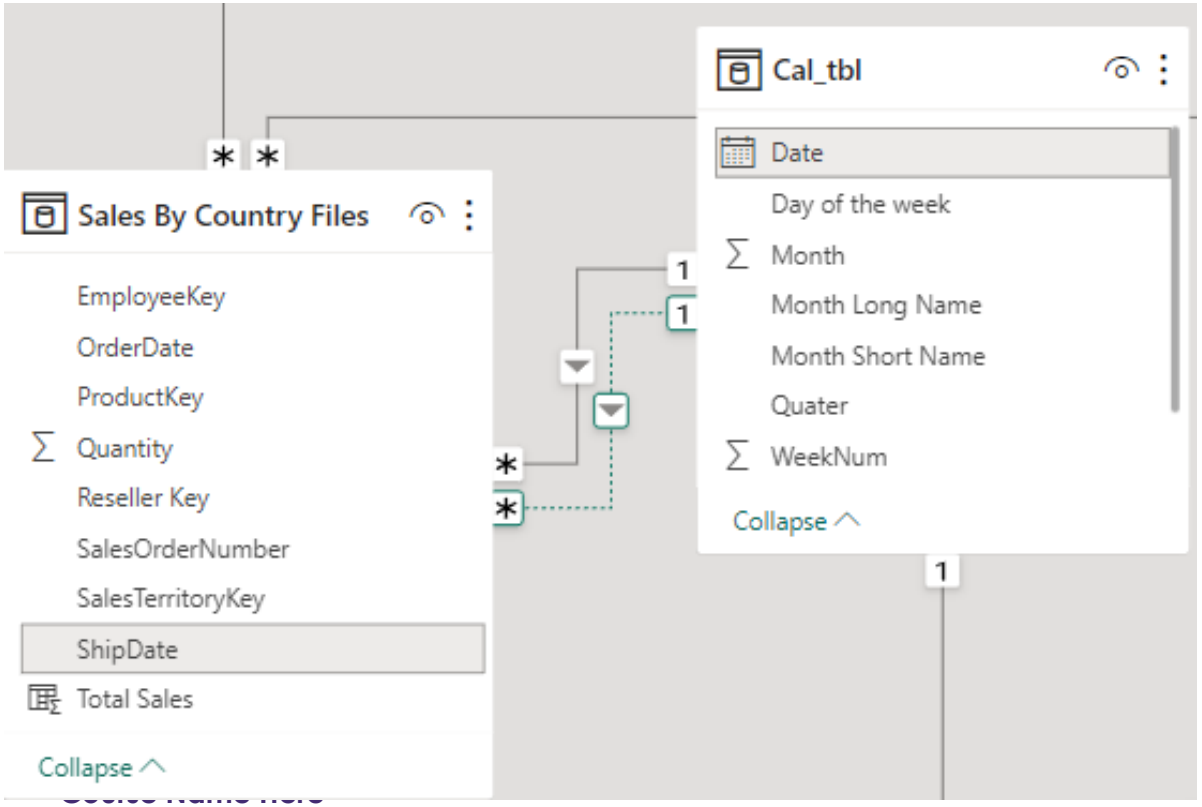
Region	Revenue	Revenue Red
Australia	$10,655,335.96	$2,681,324.79
Canada	$16,355,770.46	$3,573,412.99
Central	$7,909,009.01	$1,585,997.34
France	$7,251,555.65	$1,051,014.15
Germany	$4,878,300.38	$670,607.30
Northeast	$6,939,374.48	$1,876,016.33
Northwest	$16,084,942.55	$2,292,905.61
Southeast	$7,879,655.07	$1,457,221.07
Southwest	$24,184,609.60	$5,345,637.47
United Kingdom	$7,670,721.04	$1,063,753.75
Total	**$109,809,274.20**	**$21,597,890.81**

Existing filter overwritten

Color	Revenue	Revenue Red
Black	$38,236,124.06	$21,597,890.81
Blue	$9,602,850.97	$21,597,890.81
Grey		$21,597,890.81
Multi	$649,030.25	$21,597,890.81
NA	$1,099,303.91	$21,597,890.81
Red	$21,597,890.81	$21,597,890.81
Silver	$19,777,339.95	$21,597,890.81
Silver/Black	$147,483.91	$21,597,890.81
White	$29,745.13	$21,597,890.81
Yellow	$18,669,505.22	$21,597,890.81
Total	**$109,809,274.20**	**$21,597,890.81**

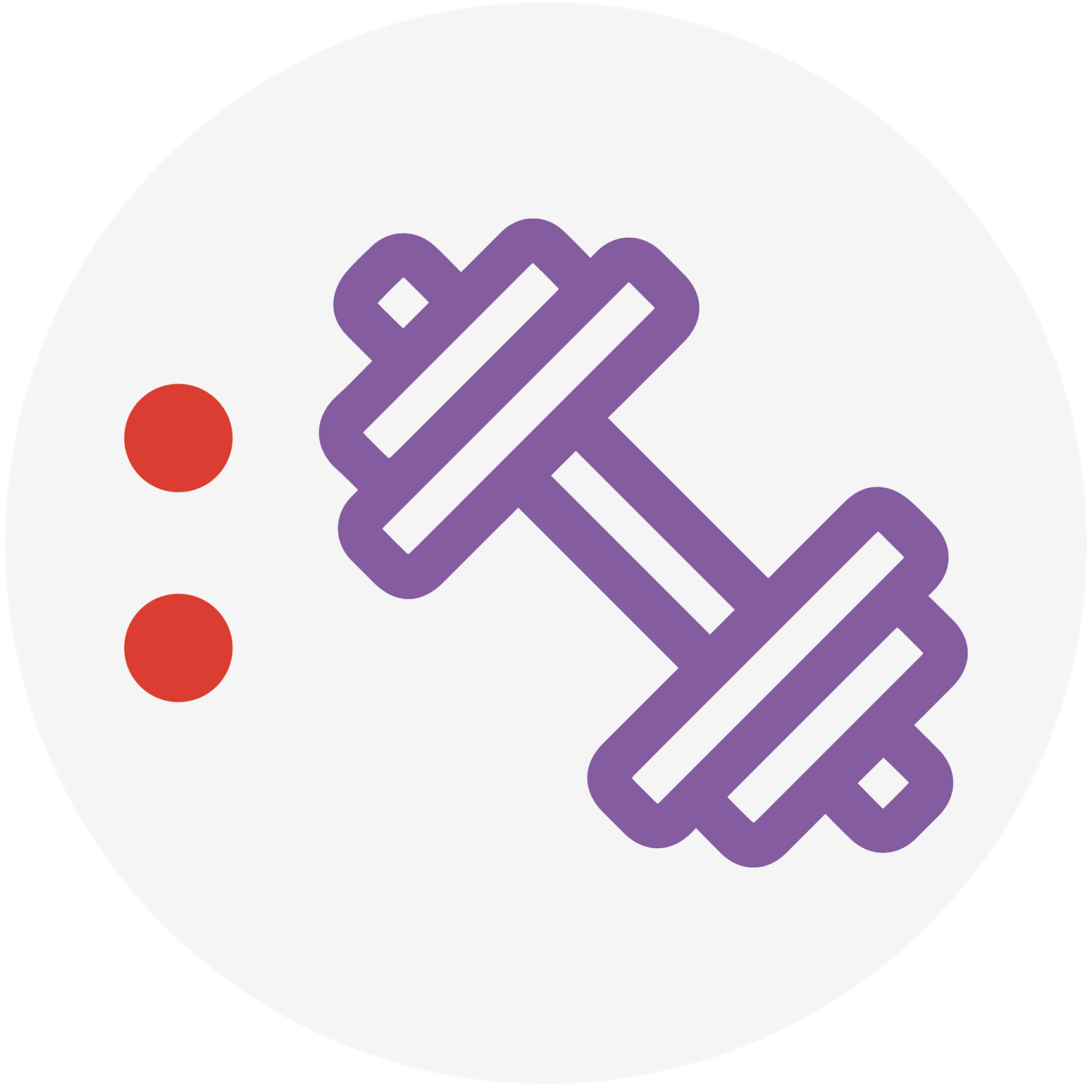
Relationship Modifiers

Use the USERRELATIONSHIP function to make an inactive relationship active during the evaluation of the CALCULATE function

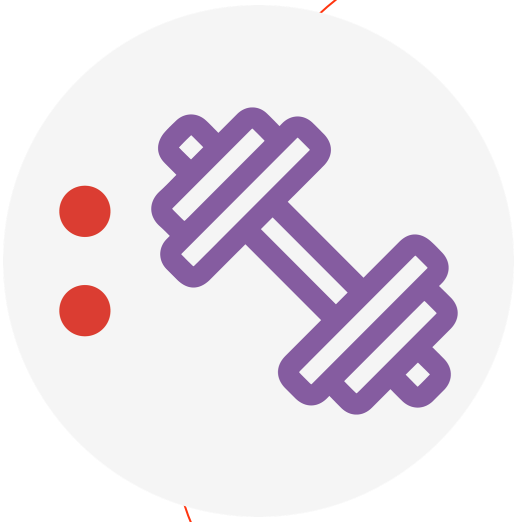



```
Revenue Shipped =
CALCULATE(
        [Revenue],
        USERELATIONSHIP(
                'Date'[DateKey],
                'Sales'[ShipDateKey]
        )
)
```

Live Demo



Exercise 3



KC5

Lesson 6
Time Intelligence DAX Patterns

After completing this module, students will be able to:

- Define time intelligence
- Use time intelligence functions
- Understand common time intelligence patterns

Define Time Intelligence

Time Intelligence relates to calculations over dates, months, quarters, or years

- It involves modifying the filter context for date filters
- The model requires a date table
- The date table can be created using the CALENDAR or CALENDARAUTO functions

Define Time Intelligence

Example

Month	Revenue	Revenue YTD
2019 Jan	$889,902.00	$889,902.00
2019 Feb	$837,304.45	$1,727,206.45
2019 Mar	$900,089.70	$2,627,296.15
2019 Apr	$6,366,809.65	$8,994,105.80
2019 May	$5,140,577.65	$14,134,683.45
2019 Jun	$3,801,978.60	$17,936,662.05
2019 Jul	$3,224,809.45	$21,161,471.50
2019 Aug	$2,705,752.60	$23,867,224.10
2019 Sep	$2,732,810.85	$26,600,034.95
2019 Oct	$6,104,340.55	$32,704,375.50
2019 Nov	$5,294,826.15	$37,999,201.65
2019 Dec	$4,271,109.60	$42,270,311.25
Total	**$42,270,311.25**	**$42,270,311.25**

Use Time Intelligence functions

DAX has many inbuilt Time Intelligence functions

- For example, TOTALYTD and PREVIOUSYEAR
- It is possible to replace them using CALCUALTE
- However, that requires more work and could be slower for Power BI to evaluate

Common Time Intelligence Patterns

- Cumulative totals
- Period comparisons
- New customer orders by month
- Stock on-hand value

Summarization over time

- DATESYTD – Returns a single-column table of dates
- TOTALYTD – Evaluates expression for year-to-date (YTD)
- DATESBETWEEN – Returns single-column table with range dates
- DATESINPERIOD – Returns single-column table with range of dates

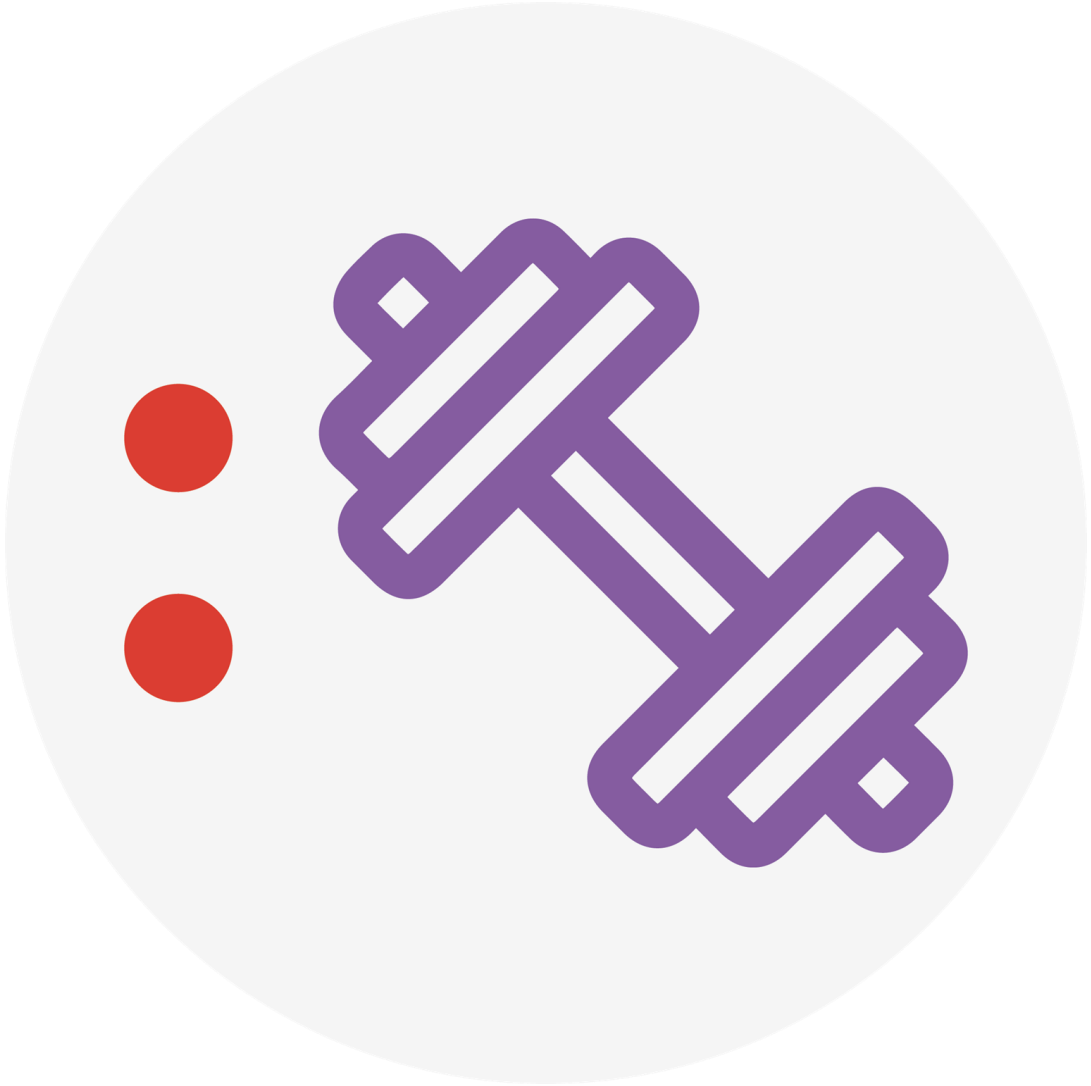
Shifting time periods

- DATEADD – Returns single-column table of dates shifted by interval
- PARALLELPERIOD – Returns single-column table of parallel dates
- SAMEPERIODLASTYEAR – Returns dates shifted back one year
- Many others – NEXTDAY, NEXTMONTH, NEXTQUARTER, NEXTYEAR, PREVDAY, PREVMONTH, PREVQUARTER ,PREVYEAR

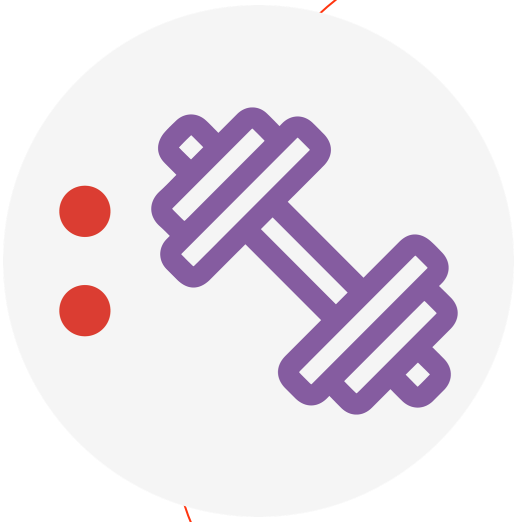
Shifting time periods

- DATEADD – Returns single-column table of dates shifted by interval
- PARALLELPERIOD – Returns single-column table of parallel dates
- SAMEPERIODLASTYEAR – Returns dates shifted back one year
- Many others – NEXTDAY, NEXTMONTH, NEXTQUARTER, NEXTYEAR, PREVDAY, PREVMONTH, PREVQUARTER ,PREVYEAR

Live Demo



Exercise 4



KC6

DATASOCIETY:

Power BI Intermediate

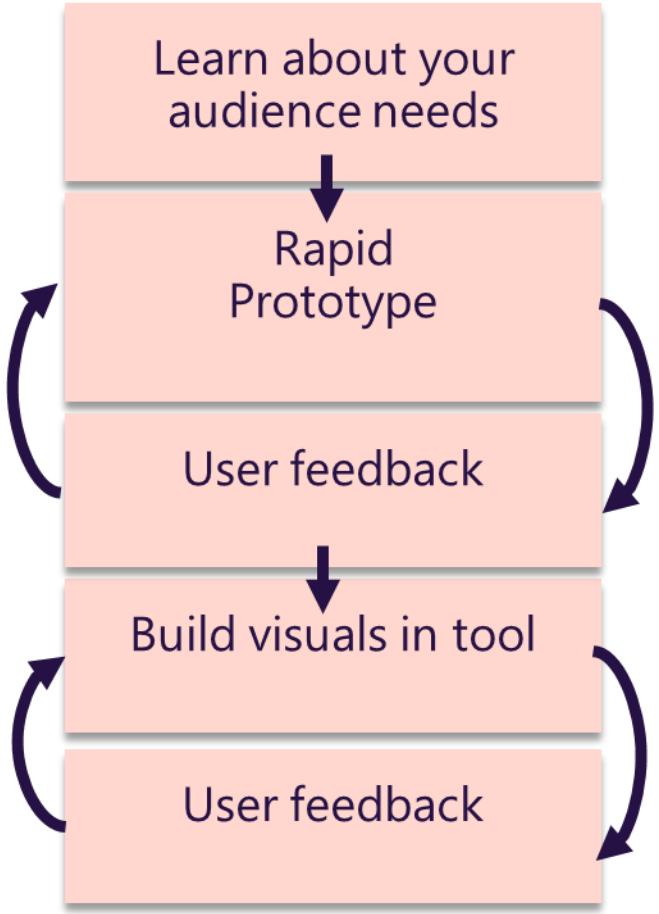
Day 4

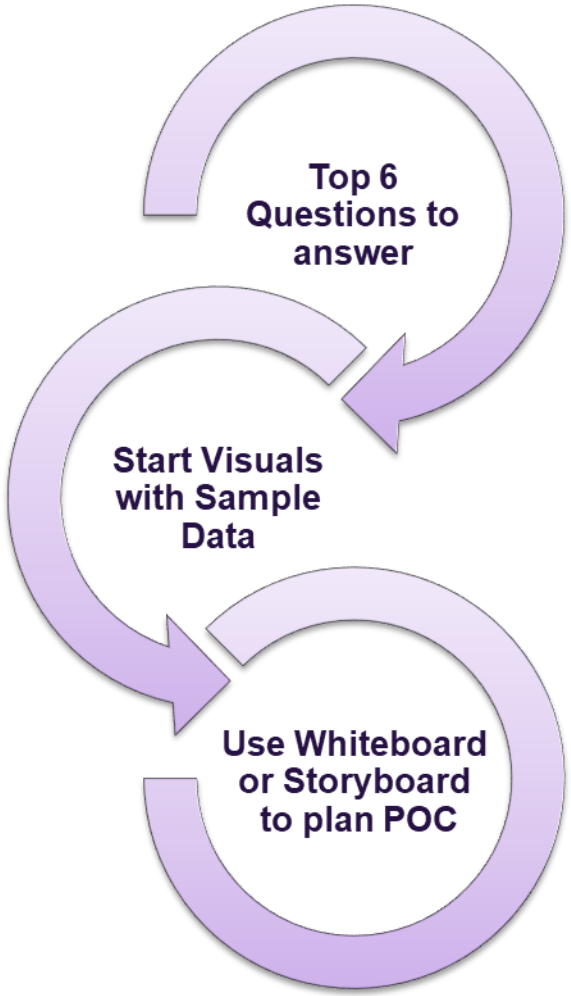
Lesson 7
Advanced Report Design Principles

After completing this module, students will be able to:

- Understand the importance of data storytelling
- Plan report layout, user interface and flow
- Utilize best practices for custom report navigation, report tooltips and drill-through options

Agile Report Creation





Who? What? When? Where? How Many? Why?

Convert the Story to a Date Model

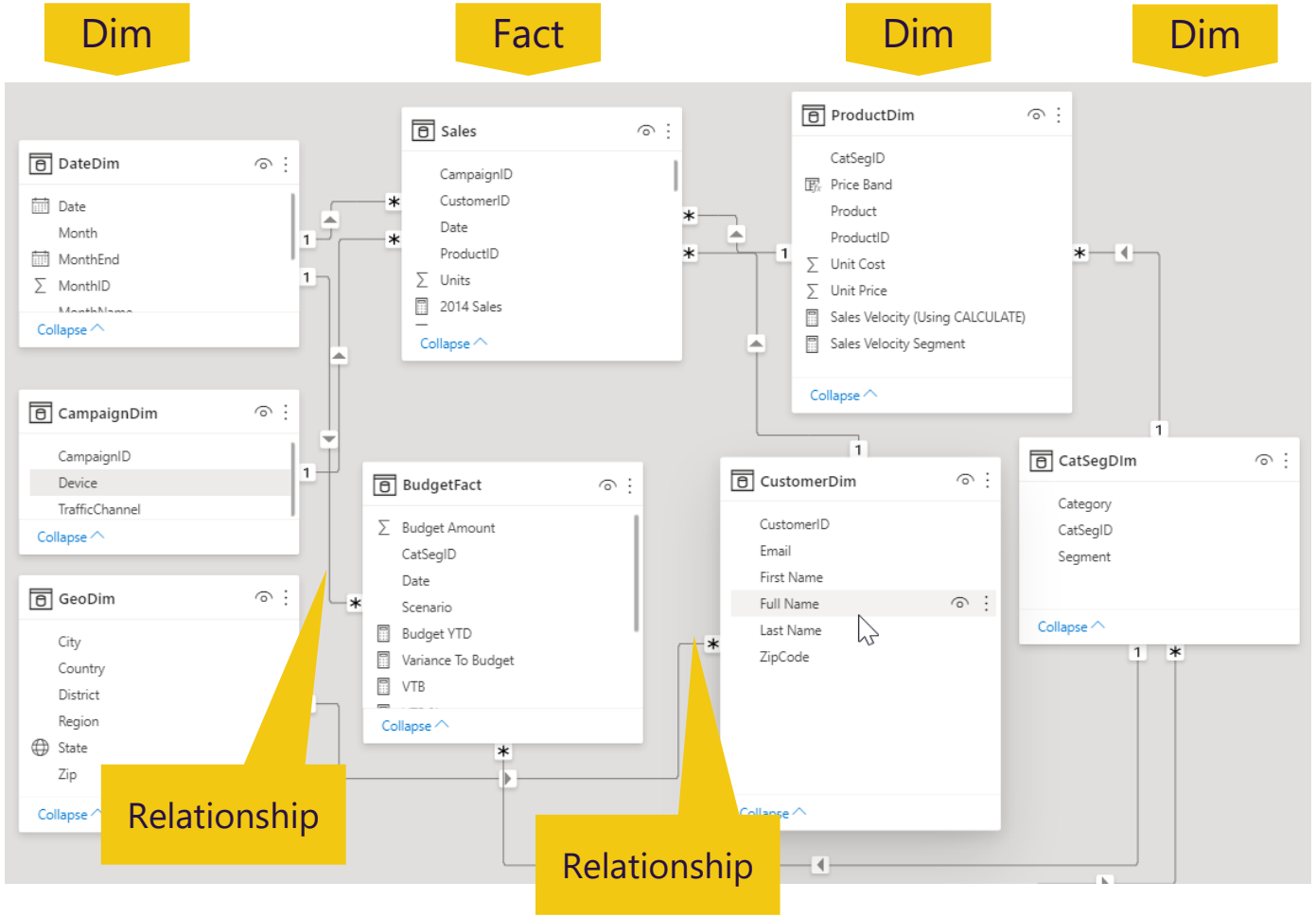
List your big questions:

- What is my Total Sales for a Selected Year and Region?
- How is my Total Sales doing Year Over Year?
- How are my Units trending for various States in my region?
- How is my Sales doing by Channel, Device, Category for selected Year?
- Which categories are performing best to worst by Total Sales ?

- Units
- Total Sales
- Gross Profit

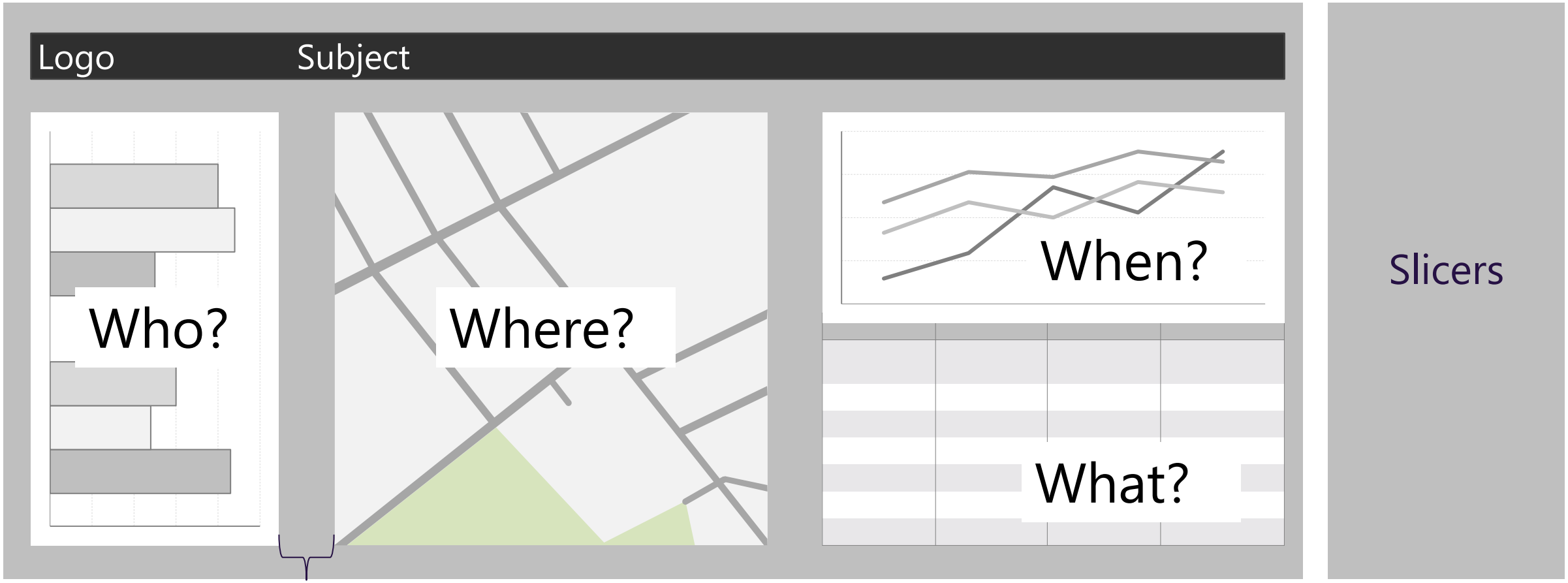
- By Time (Year, Month)
- By Geography (Region State or City)
- By Campaign (Channel or Device)

Convert Story to a Data Model



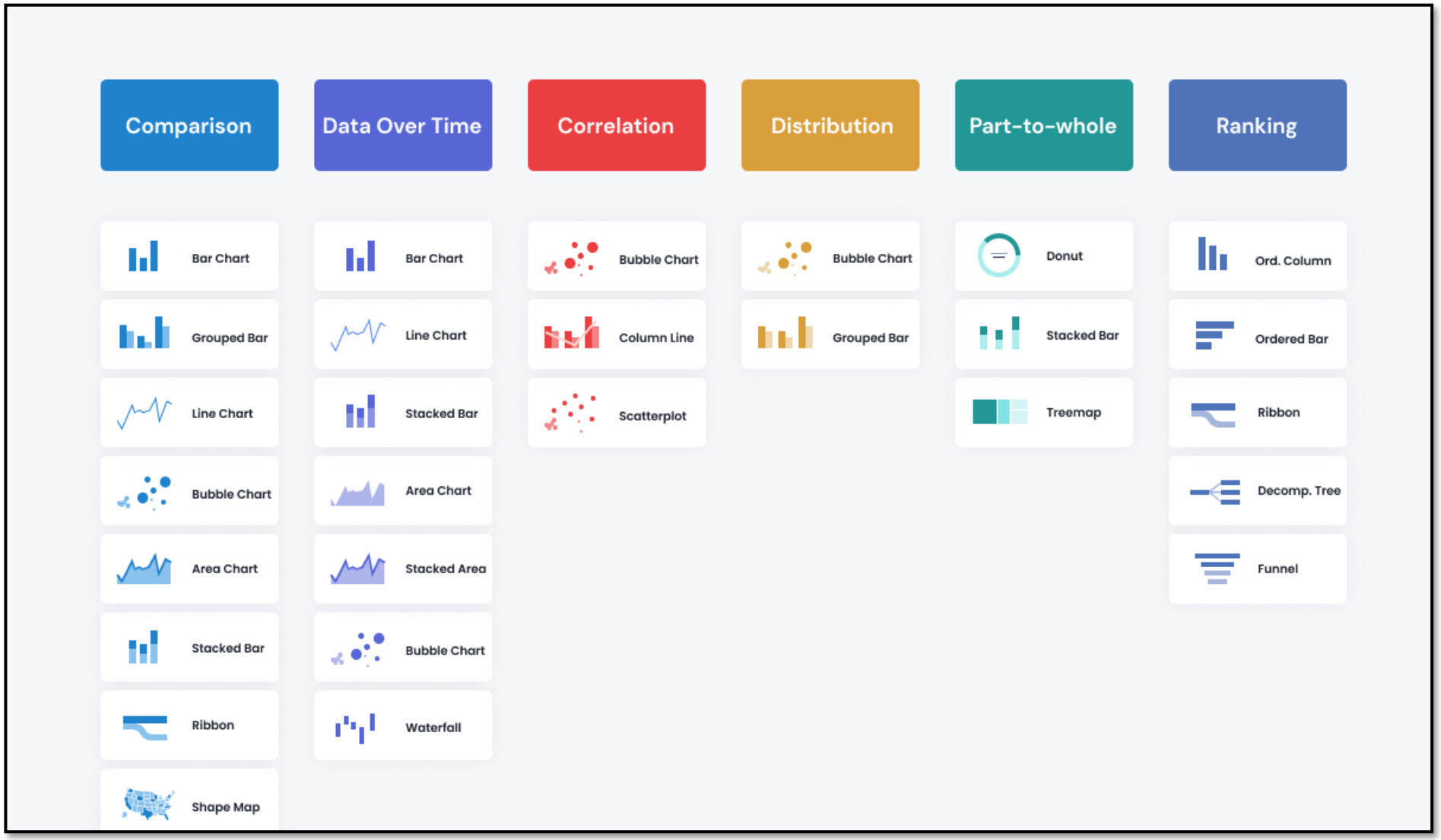
- Measures (e.g. Units or Sales) live in **fact** tables
- Descriptive **attributes** (e.g. Campaign, Customer name) live in **dimension** tables
- **Relationships** tie the data together so you can slice your measures by your attributes

Use Storyboarding to Design Report Layout

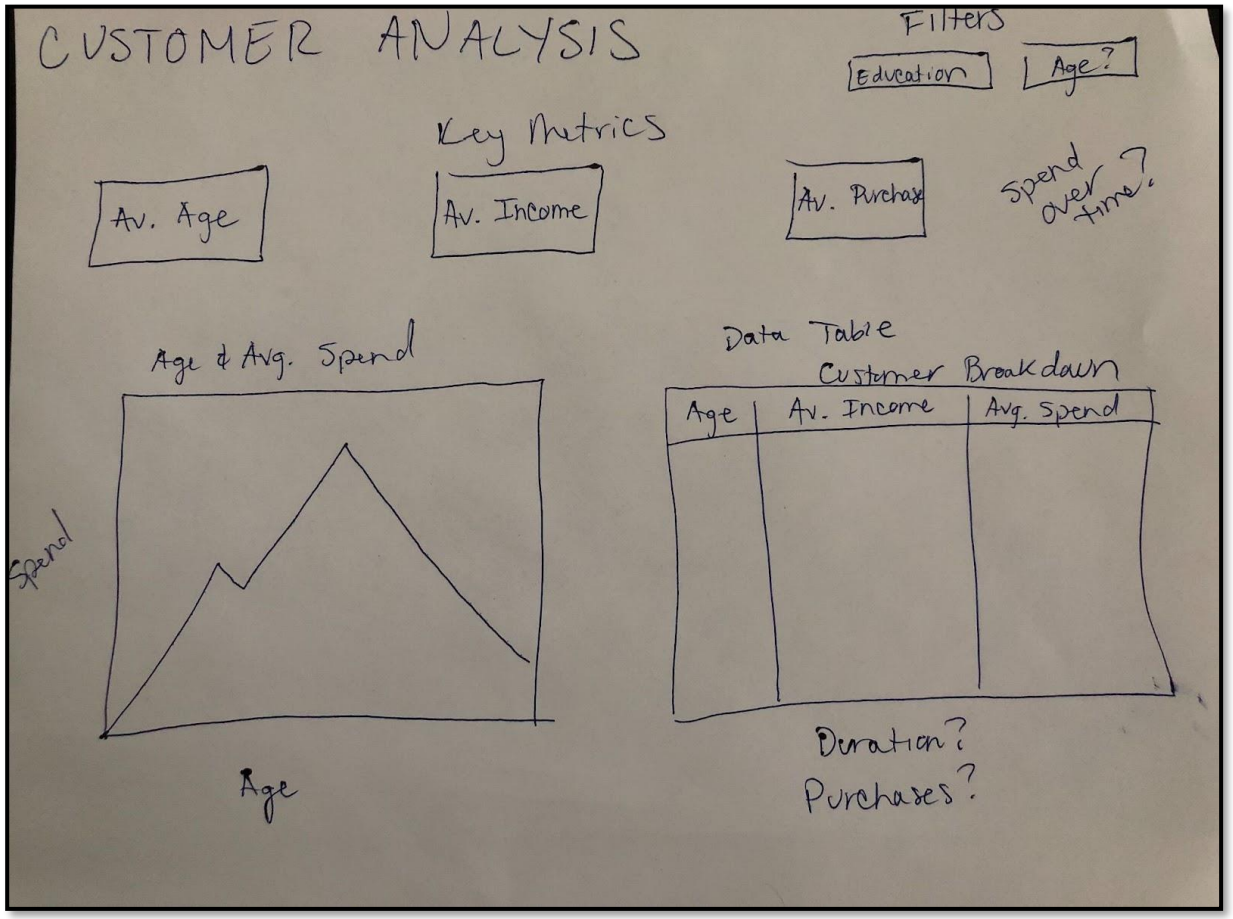


Control this space. If all objects tell the same story, use the same gap space. If some objects tell a different story, make the gap bigger

Visual Selection



Whiteboard Mockup

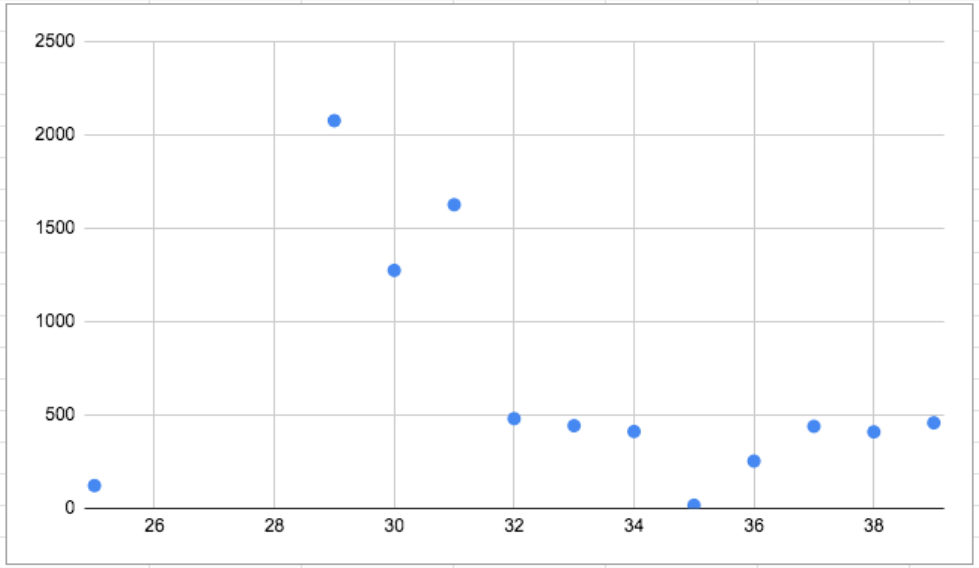


Wireframe

Customer Analysis

Av. Age	Av. Income	Av. Purchases
52.25	**$52,224.9**	**12.73**

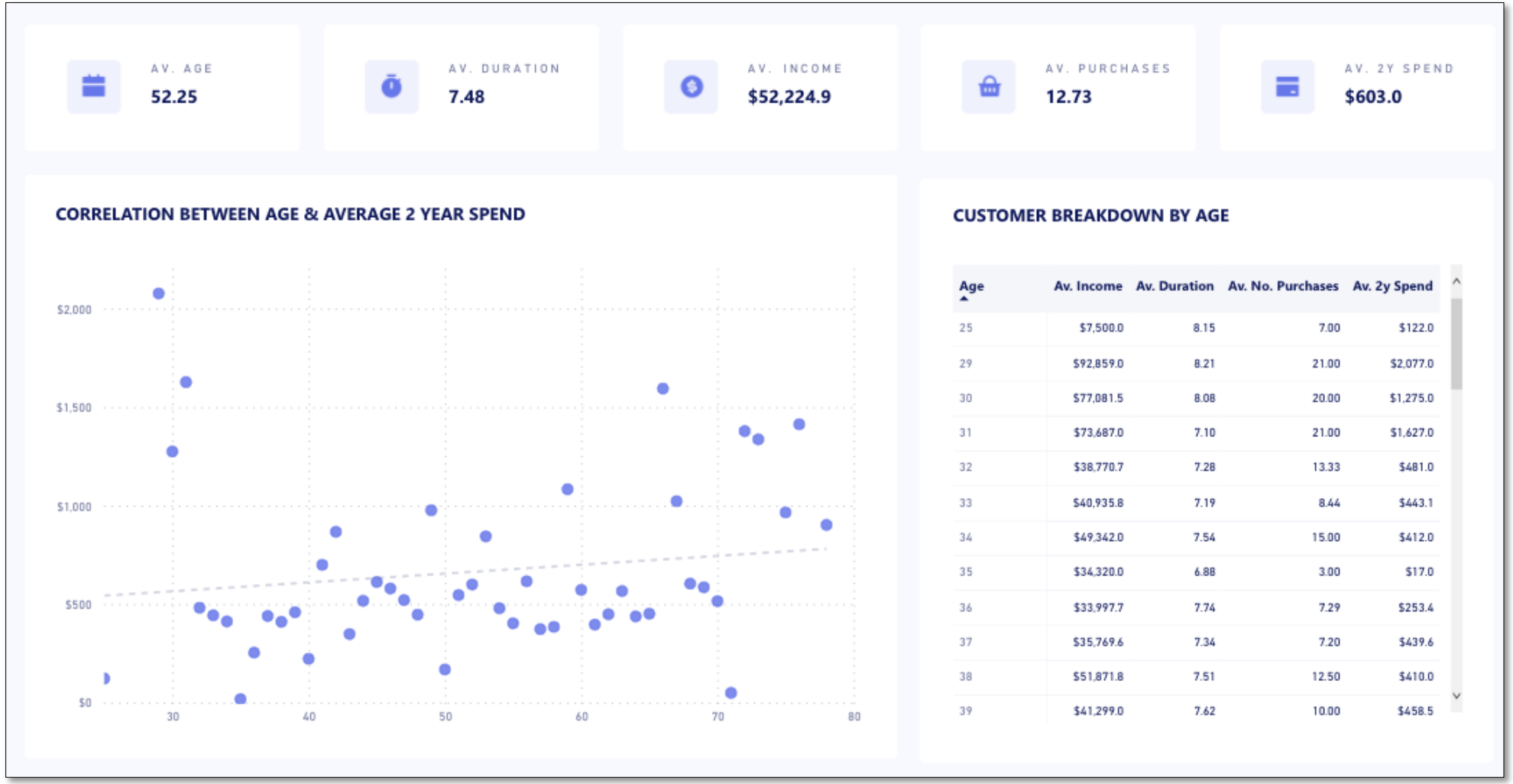
Age & Av. Spend



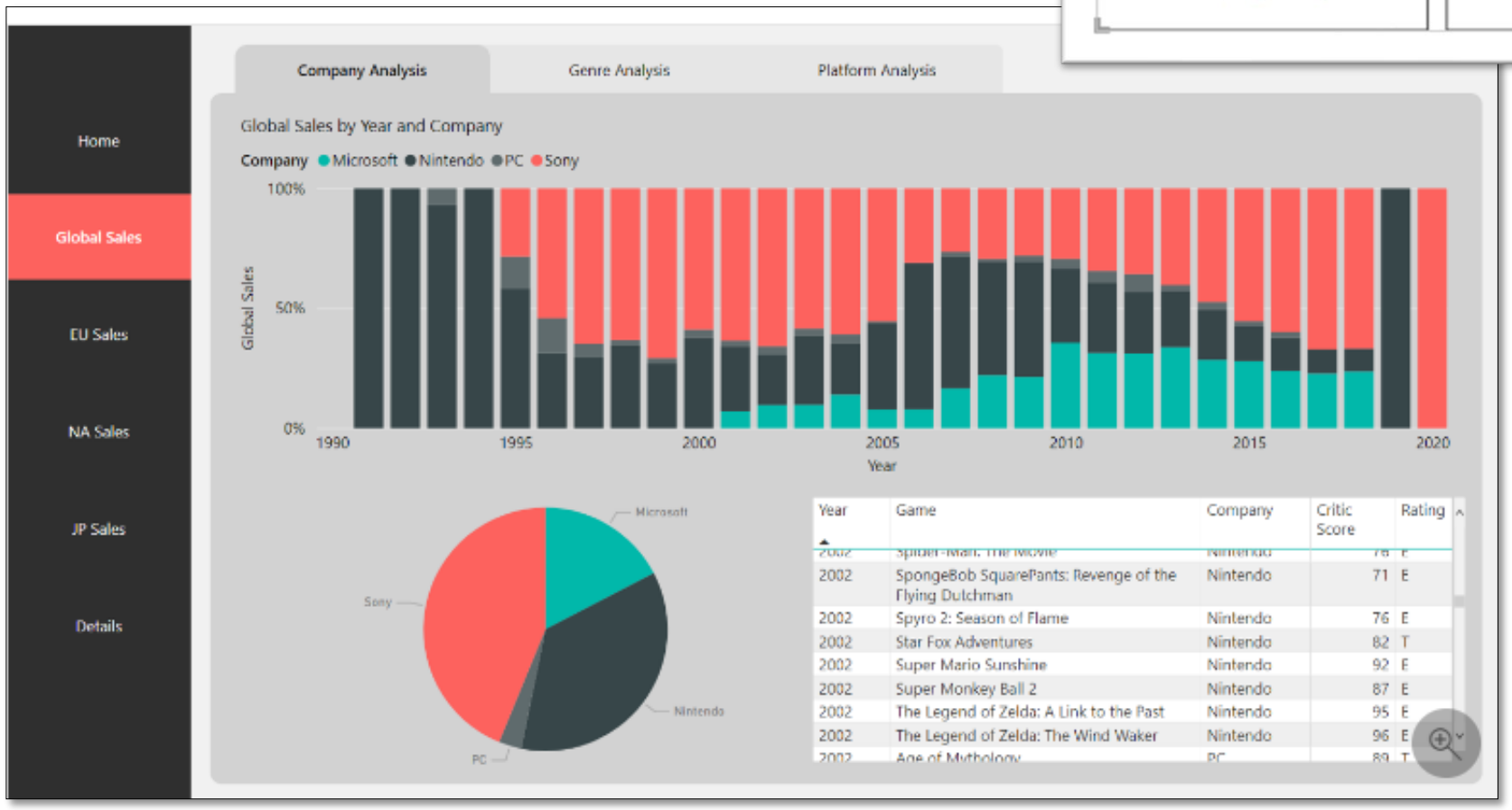
Customer Breakdown

Age	Av 2y Spend
25	122
29	2077
30	1275
31	1627
32	481
33	443.1
34	412
35	17
36	253.4
37	439.6
38	410
39	458.5

Final Product



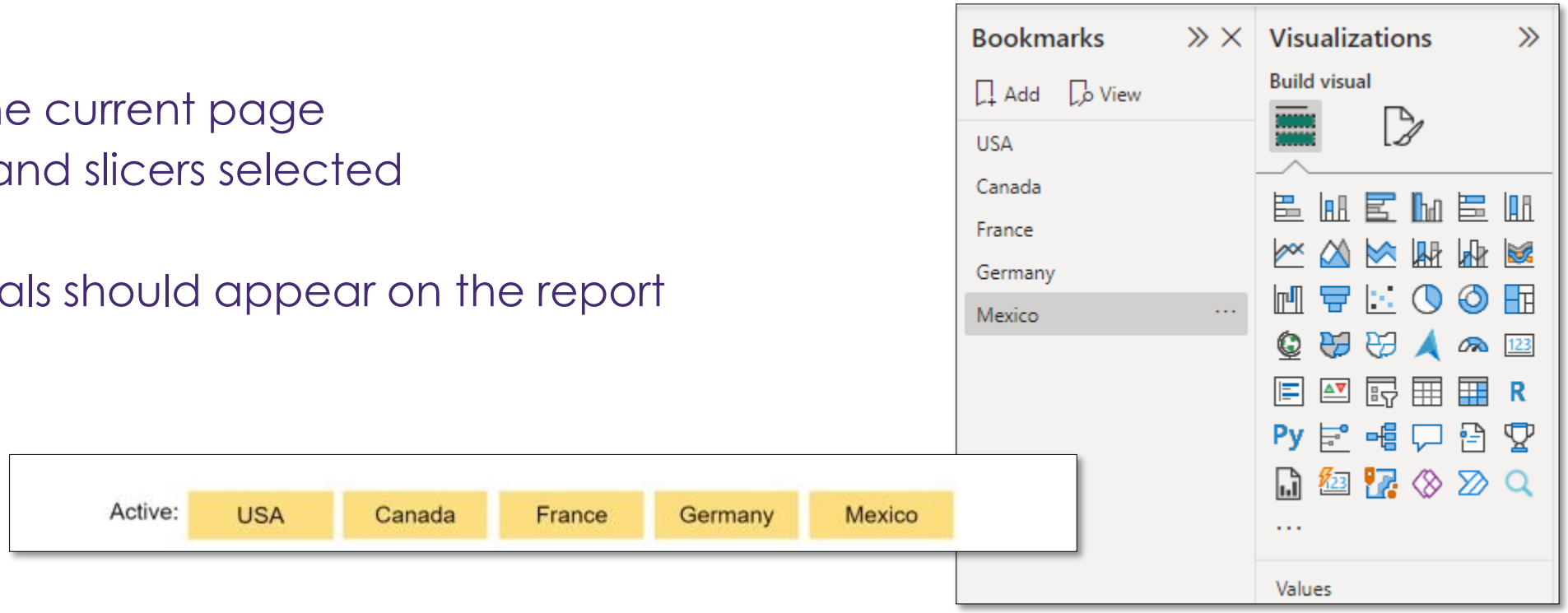
Page Navigation



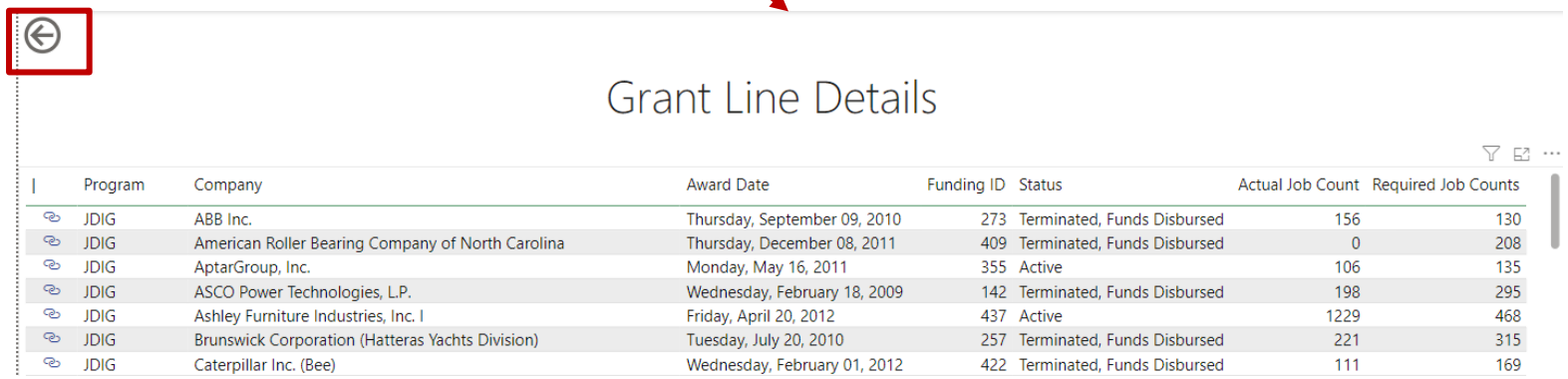
Bookmark

Bookmarks let you save interesting states as part of your report. Once you have a list of bookmarks, you can use these in several ways including organizing and transitioning visuals:

- Includes the current page
- Any filters and slicers selected
- Sort Order
- Which visuals should appear on the report

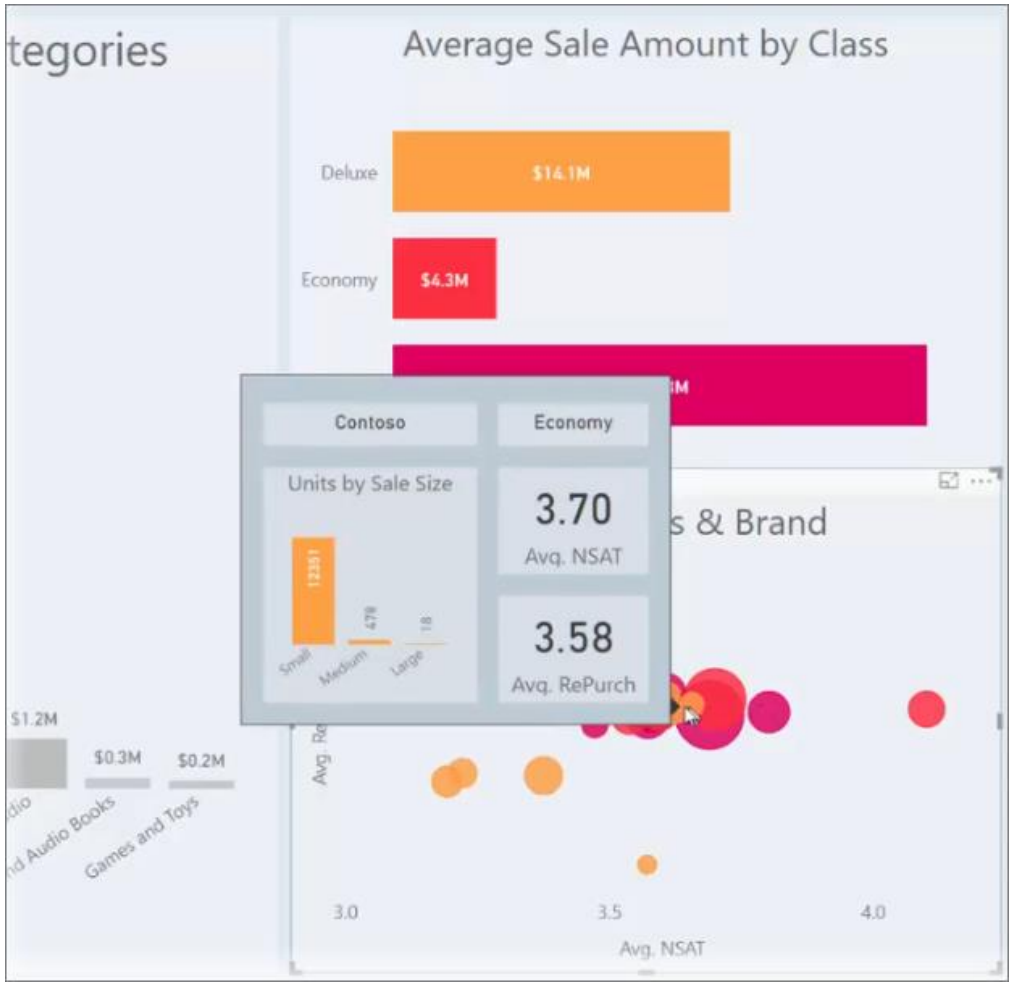


Report Drill-through



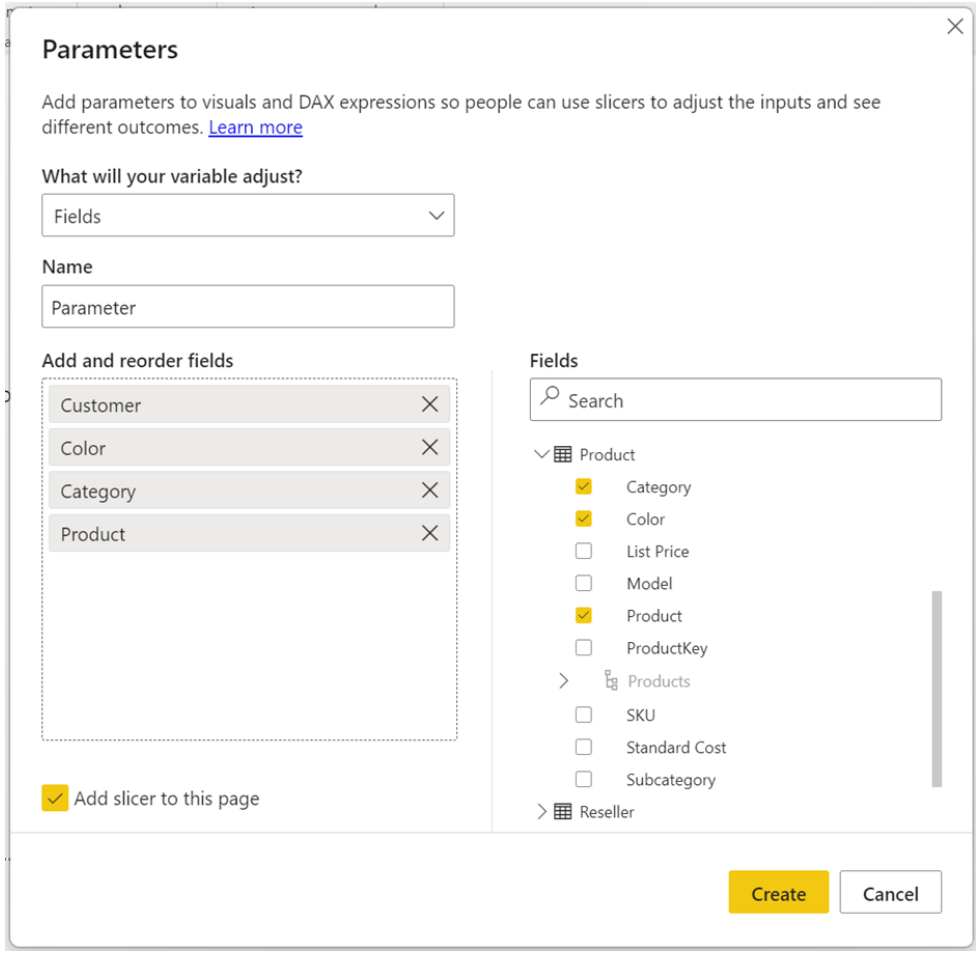
	Program	Company	Award Date	Funding ID	Status	Actual Job Count	Required Job Counts
	JDIG	ABB Inc.	Thursday, September 09, 2010	273	Terminated, Funds Disbursed	156	130
	JDIG	American Roller Bearing Company of North Carolina	Thursday, December 08, 2011	409	Terminated, Funds Disbursed	0	208
	JDIG	AptarGroup, Inc.	Monday, May 16, 2011	355	Active	106	135
	JDIG	ASCO Power Technologies, L.P.	Wednesday, February 18, 2009	142	Terminated, Funds Disbursed	198	295
	JDIG	Ashley Furniture Industries, Inc. I	Friday, April 20, 2012	437	Active	1229	468
	JDIG	Brunswick Corporation (Hatteras Yachts Division)	Tuesday, July 20, 2010	257	Terminated, Funds Disbursed	221	315
	JDIG	Caterpillar Inc. (Bee)	Wednesday, February 01, 2012	422	Terminated, Funds Disbursed	111	169

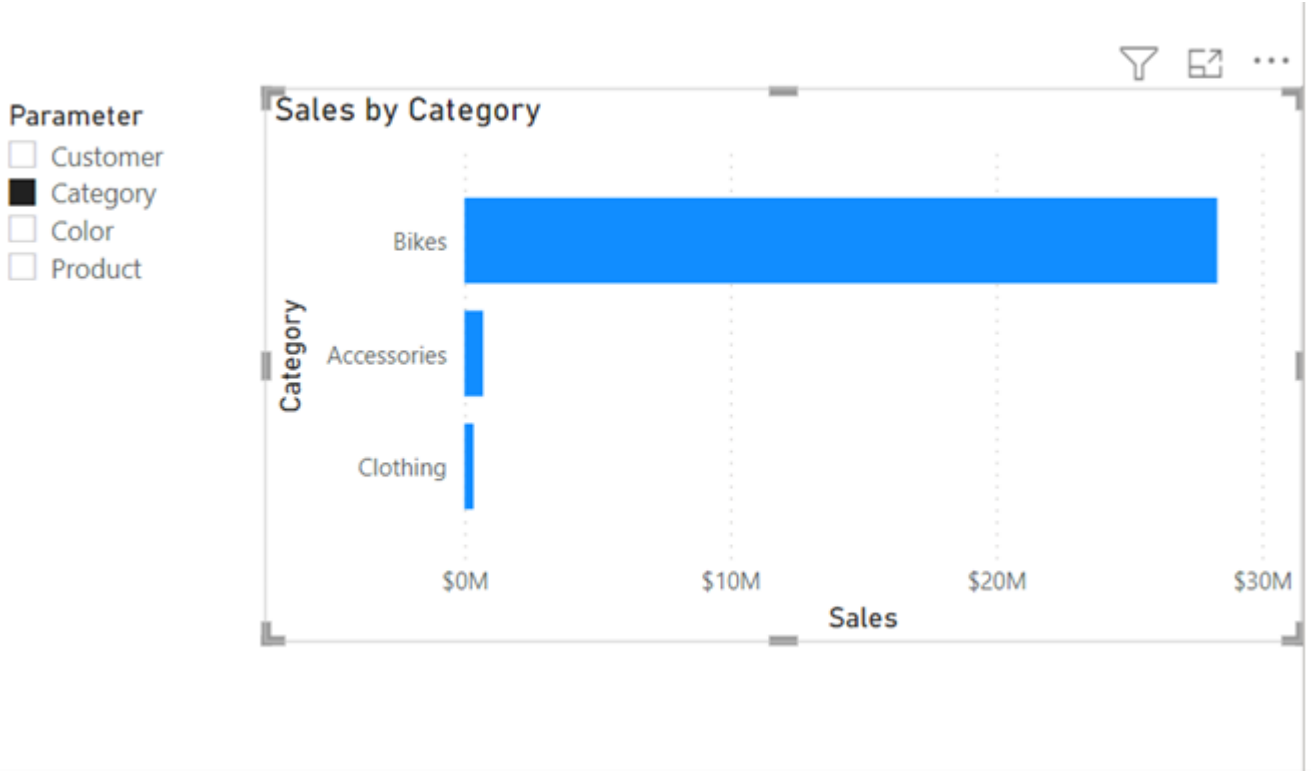
Custom Tooltip



Custom tooltips in Power BI let you create personalized and interactive data point descriptions that appear when users hover over visuals, enhancing the user experience with tailored information and insights.

Field Parameters

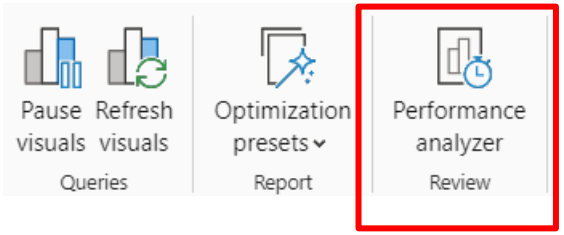


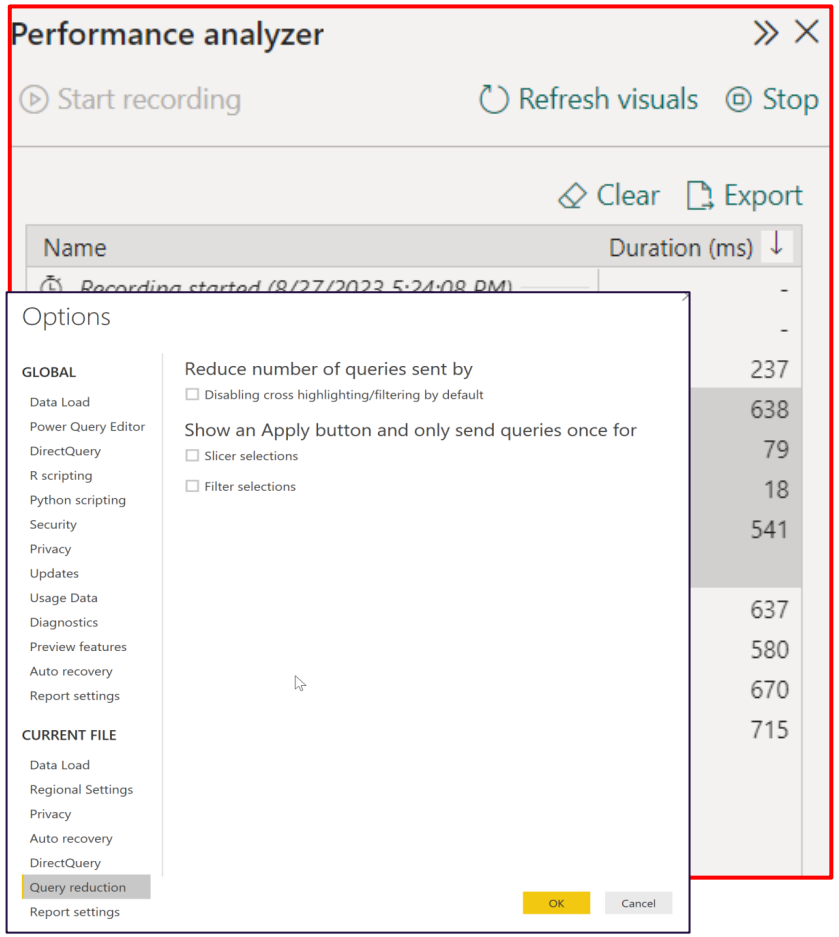


Performance Analyzer

Built-in performance analysis

- Each visual has three components
  - How long did the DAX query take?
  - How long did my visual take to render?
  - How long was everything else?





Use Query Reduction Features

Scenario

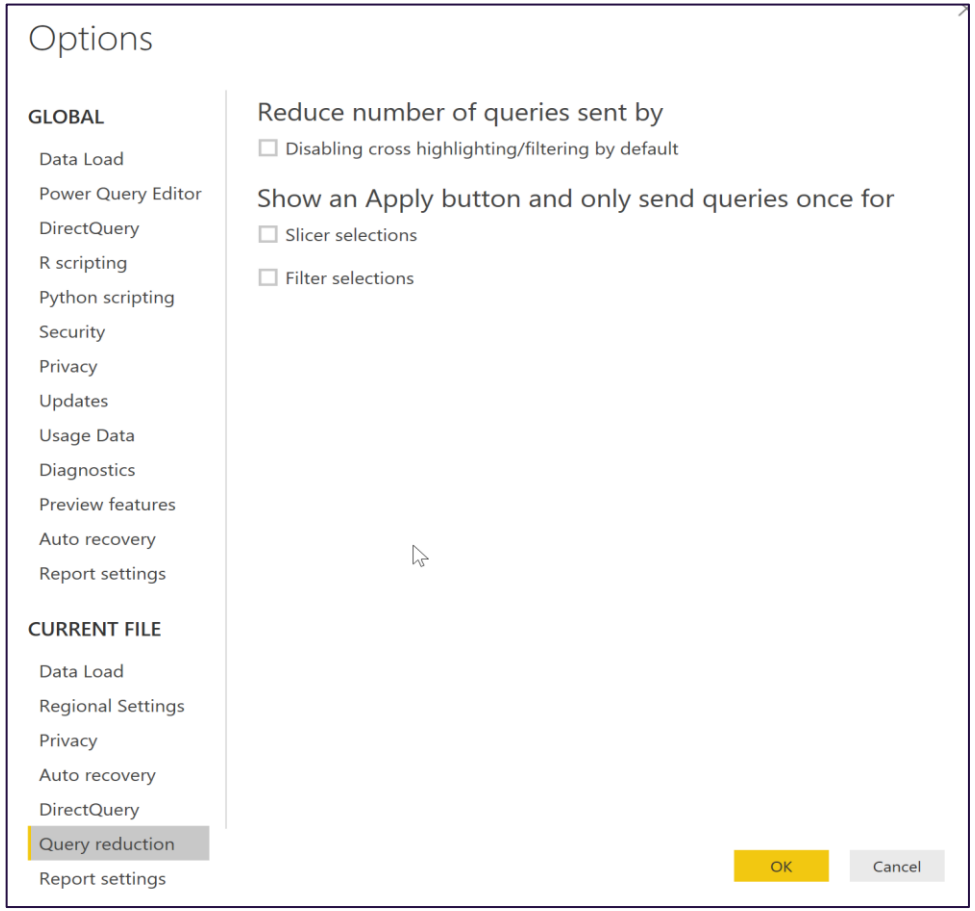
- Previously discussed high number of visuals/slicers

Why is it desired?

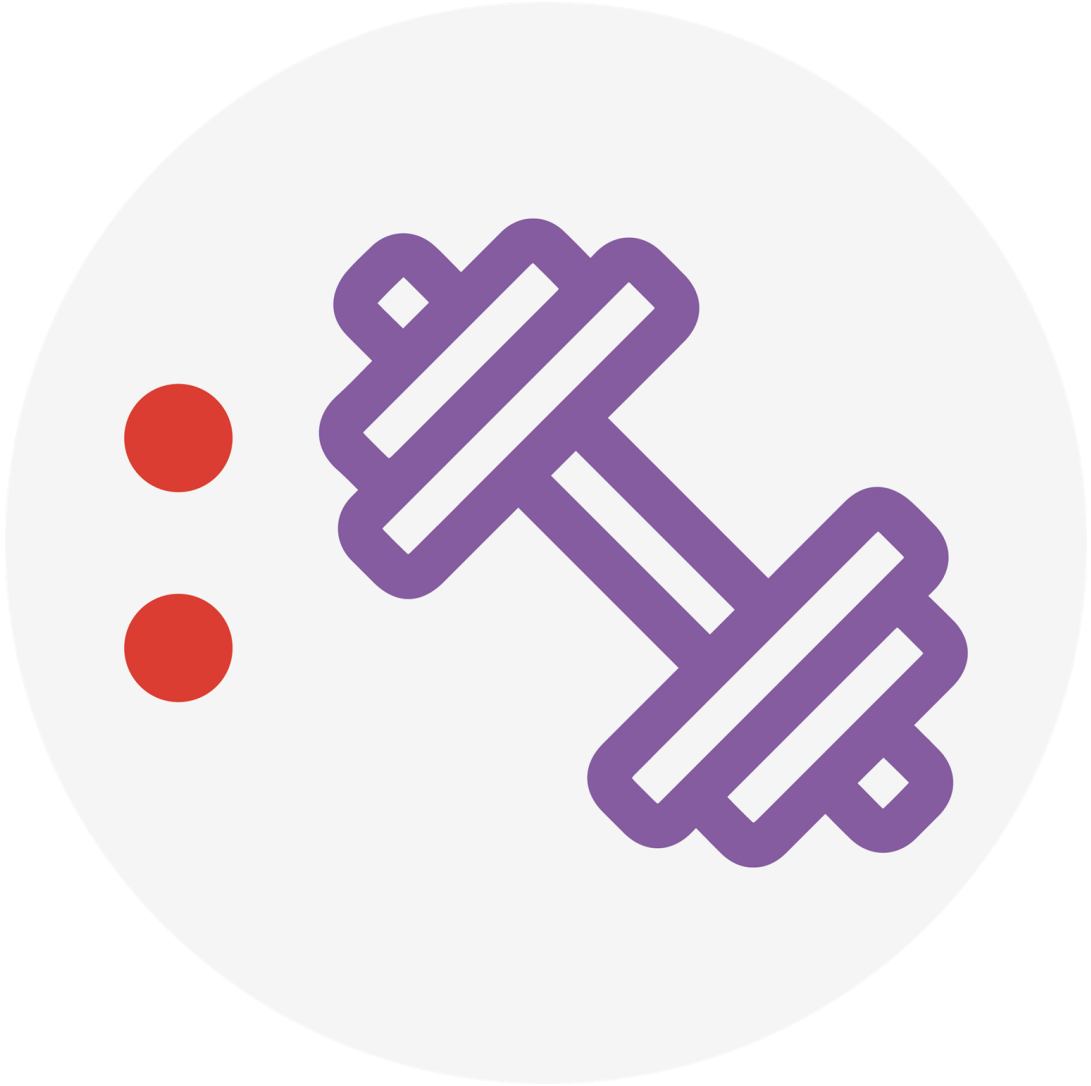
- Lower memory load, less data to fetch and process

Proposed solutions

- consider no cross-highlighting, and
- adding Apply buttons to slicers and filters



Live Demo



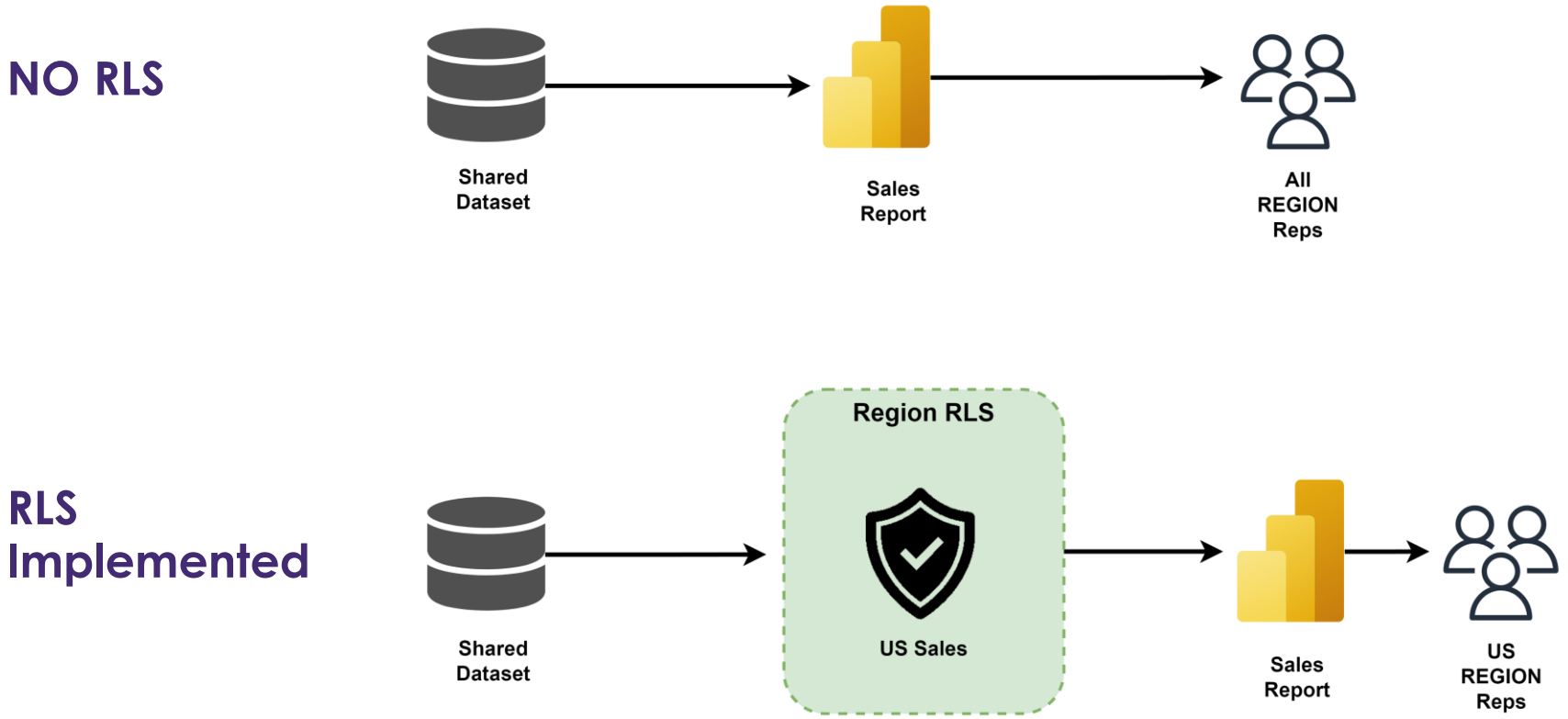
KC7

Lesson 8
Row-Level Security

After completing this module, students will be able to:

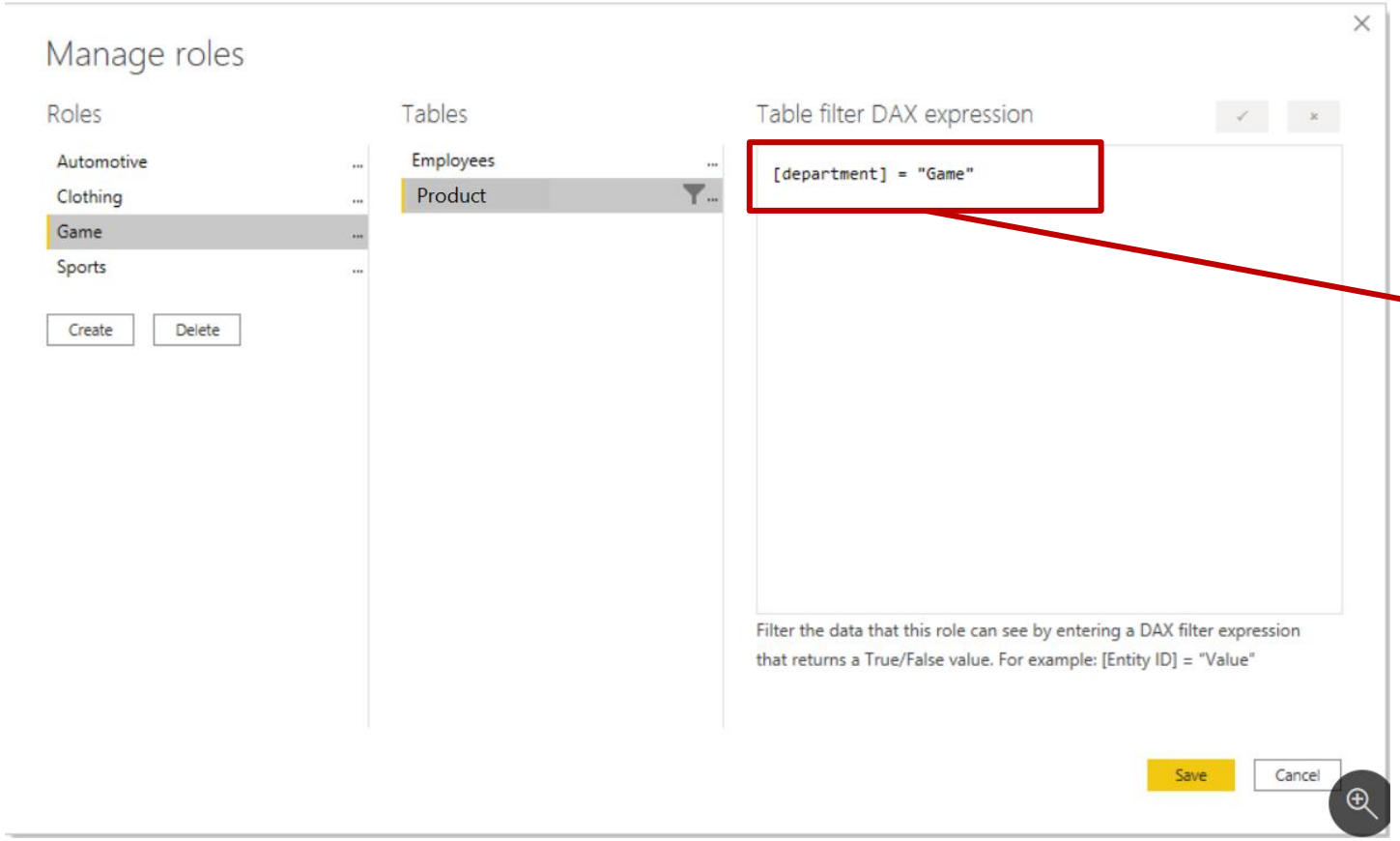
- Define RLS
- Implement Static RLS
- Implement Dynamic RLS

What is Row-Level Security



Row-Level Security (RLS) in Power BI is a security feature that allows you to control access to data at a granular level. With RLS, you can define rules that restrict which rows of data users can see in a dataset or report based on their roles.

Static Row-Level Security

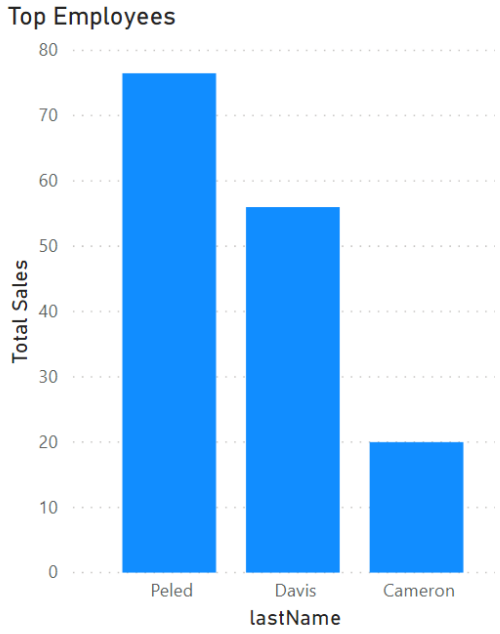


152.47

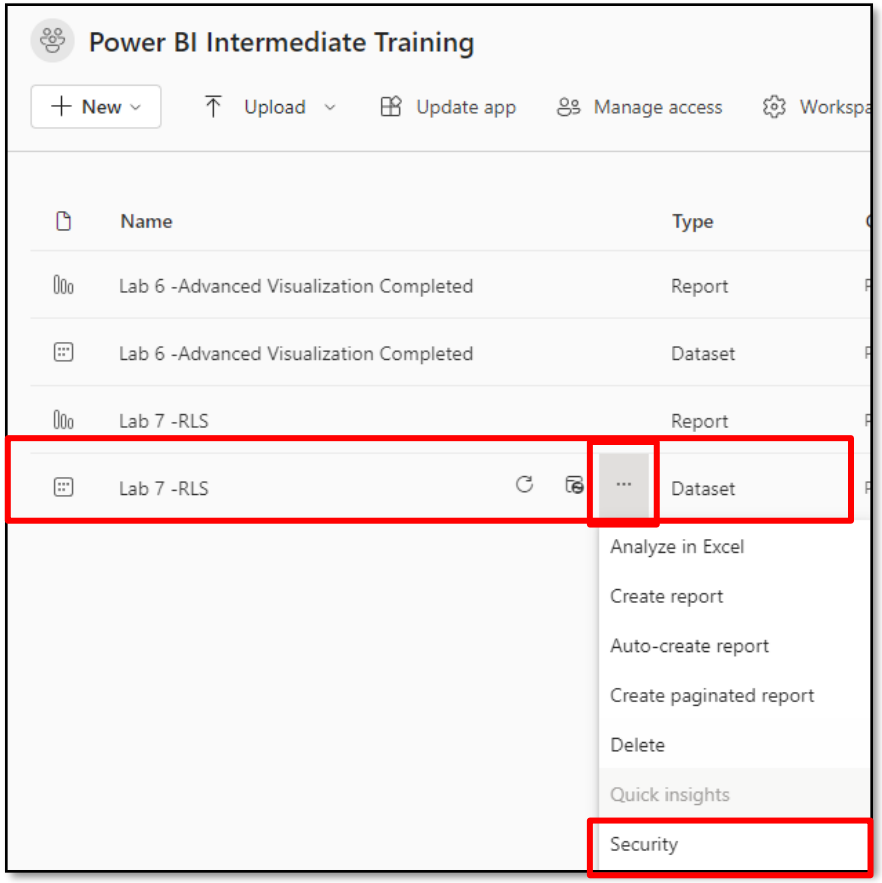
Total Sales

Top Products Sales

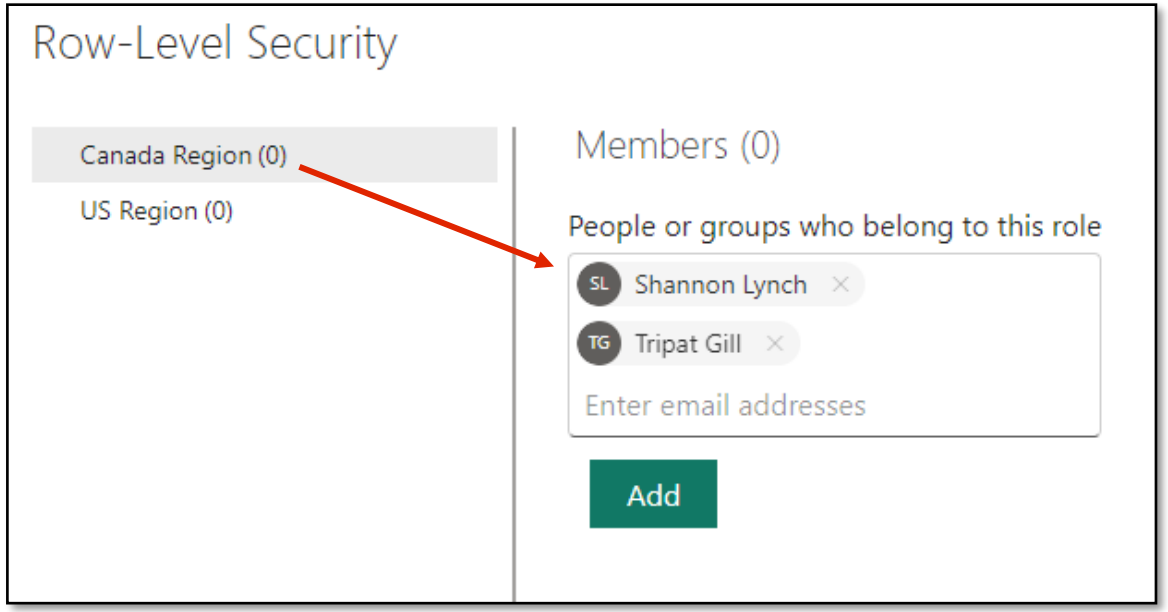
product	department	Total Sales
Spider, spider	Game	44.50
Invest in it All	Game	31.99
Santo Domingo	Game	31.00
Settlers of Air	Game	24.99
Lords of Avalon	Game	19.99
Total		**152.47**



Setting Up RLS in Power BI Service



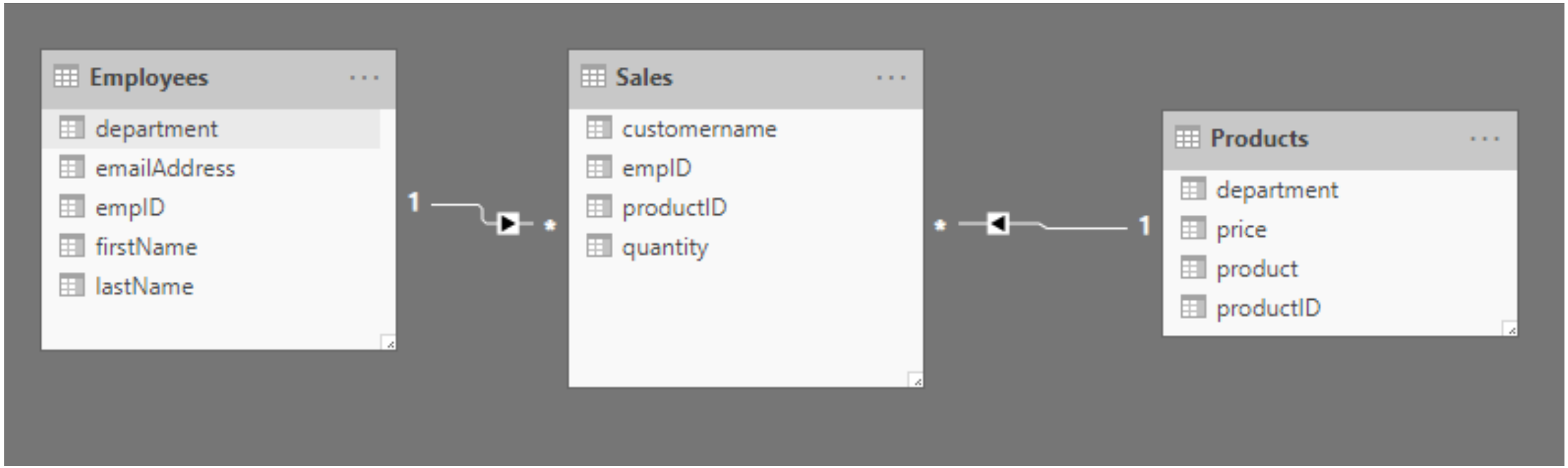
From your Power BI service workspace you will need to select the dataset security settings



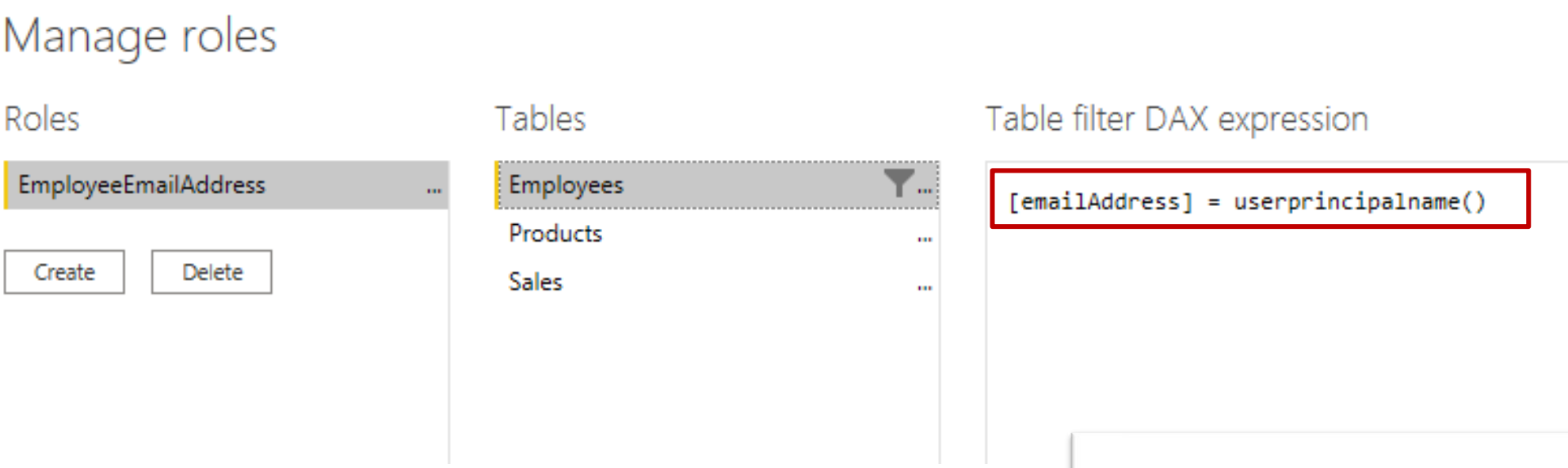
Add the users that should be assigned to the applicable RLS role

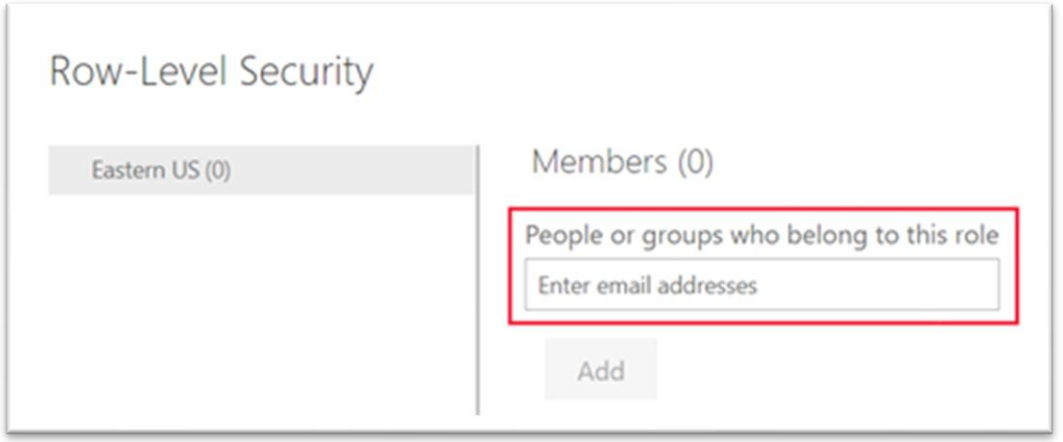
RLS Security

Secure reports and workspace by sharing them to Active Directory users and groups.

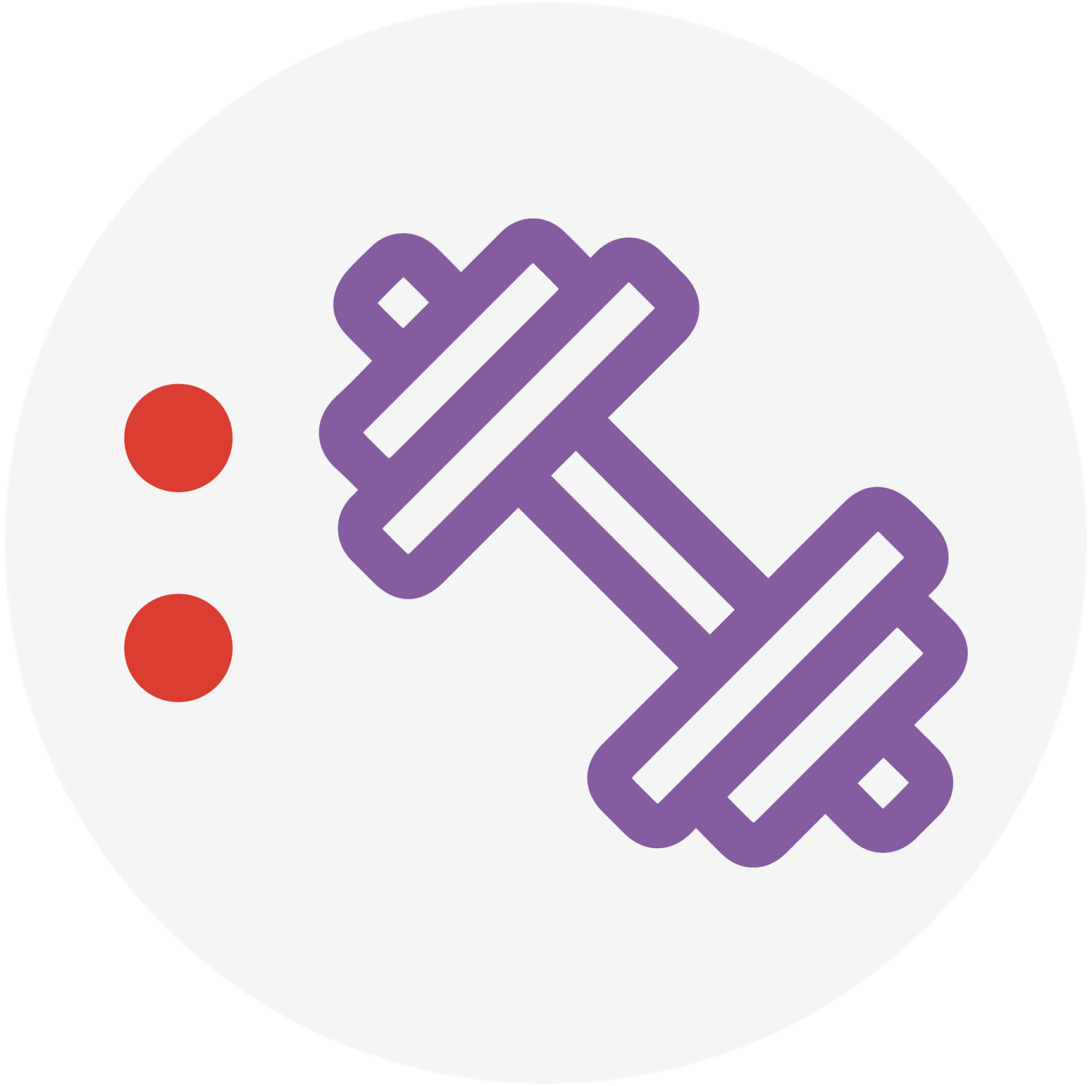


Dynamic RLS

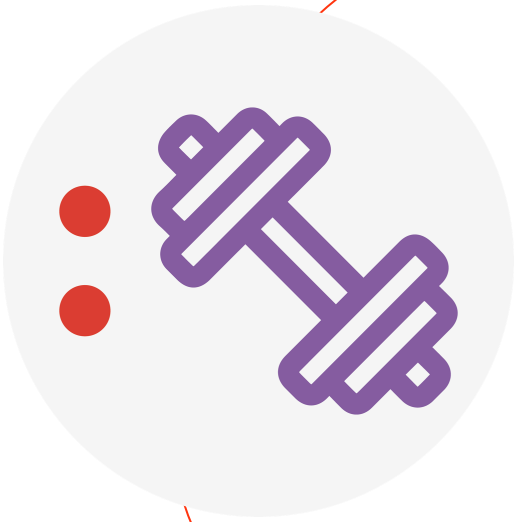

Manage roles
Roles
EmployeeEmailAddress
Create
Delete
Tables
Employees
Products
Sales
Table filter DAX expression
[emailAddress] = userprincipalname()


Row-Level Security
Eastern US (0)
Members (0)
People or groups who belong to this role
Enter email addresses
Add

Live Demo



Exercise 5



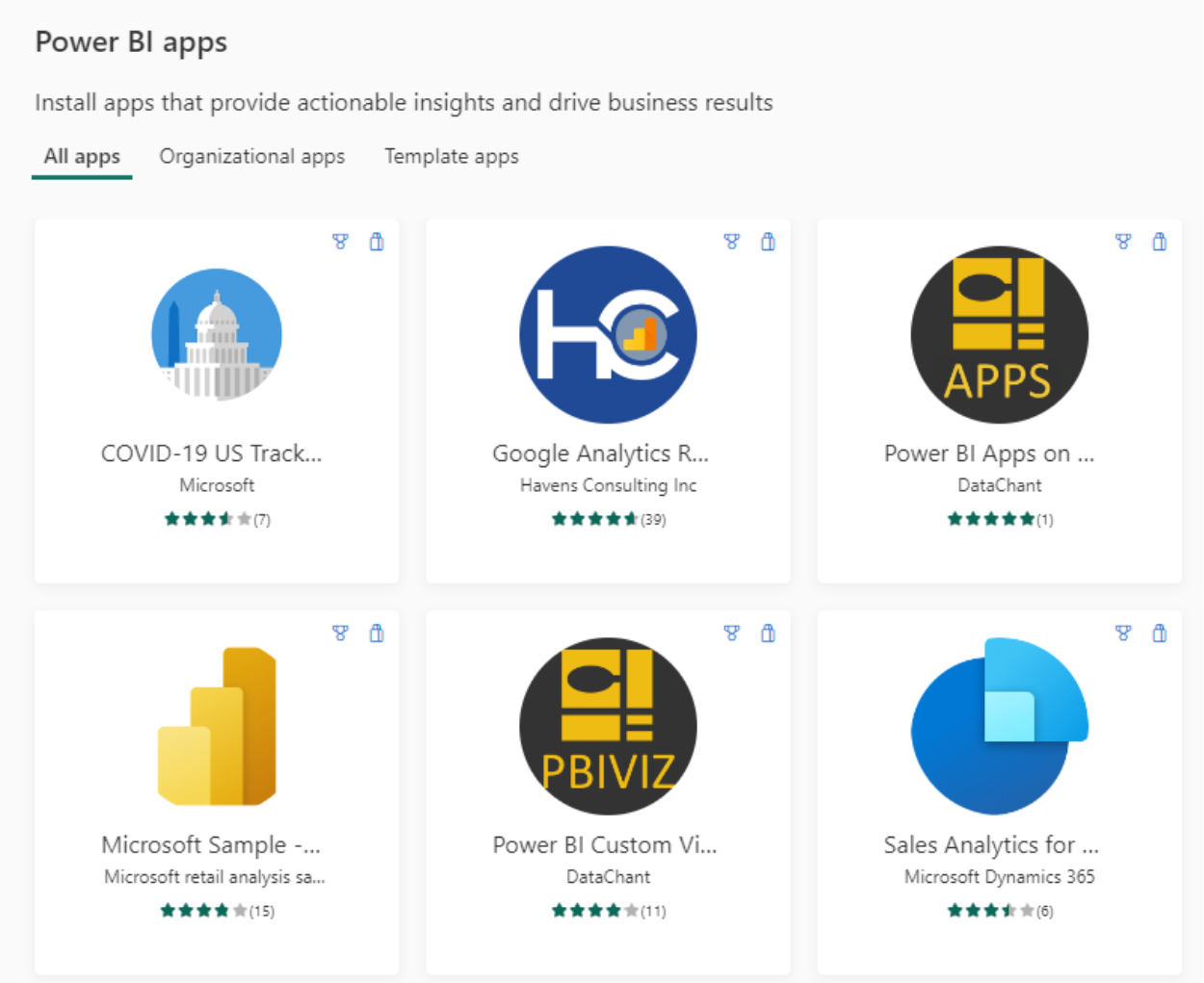
KC8

Power BI App Deployment

Publishing your report App

Within Power BI, you have the capability to craft formalized content packages and subsequently share them widely in the form of apps.

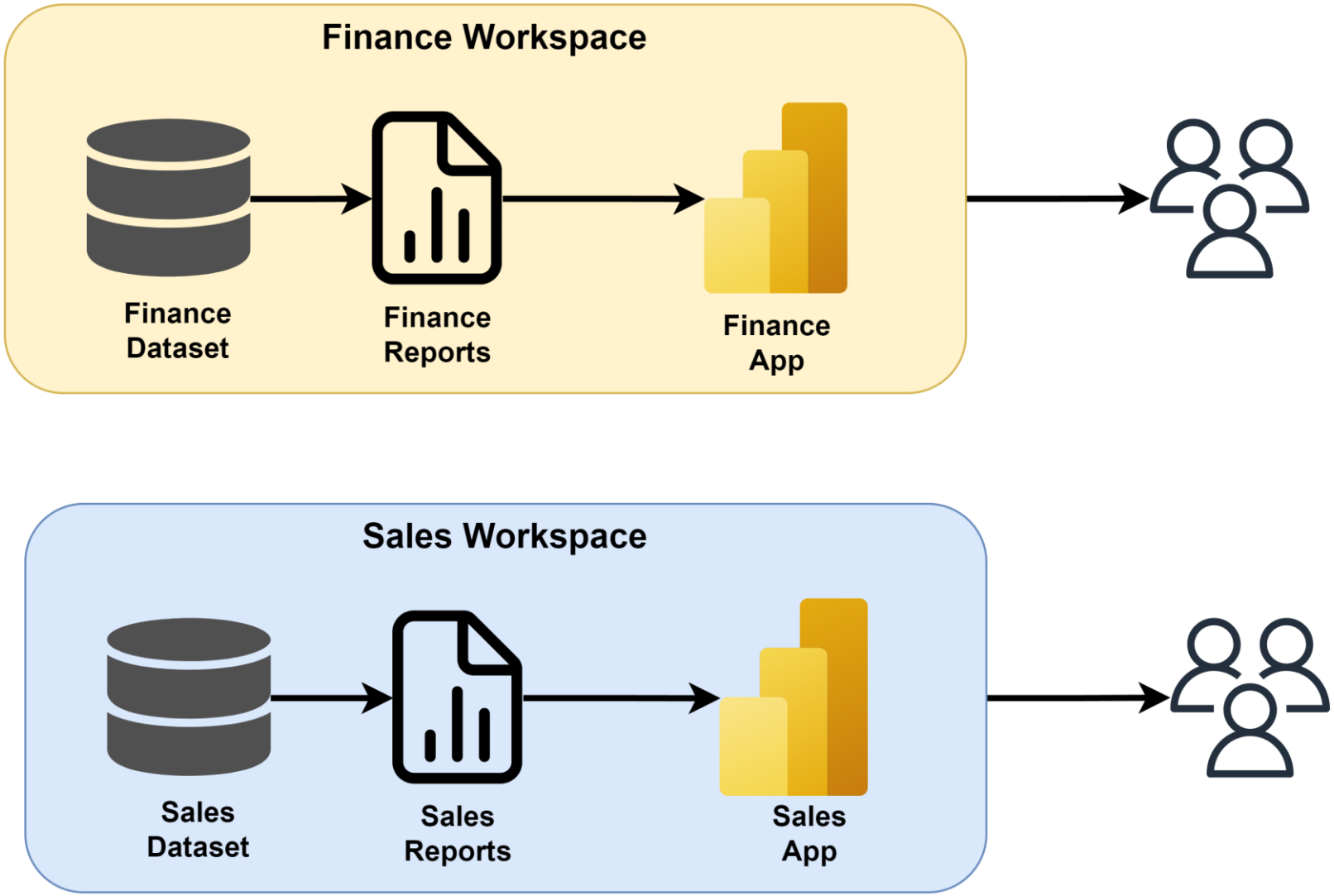
These apps are developed within workspaces, fostering collaborative work on Power BI content alongside your peers. After completion, you can then release the finalized app to a broad audience within your organization.



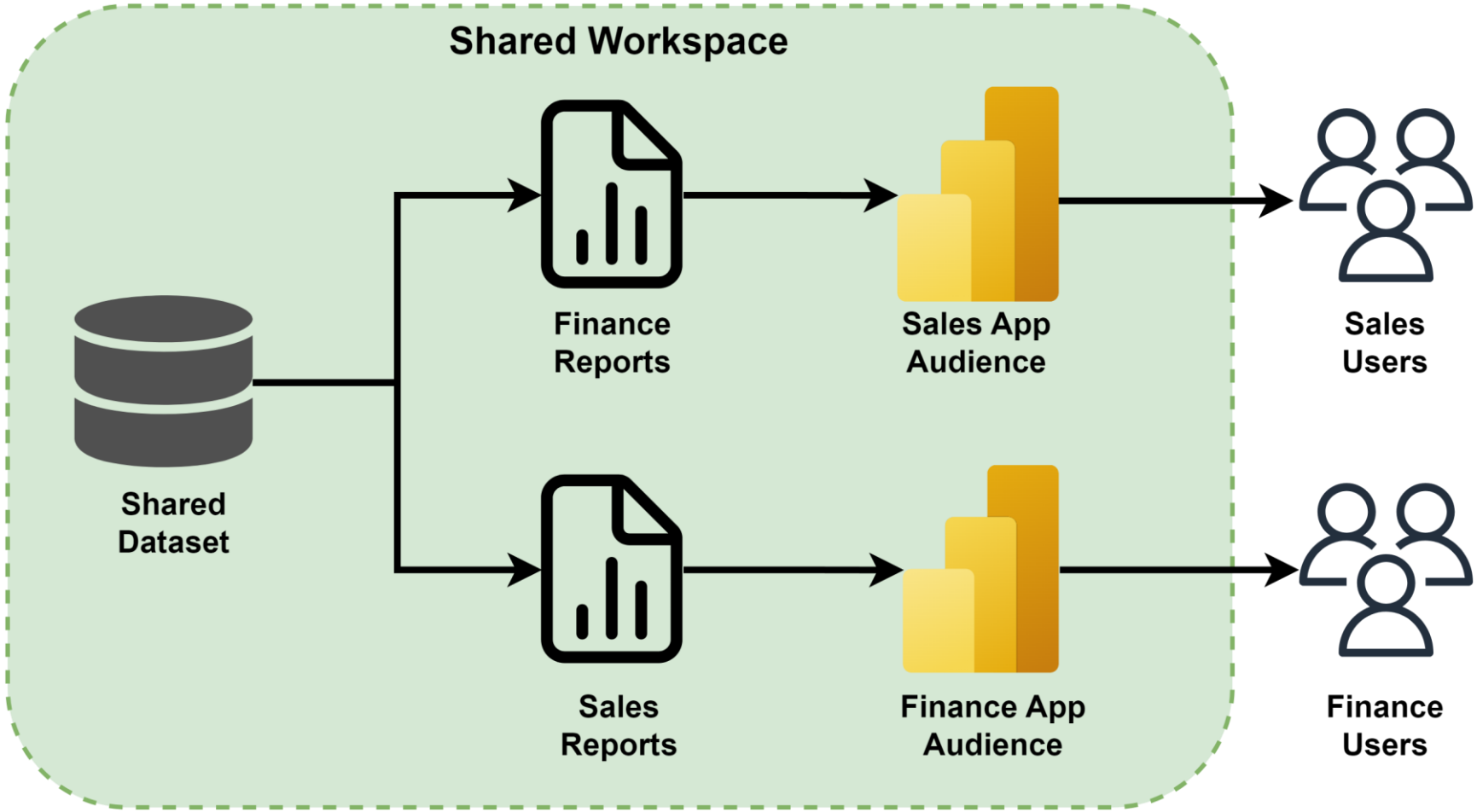
App Benefits of Report Distribution

- ✓ Streamlined Access
- ✓ Enhanced Security
- ✓ Consistency
- ✓ Simplified Sharing
- ✓ Usage Metrics
- ✓ Mobile Access
- ✓ Automatic Updates
- ✓ User-Friendly Experience
- ✓ Scalability
- ✓ Version Control

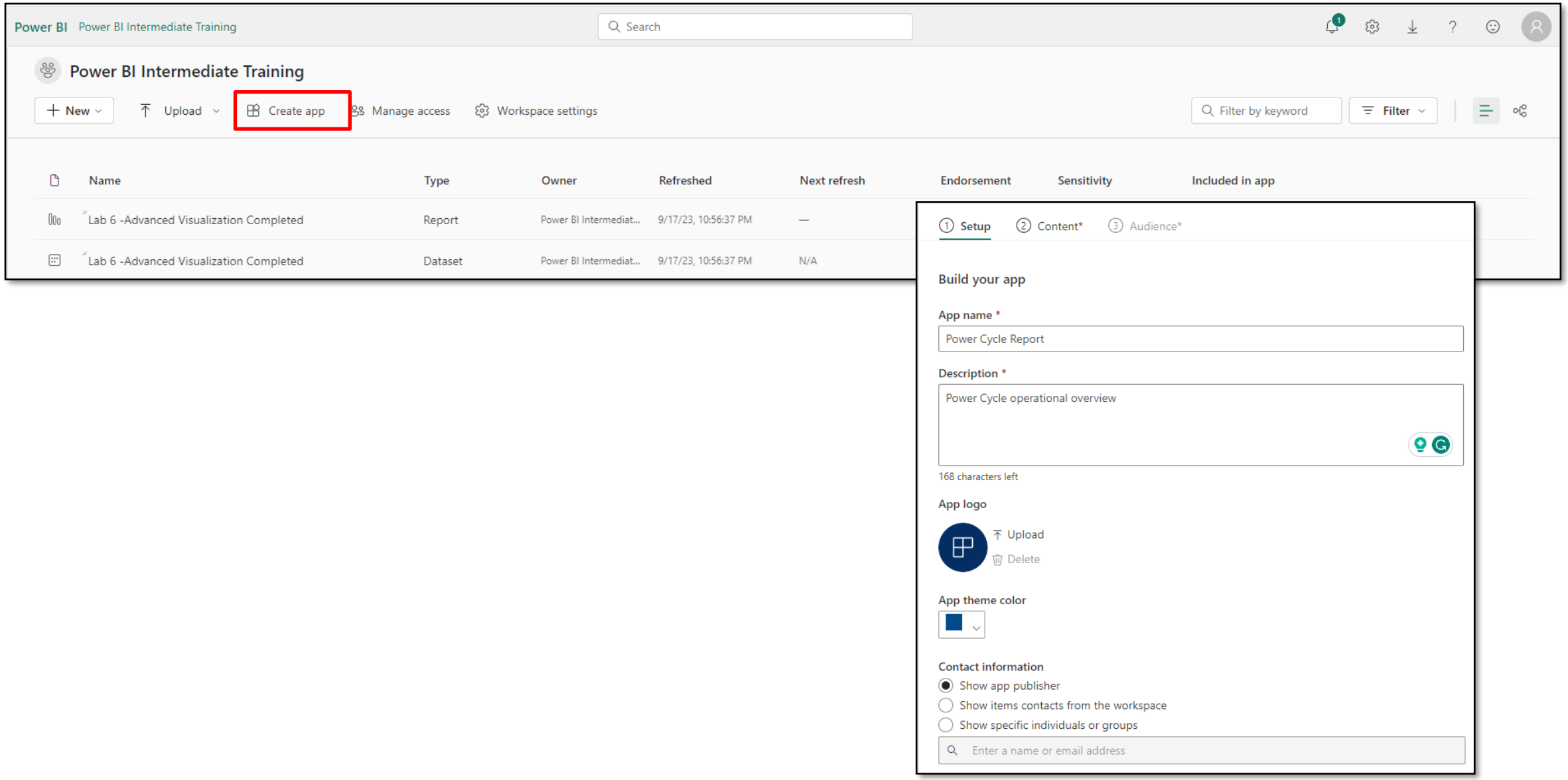
Strategy #1: Separate Workspaces for Each User Group



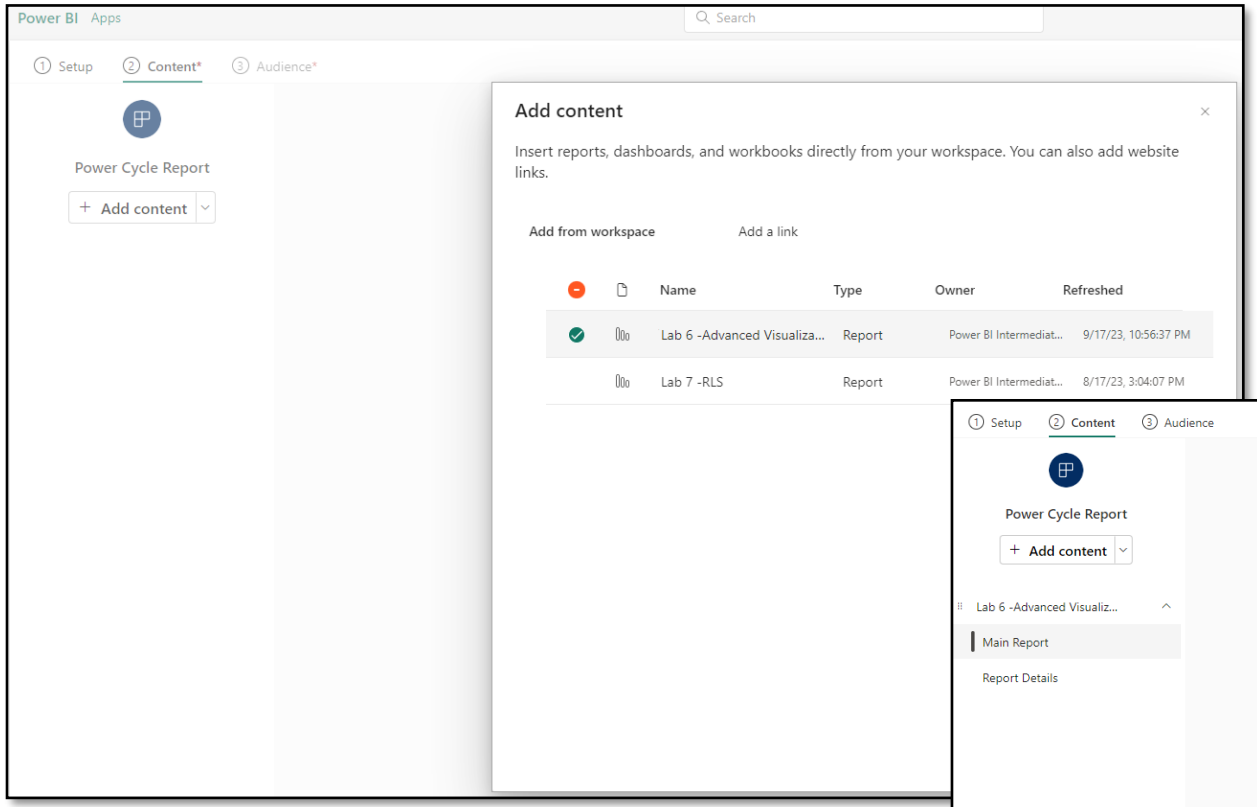
Strategy #2: Shared Workspace for Multiple User Groups

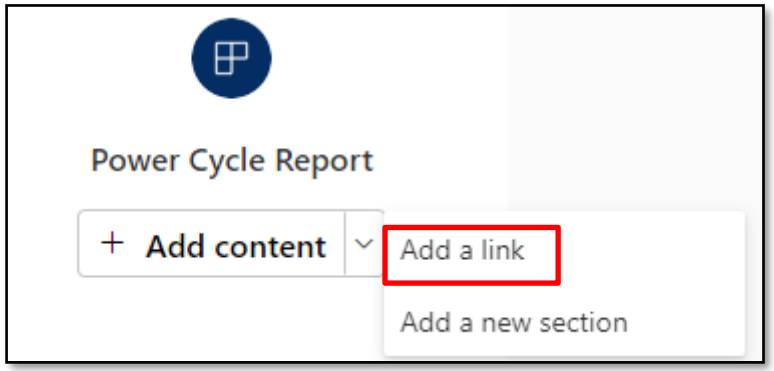


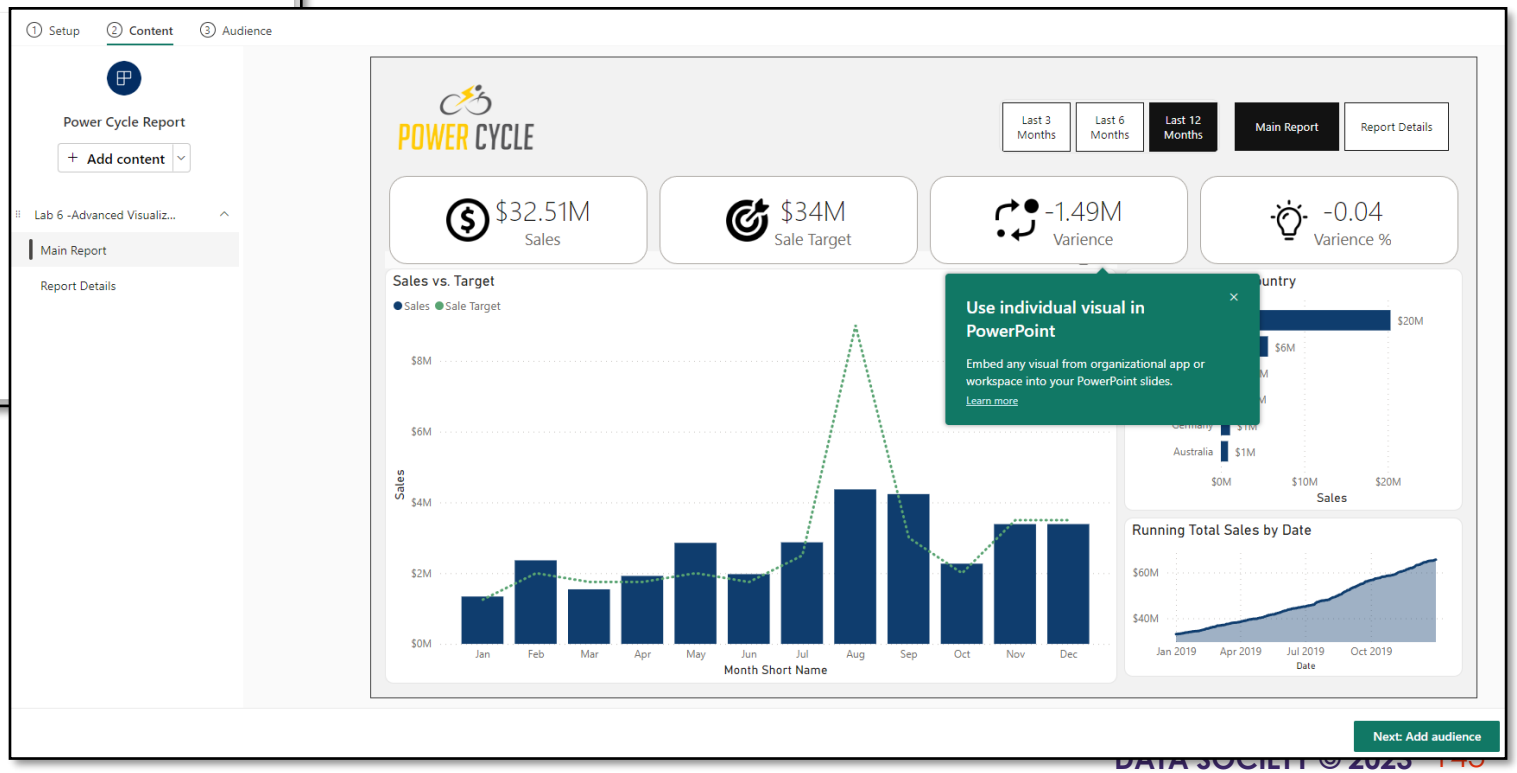
Create and publish your App



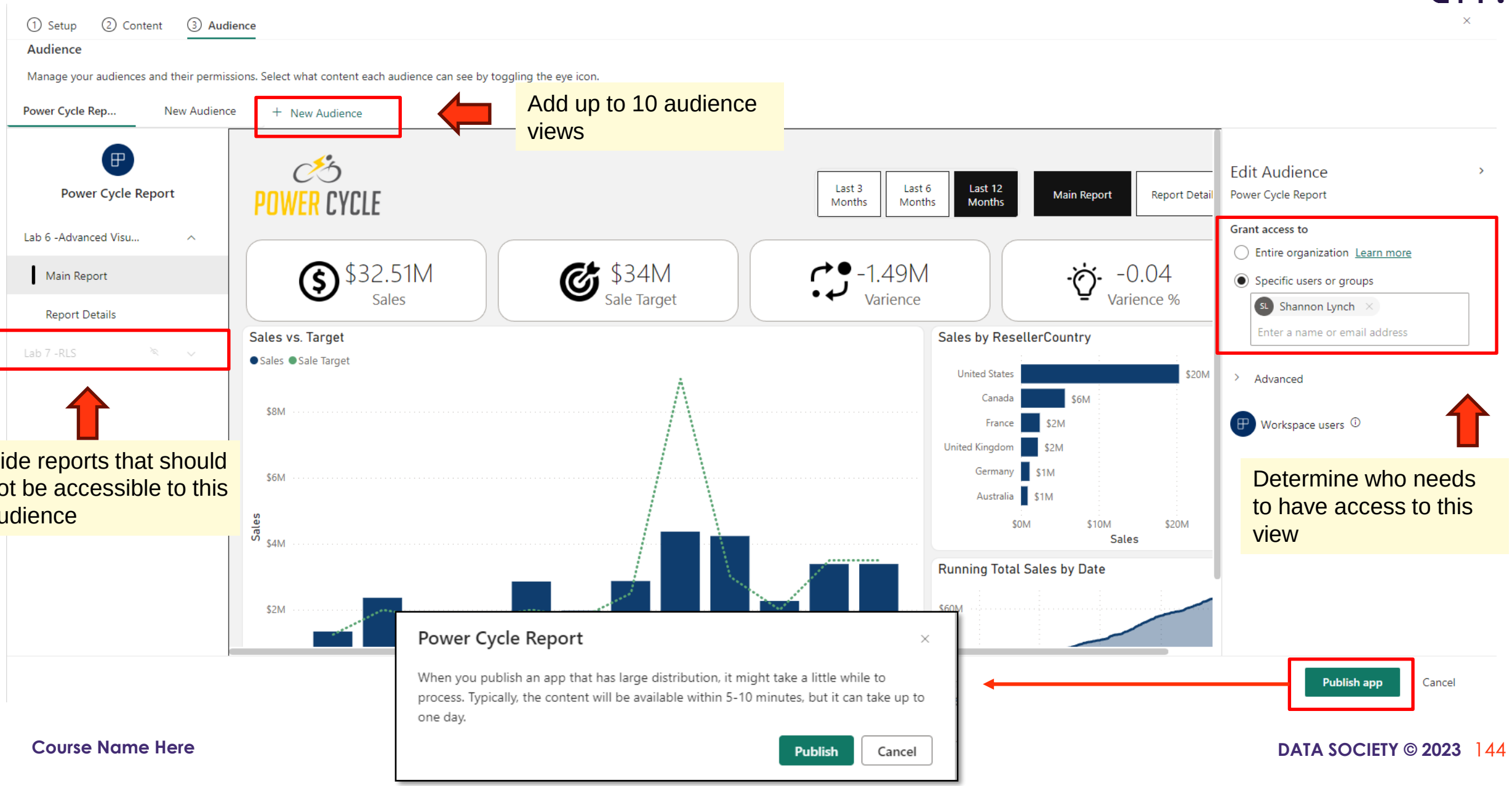
Report Content



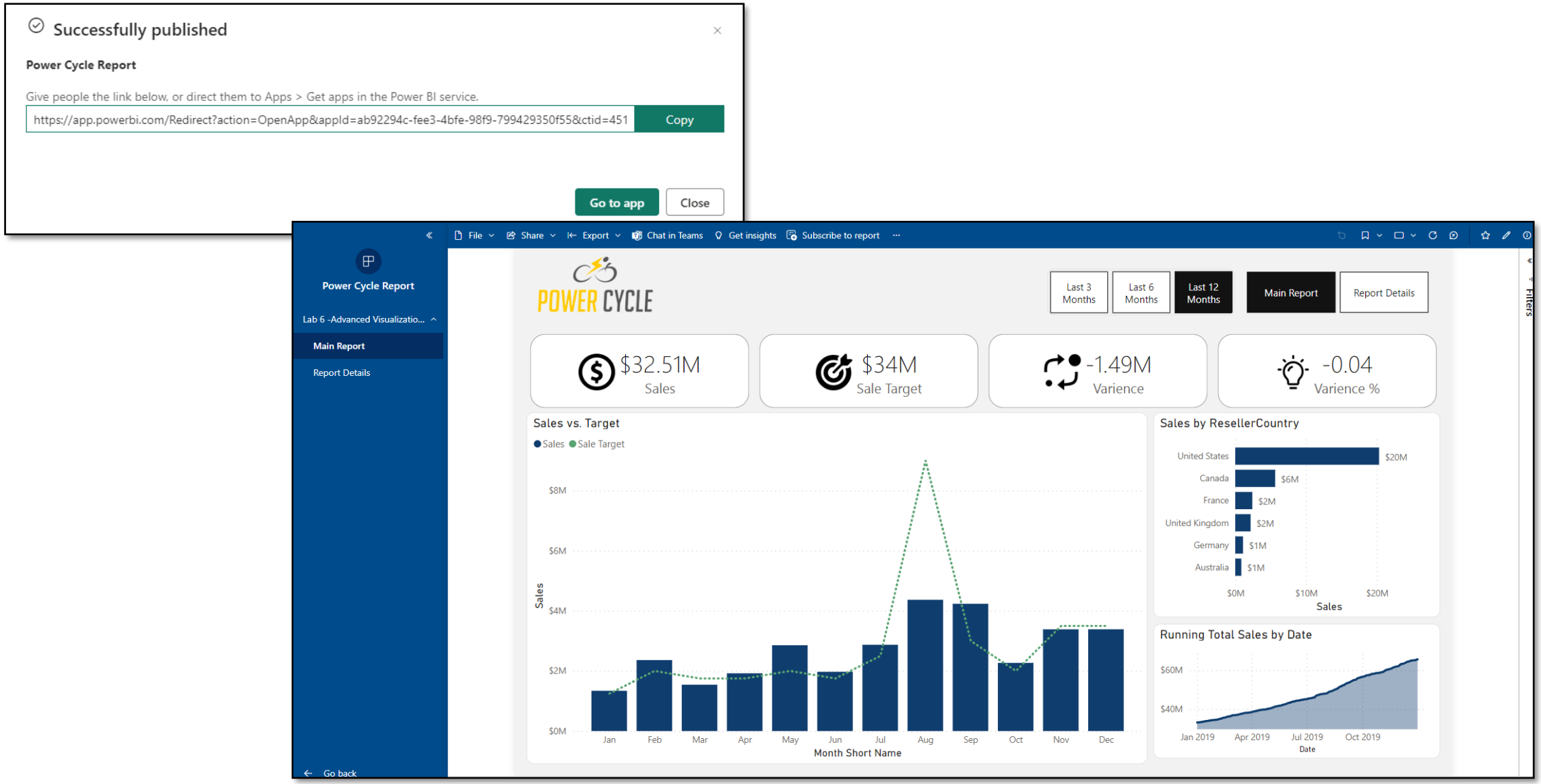




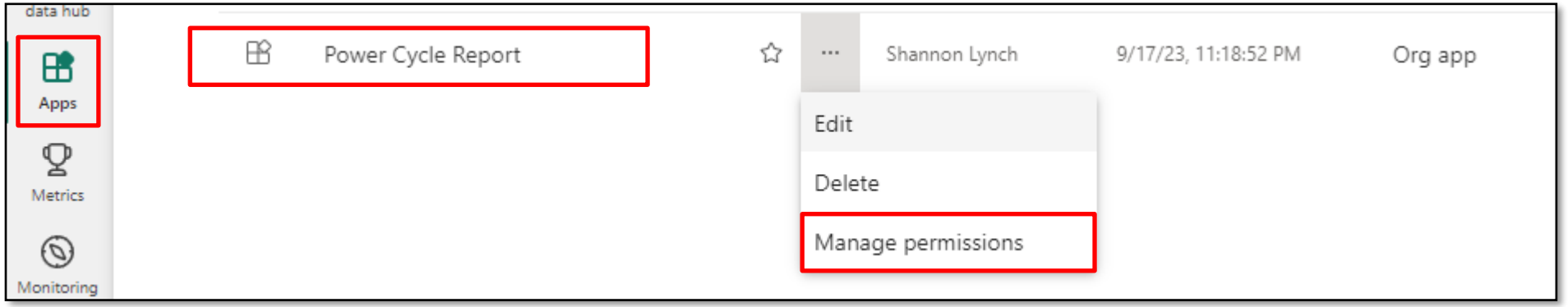
Managing App Audiences

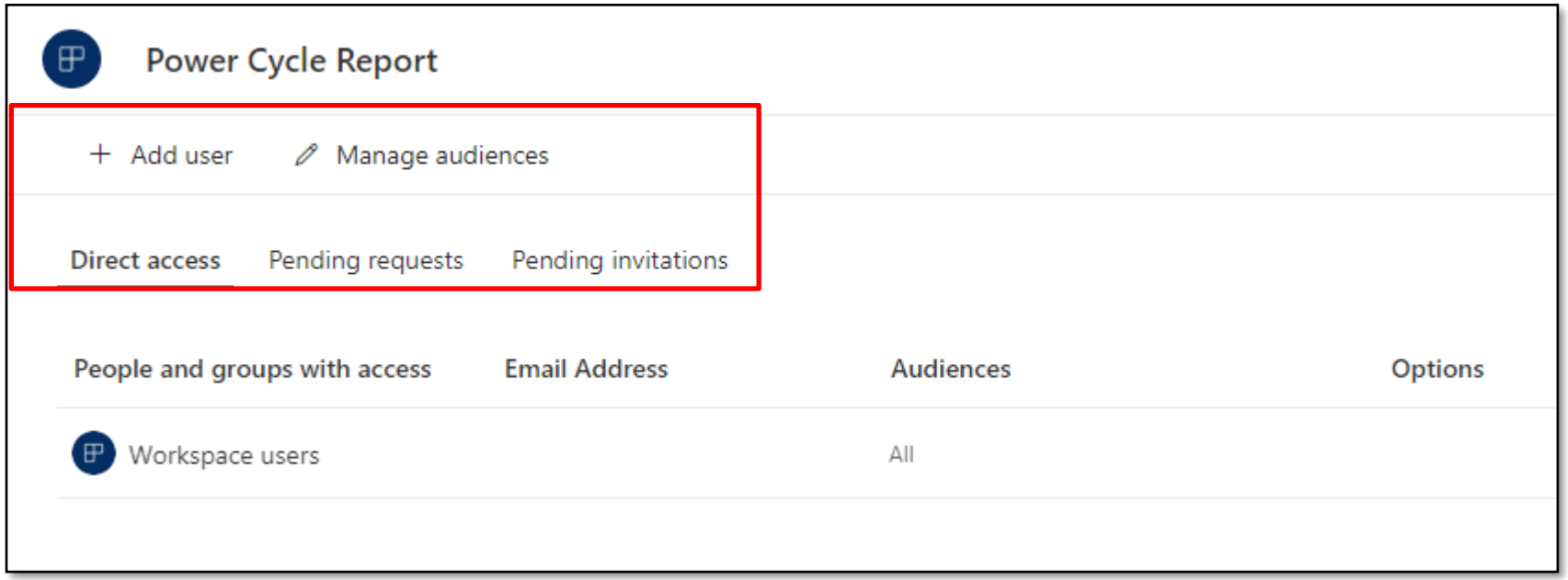


Link to App

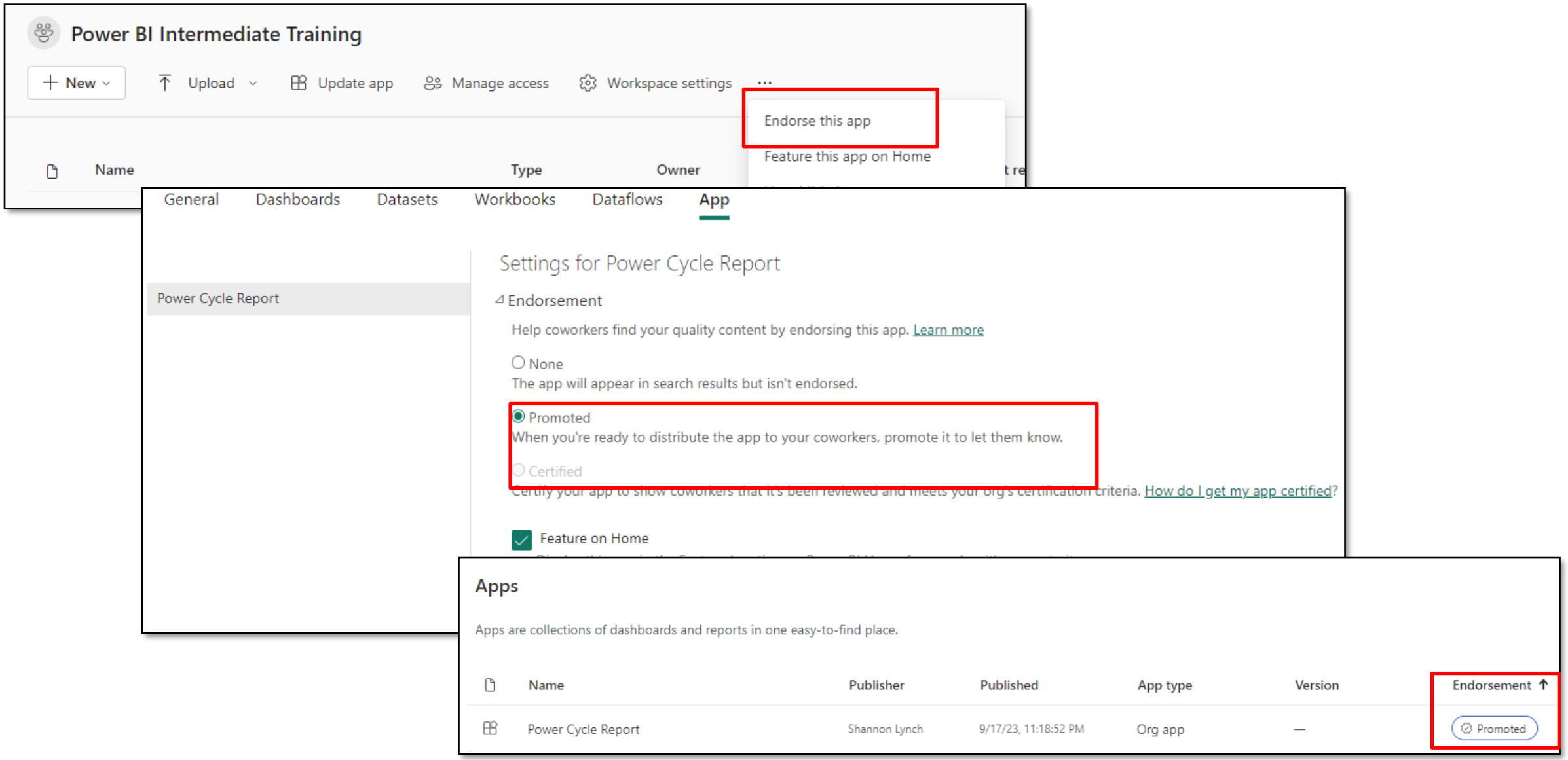


Managing App users

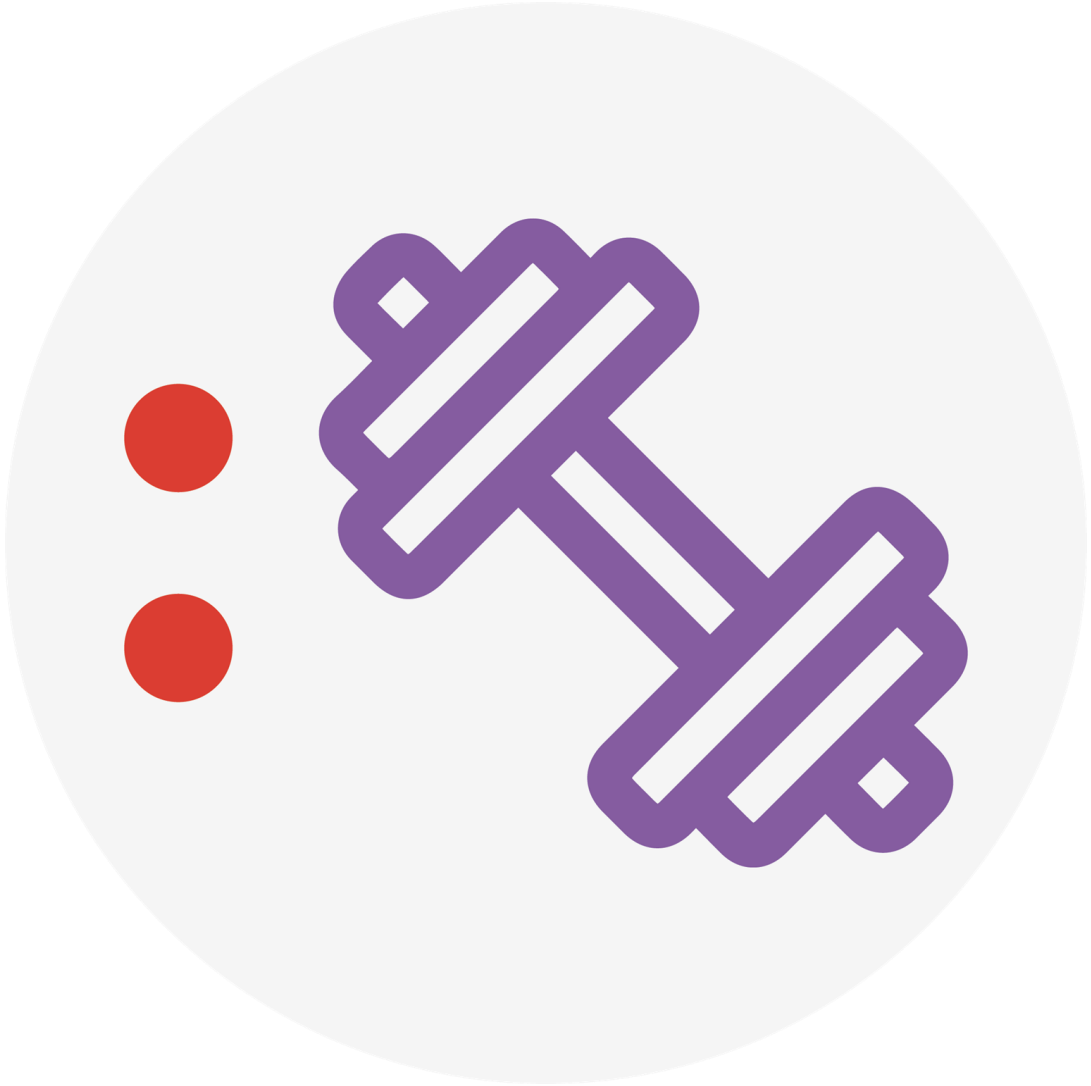




App Endorsement



Live Demo



KC9

Power BI Support Resources

Community.PowerBI.com – Community Forum
Report Theme Gallery – A showcase for stunning report themes
Data Stories Gallery – Get inspired with Data Stories by other Power BI users
R-Visuals Gallery – Get inspired by others use of R for analyzing their data
Store.office.com – Custom PBI visuals and R visuals you can download and use in your story

Power BI Blog - weekly updates
User Voice for Power BI – Vote on (or submit) your favorite new ideas for Power BI
Issues.PowerBI.Com – log issues with the community
Whitepaper - Creating an Enterprise Class Dashboard *Solution with Power BI*
Guided Learning Self Service Power BI training

DAX Formula Language – syntax for DAX
DAX Patterns – Great website to learn new patterns for the DAX Language
Power Query Formula Language – syntax for the "Query" language
Paletton.com – a color scheme generator
https://unicode-table.com/en/ – Unicode Character Table
Theme Generator
Contrast Analyzer: a tool that creates a "lens" to show how people with different visual disabilities might see your reports
Charticulator: a tool that helps to build custom visuals

Thank you!

hello@datasociety.com

1100 15th St. NW, Floor 4
Washington, D.C. 20005

(202)600-9635

datasociety.com