

# Designing Queries to Extract and Transform Data



# Agenda

- Deciding What To Measure
  - Query Design Fundamentals
  - Designing Data Model using a Star Schema
  - Working with the Query Editor Window
  - Importing Content From SharePoint Online
  - Understanding Parameters and Template Files
  - Designing with Function Queries



# Data Discovery

- Data can live in a variety of sources
  - Files (e.g. CSV file, Excel workbook)
  - OLTP Databases
  - OLAP Databases
  - SharePoint Lists and Document Libraries
  - Azure-based services
  - Online services & SaaS applications



# Deciding What To Measure

- You Must Determine Measurable Objectives
  - Financial (revenue, expenses, profit margin, etc.)
  - Business processes efficiency
  - Customer Satisfaction Levels



# Defining Grain Statements

- Grain statements should be defined in initial design phase
  - Grain statements helps determine requirements for BI queries
  - Grain statements can be created & understood by business users
- Example grain statements for BI project at Wingtip Toys
  - What was the total sales revenue over the last 4 years?
  - What was the sales revenue by year, quarter and month?
  - What was the sales revenue by region, state, city and zip code?
  - What was the sales revenue by category, subcategory and product?
  - What was the growth in sales revenue from month to month in 2013?
  - What was profit margin for each product by year, quarter and month?
  - Have their been any products with significantly decreasing profit margin?



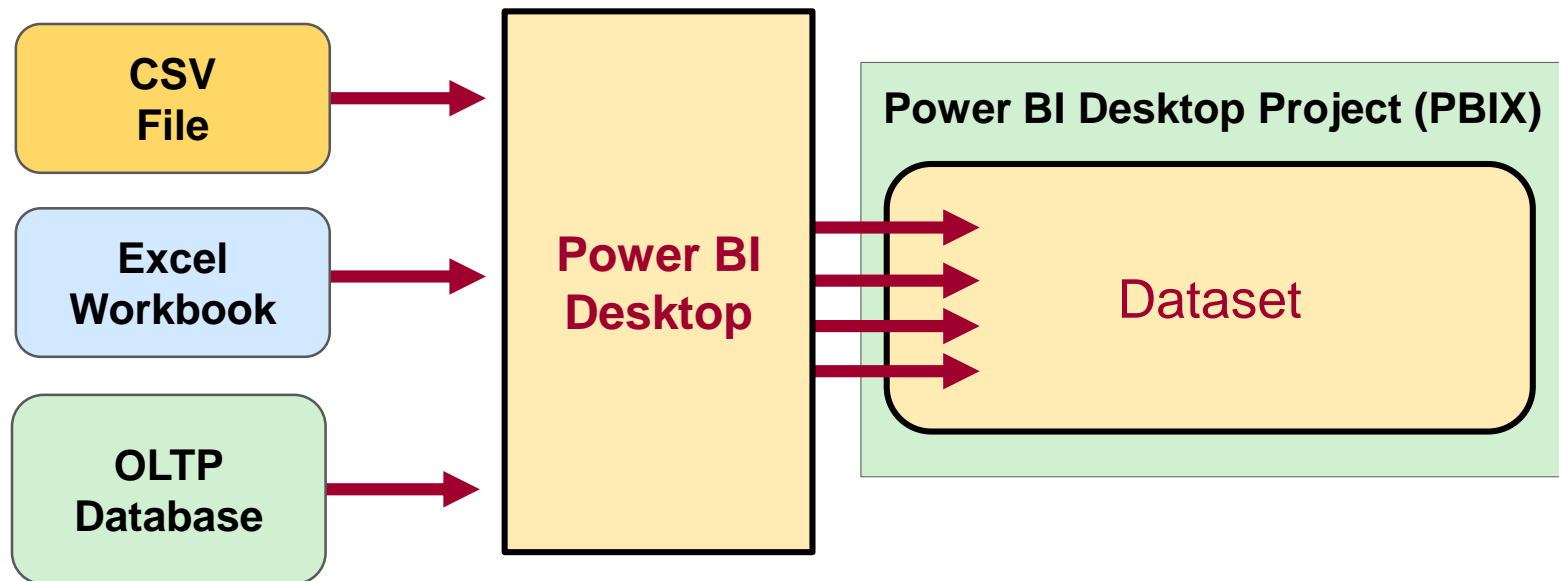
# Agenda

- ✓ Deciding What To Measure
- Query Design Fundamentals
  - Designing Data Model using a Star Schema
  - Working with the Query Editor Window
  - Importing Content From SharePoint Online
  - Understanding Parameters and Template Files
  - Designing with Function Queries



# Power BI Desktop is an ETL Tool

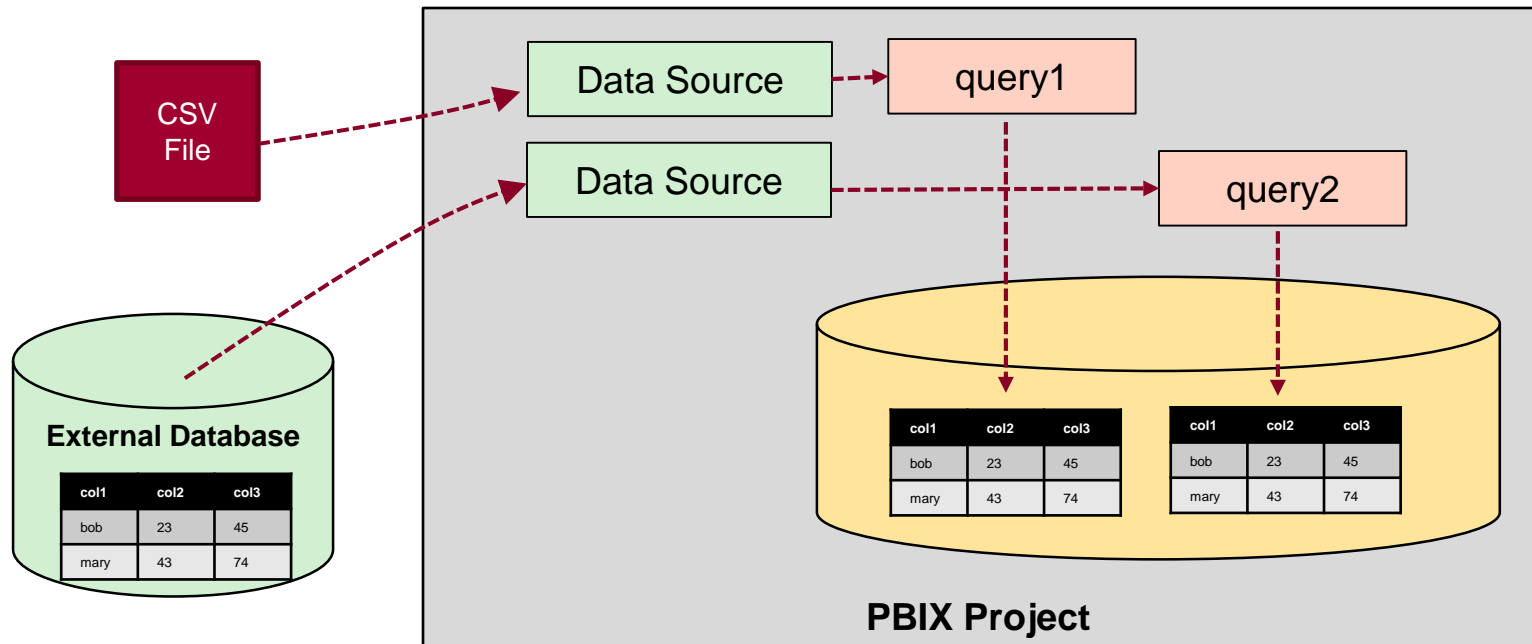
- ETL process is essential part of any BI Project
  - **Extract** the data from wherever it lives
  - **Transform** the shape of the data for better analysis
  - **Load** the data into dataset for analysis and reporting





# Understanding Query Input and Output

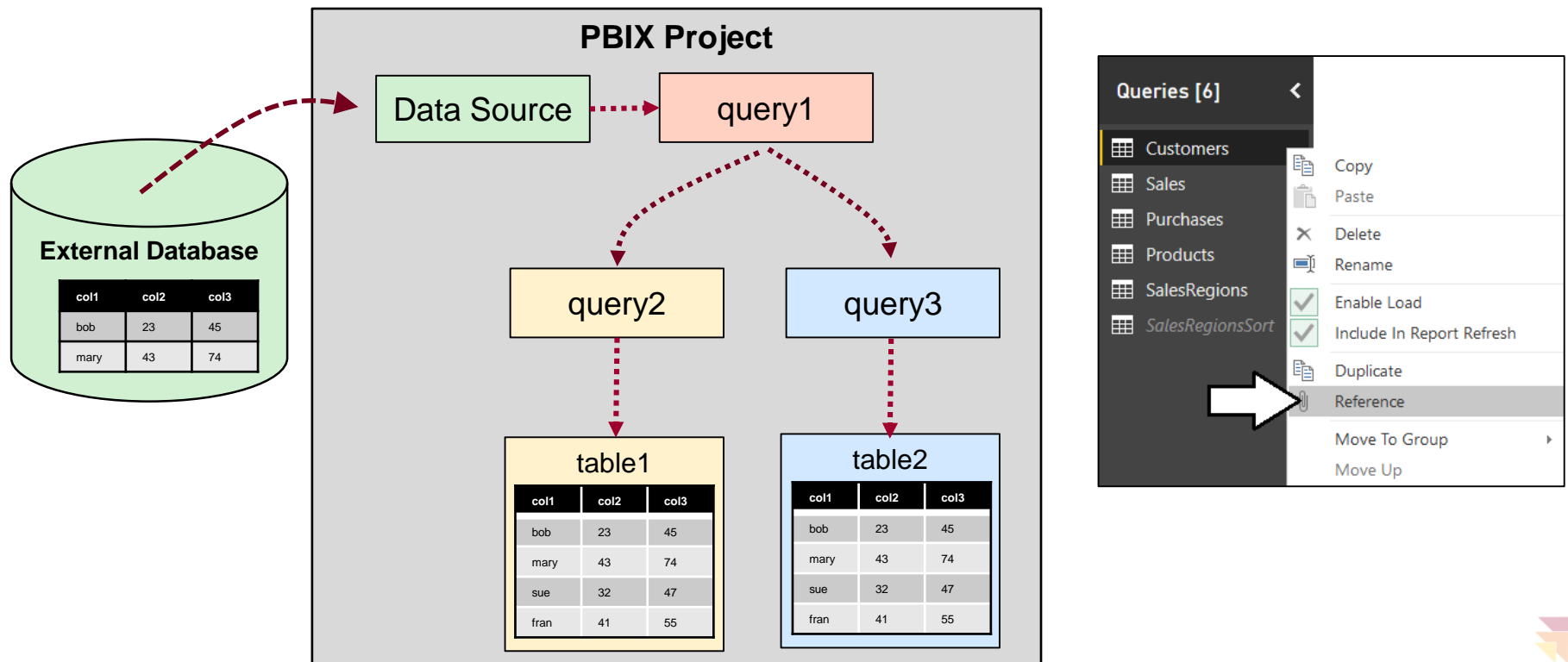
- PBIX project is container for data sources and queries
  - Queries created and saved within scope of Power BI project
  - Queries can pull data from local files
  - Queries can pull data from external content sources
  - Queries main purpose is to load imported data into data model





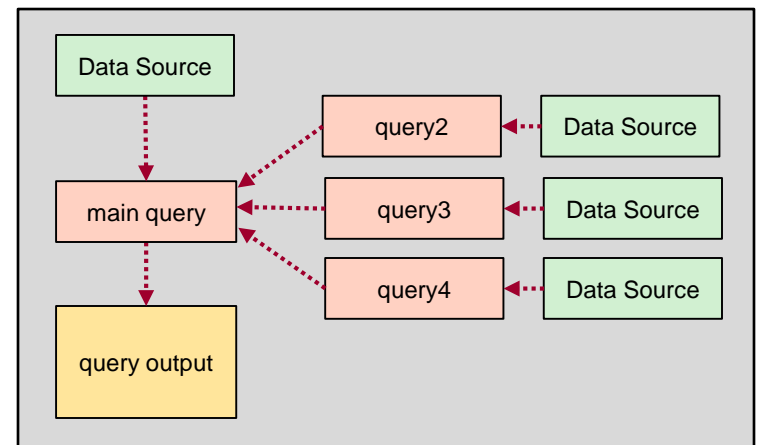
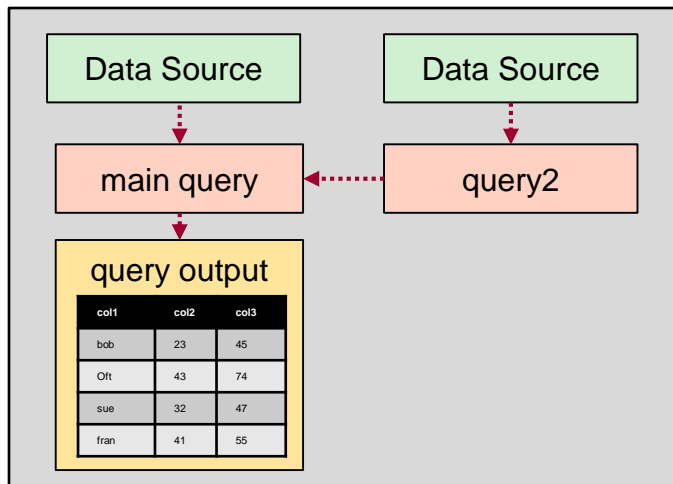
# Query Composition

- Query can serve as source for other queries
  - Allows for creation of reusable base queries & query composition
  - Complexity can be hidden in base queries
  - **Reference** command creates new query based on another query



# Combining Queries

- Query can be merged or appended with another query
  - Merge operation allows you combine columns from two tables
  - Append operation allows you to combine rows from two tables
- Two queries are combined into single output for loading
  - Load settings of main query determines where output is loaded
  - Secondary query acts as source for main query
  - Secondary query can be created with connection-only load setting



# Query Editor Window

- Power BI Desktop provides separate Query Editor window
  - Provides powerful features for designing queries
  - Displays list of all queries in project on the left
  - Displays **Properties** and **Applied Steps** for selected query on right
  - Preview of table generated by query output shown in the middle
  - Query can be executed using **Apply** or **Close & Apply** command



# Query Steps

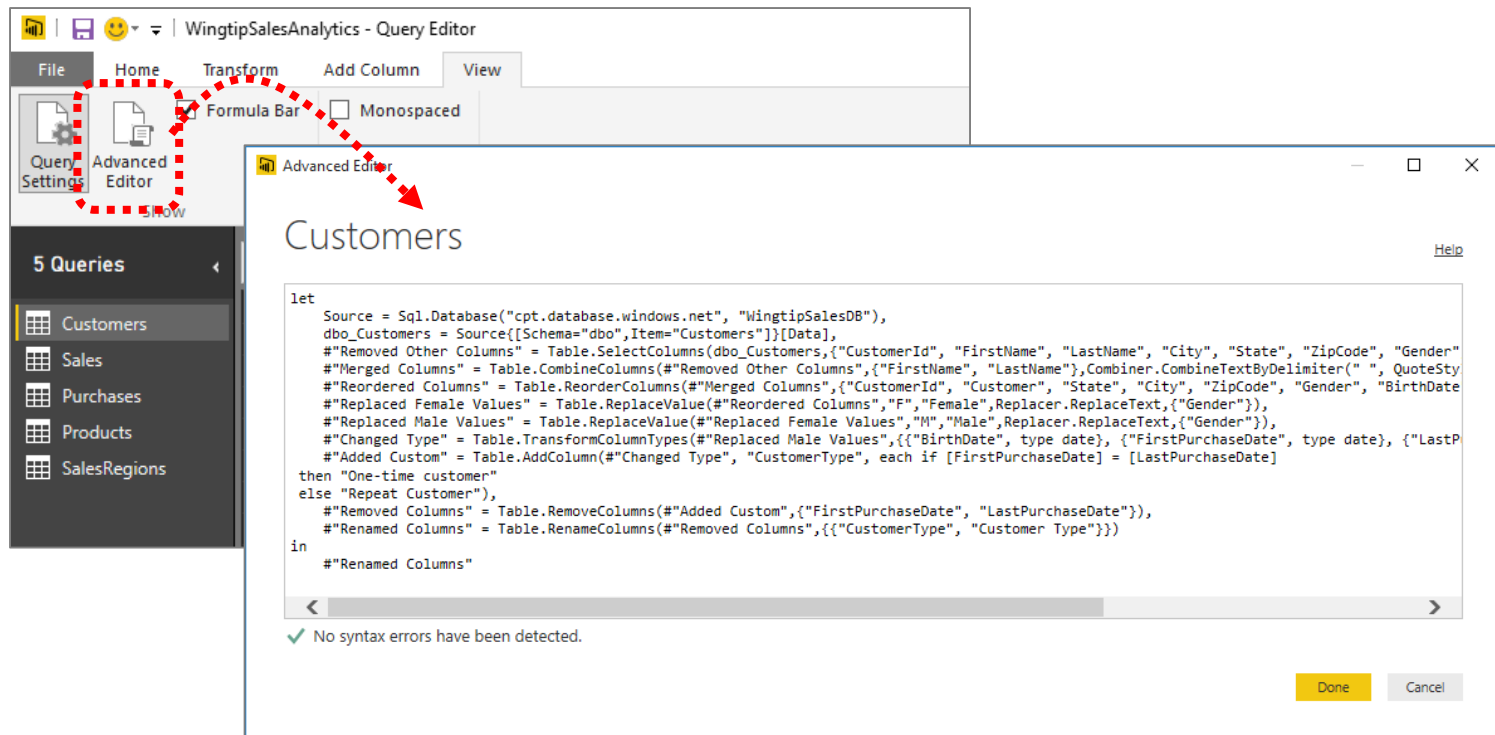
- A query is created as a sequence of steps
  - Each step is a parameterized operation on the data
  - Each step has formula which can be viewed/edited in formula bar
  - Query starts with Source step to extract data from a data source
  - Additional steps added to perform transform operations on data
  - You can replay query operations one by one by clicking on steps

The screenshot displays the Power BI Query Editor interface. At the top, the ribbon includes 'File', 'Home', 'Transform', 'Add Column', and 'View'. Below the ribbon, the 'Formula Bar' is active, showing the formula: `= Table.ReplaceValue("#Replaced Female Values","M","Male",Replacer.ReplaceText,`. A red dashed box highlights the formula bar, with a callout box labeled 'step formula bar' pointing to it. On the left, the 'Queries [6]' pane lists 'Customers', 'Sales', 'Purchases', 'Products', 'SalesRegions', and 'SalesRegionsSort'. The main area shows a data table with columns: CustomerId, Customer, State, City, Zipcode, and Gender. The table contains 14 rows of data. On the right, the 'Query Settings' pane is open, showing the 'Properties' section with 'Name' set to 'Customers'. Below it, the 'Applied Steps' section is highlighted with a red dashed box, showing a sequential list of steps: Source, Navigation, Removed Other Columns, Merged Columns, Reordered Columns, Replaced Female Values, Replaced Male Values (selected), Changed Type, and Added Conditional Column. A callout box labeled 'sequential list of steps for query' points to this list.

CustomerId	Customer	State	City	Zipcode	Gender
1	Nina Diaz	CA	Eureka	95501	Female
2	Melinda Carter	CA	Napa	94558	Female
3	Pam Miller	CA	Napa	94558	Female
4	Merle Blackwell	CA	Sacramento	95823	Female
5	Ariel Hale	CA	Sacramento	95818	Male
6	Randy Carter	CA	Sacramento	95818	Male
7	Lillie Hinton	CA	Eureka	95501	Female
8	Ladonna Moody	CA	Napa	94559	Female
9	Buddy McKay	OR	Bend	97701	Male
10	Warren Sykes	CA	Sacramento	95818	Male
11	Jan Rutledge	OR	Portland	97216	Female
12	Dallas Lester	OR	Eugene	97402	Male
13	Matthew Zimmerman	OR	Portland	97220	Male
14	Sheryl Hernandez	CA	Sacramento	95823	Female

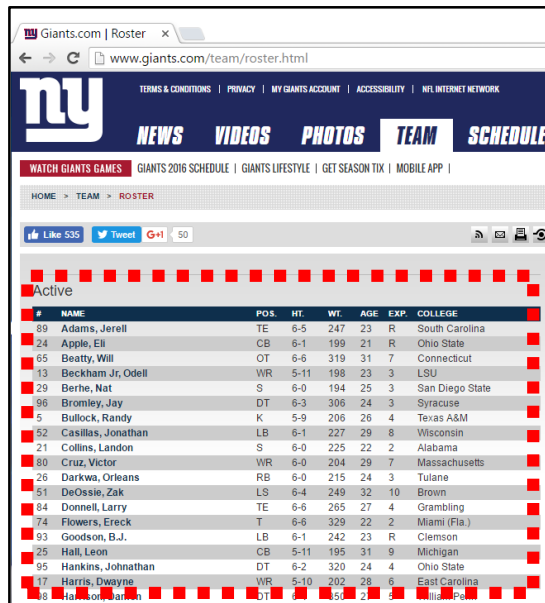
# Advanced Editor

- Power BI Desktop based on "M" functional language
  - Query in Power BI Desktop saved as set of M statements in code
  - Query Editor generates code in M behind the scenes
  - Advanced users can view & modify query code in Advanced Editor



# Working with Web Data Sources

- Many public websites publish data using HTML tables
- Power BI desktop can scrape data from tables in HTML pages



Active

#	NAME	POS.	HT.	WT.	AGE	EXP.	COLLEGE
89	Adams, Jerrell	TE	6-5	247	23	R	South Carolina
24	Apple, Eli	CB	6-1	199	21	R	Ohio State
65	Beatty, Will	OT	6-6	319	31	7	Connecticut
13	Beckham Jr, Odell	WR	5-11	198	23	3	LSU
29	Berhe, Nat	S	6-0	194	25	3	San Diego State
96	Bromley, Jay	DT	6-3	306	24	3	Syracuse
5	Bullock, Randy	K	5-9	206	26	4	Texas A&M
52	Casillas, Jonathan	LB	6-1	227	29	8	Wisconsin
21	Collins, Landon	S	6-0	225	22	2	Alabama
80	Cruz, Victor	WR	6-0	204	29	7	Massachusetts
26	Dar kwa, Orleans	RB	6-0	215	24	3	Tulane
51	DeOssie, Zak	LS	6-4	249	32	10	Brown
84	Donnell, Larry	TE	6-6	265	27	4	Grambling
74	Flowers, Ereck	T	6-6	329	22	2	Miami (Fla.)
93	Goodson, B.J.	LB	6-1	242	23	R	Clemson
25	Hall, Leon	CB	5-11	195	31	9	Michigan
55	Hankins, Johnathan	DT	6-2	320	24	4	Ohio State
17	Harris, Devyne	WR	5-10	202	28	6	East Carolina
16	Hunter, Brian	DT	6-4	255	22	5	Illinois

From Web

Basic Advanced

Enter a Web page URL.

URL

Open file as

OK Cancel

Query Input

	#	Name	Pos.	Ht.	Wt.	Age	Exp.	College
1	89	Adams, Jerrell	TE	6-5	247	23	R	South Carolina
2	24	Apple, Eli	CB	6-1	199	21	R	Ohio State
3	65	Beatty, Will	OT	6-6	319	31	7	Connecticut
4	13	Beckham Jr, Odell	WR	5-11	198	23	3	LSU
5	29	Berhe, Nat	S	6-0	194	25	3	San Diego State
6	96	Bromley, Jay	DT	6-3	306	24	3	Syracuse

Query Output

	Number	Last Name	First Name	Weight	Height	Age	Experience	Position	Category	Side	College
1	89	Adams	Jerrell	247	77	23	0	Tight End	Backs and Receivers	Offense	South Carolina
2	84	Donnell	Larry	265	78	27	4	Tight End	Backs and Receivers	Offense	Grambling
3	45	Tye	Will	262	74	24	1	Tight End	Backs and Receivers	Offense	Stony Brook
4	24	Apple	Eli	199	73	21	0	Cornerback	Defensive Backs	Defense	Ohio State
5	25	Hall	Leon	195	71	31	9	Cornerback	Defensive Backs	Defense	Michigan
6	20	Jenkins	Janoris	198	70	27	5	Cornerback	Defensive Backs	Defense	North Alabama
7	41	Rodgers-Cromartie	Dominique	205	74	30	8	Cornerback	Defensive Backs	Defense	Tennessee State
8	65	Beatty	Will	319	78	31	7	Offensive Tackle	Offensive Line	Offense	Connecticut
9	13	Beckham Jr	Odell	198	71	23	3	Wide Receiver	Backs and Receivers	Offense	LSU
10	80	Cruz	Victor	204	72	29	7	Wide Receiver	Backs and Receivers	Offense	Massachusetts

# Agenda

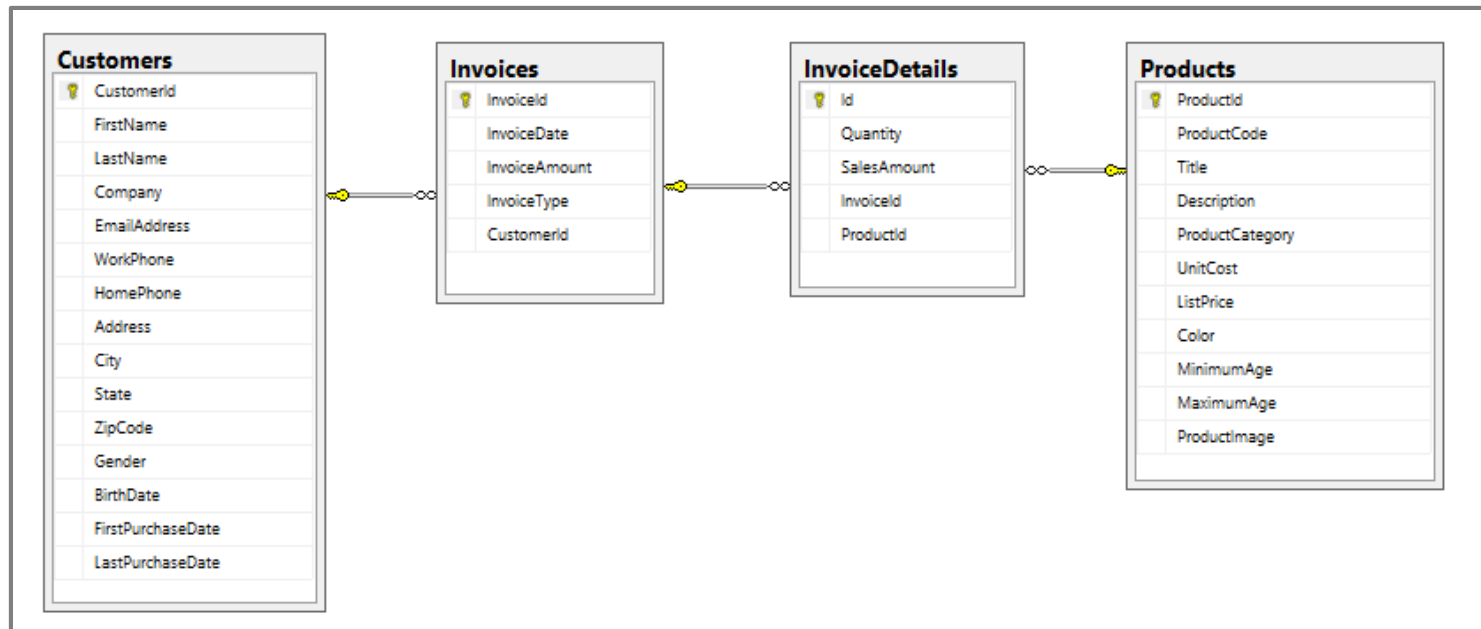
- ✓ Deciding What To Measure
- ✓ Query Design Fundamentals
- Designing Data Model using a Star Schema
  - Working with the Query Editor Window
  - Importing Content From SharePoint Online
  - Understanding Parameters and Template Files
  - Designing with Function Queries





# Sample OLTP Database: WingtipSalesDB

- Online Transaction Processing (OLTP) System
  - Used for real-time data access and transaction-based data entry
  - Optimized for faster transactions (e.g. inserts, updates & deletes)
  - Tables normalized to reduce/eliminate redundancies
  - Table schemas can be hard for business users to understand



# Data Modeling using a Star Schema

- OLAP Modeling often based on Star Schema
  - Tables defined as fact tables or dimension tables
  - Fact tables related to dimension table using 1-to-many relationships



# Designing Queries to Build a Star Schema

- Converts OLTP Data Model to OLAP Data Model
  - Sales table is modeled as a OLAP Fact Table
  - Other tables are modeled as OLAP Dimension tables
  - Requires pulling CustomerId column into Sales table
  - All dimension tables should be directly related to fact table





**DEMO**

# Exploring the Wingtip Sales Analysis Demo Project



# Agenda

- ✓ Deciding What To Measure
- ✓ Query Design Fundamentals
- ✓ Designing Data Model using a Star Schema
- Working with the Query Editor Window
  - Importing Content From SharePoint Online
  - Understanding Parameters and Template Files
  - Designing with Function Queries



# Query Editor Ribbon Tabs

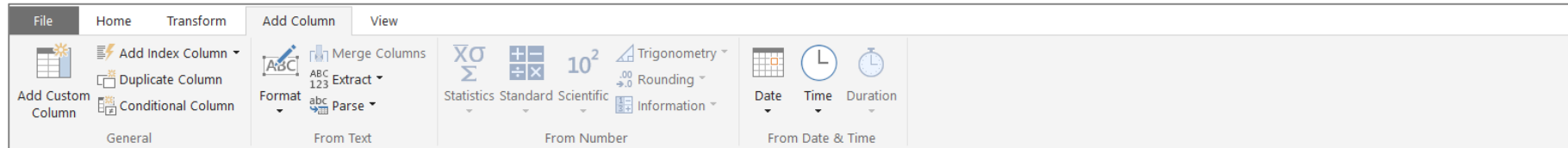
## Home tab



## Transform tab



## Add Column tab



## View tab



# Examples of Basic Power BI Desktop Steps

- Rename column
- Convert column type
- Trim and clean column values
- Replace column values
- Format column values
- Expanding related column
- Merging columns
- Splitting columns

















# Cleaning Data

- Special steps available to clean up string-based data
  - **Transform > Trim** removes whitespace
  - **Transform > Clean** removed non-printable characters



# Converting Column Types

- Transform data to make it more reliable
  - Convert date-time column to date column
- Transform data to make it more efficient
  - Convert decimal to fixed decimal number for currency

 PurchaseDate		 Quantity		 SalesAmount		 ProductCost	
1/28/2012		1		2.95		1.2	Decimal Number
1/28/2012		6				\$	Fixed Decimal Number
1/28/2012		1		19.95			Whole Number
1/28/2012		5		249.75			Date/Time
1/28/2012		1		2.95			Date



# Expanding Related Columns

- Used to pull data from related tables
  - Saves you from performing SQL joins or VLOOKUP

SalesAmount	Invoices	
119.8	Value	Value
29.95	Value	Value
59.9	Value	Value
399.6	Value	Value
29.9	Value	Value
59.8	Value	Value

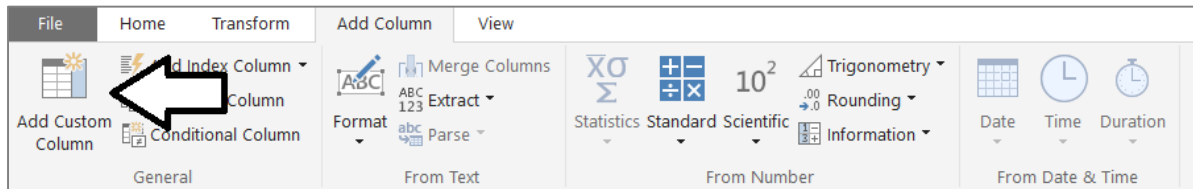
Id	InvoiceId	ProductId	Quantity	SalesAmount	Invoices	Products
1	1	1				Value
2	2	1				Value
3	3	2				Value
4	4	3				Value
5	5	3				Value
6	6	3				Value
7	7	4				Value
8	8	5				Value
9	9	6				Value
10	10	6				Value
11	11	7				Value
12	12	7				Value
13	13	8				Value
14	14	9				Value

Id	InvoiceId	ProductId	Quantity	SalesAmount	InvoiceDate	CustomerId	Products
1	1	1	22	4	119.8	1/28/2012 12:00:00 AM	1 Value
2	2	1	22	1	29.95	1/28/2012 12:00:00 AM	1 Value
3	3	2	22	2	59.9	1/28/2012 12:00:00 AM	2 Value
4	4	3	17	8	399.6	1/28/2012 12:00:00 AM	3 Value
5	5	3	18	2	29.9	1/28/2012 12:00:00 AM	3 Value
6	6	3	18	4	59.8	1/28/2012 12:00:00 AM	3 Value
7	7	4	16	1	2.95	1/28/2012 12:00:00 AM	4 Value



# Adding a Custom Column

- Custom column provide custom logic
  - Logic must be written in M programming language



**Add Custom Column**

New column name:

Custom column formula:  

```
= if [FirstPurchaseDate]=[LastPurchaseDate]  
then "One-time Customer"  
else "Repeat Customer"
```

Available columns:  
CustomerId  
Customer  
State  
City  
ZipCode  
Gender  
BirthDate  
FirstPurchaseDate  
LastPurchaseDate  
CustomerType

<< Insert

[Learn about Power BI Desktop formulas](#)

✓ No syntax errors have been detected.

OK Cancel

FirstPurchaseDate	LastPurchaseDate	CustomerType
1/28/2012	1/28/2012	One-time Customer
1/28/2012	1/28/2012	One-time Customer
1/28/2012	1/28/2012	One-time Customer
1/28/2012	1/28/2012	One-time Customer
1/28/2012	1/28/2012	One-time Customer
1/28/2012	1/28/2012	One-time Customer
1/29/2012	11/22/2015	Repeat Customer
1/29/2012	10/2/2015	Repeat Customer
1/29/2012	1/29/2012	One-time Customer
1/29/2012	5/6/2015	Repeat Customer
1/29/2012	1/29/2012	One-time Customer



# Adding a Conditional Column

- Abstracts away need to write M code



**Add Conditional Column** ✕

Add a conditional column that is computed from the other columns or values.

New column name

	Column Name	Operator	Value		Output
If	FirstPurchaseDate	equals	LastPurchaseDate	Then	One-time Customer

Otherwise

ABC 123	Repeat Customer
---------	-----------------







**DEMO**

## **Using Queries to Transform Data During the Load Process**

# Agenda

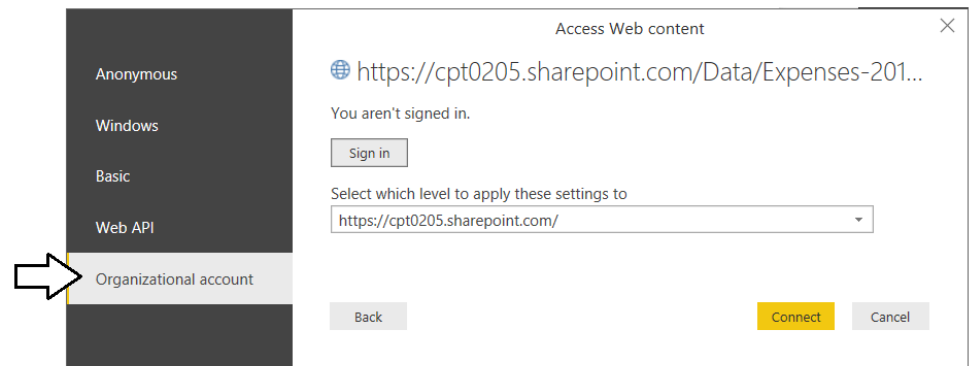
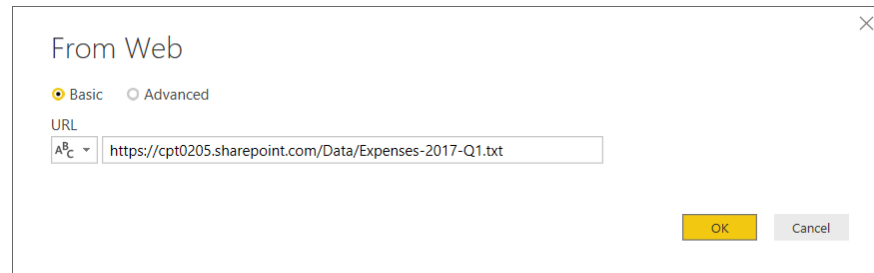
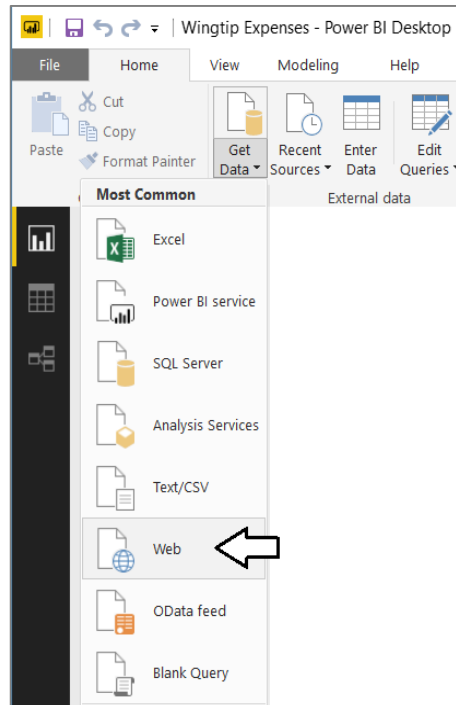
- ✓ Deciding What To Measure
- ✓ Query Design Fundamentals
- ✓ Designing Data Model using a Star Schema
- ✓ Working with the Query Editor Window
- Importing Content From SharePoint Online
  - Understanding Parameters and Template Files
  - Designing with Function Queries





# Importing Files using the Web Datasource

- Files in SharePoint document library exposed via HTTPS
  - Use **Web** datasource to import files in SharePoint Online
  - Use the absolute path to file in document library
  - Authenticate using **Organizational account**



# Importing using the SharePoint Folder

- Select the **SharePoint folder** datasource



- Query returns a row for each file in the site

The query results window displays the URL <https://cpt0205.sharepoint.com> and a table of files. The table has columns for Content, Name, Extension, Date accessed, Date modified, Date created, Attributes, and Folder Path. The data shows five files, all created on 2/3/2018 at 8:09 AM.

Content	Name	Extension	Date accessed	Date modified	Date created	Attributes	Folder Path
Binary	Expenses-2017-Q2.txt	.txt	null	2/3/2018 8:09 AM	2/3/2018 8:09 AM	Record	https://cpt0205.sharepoint.com/Data/
Binary	Expenses-2017-Q1.txt	.txt	null	2/3/2018 8:09 AM	2/3/2018 8:09 AM	Record	https://cpt0205.sharepoint.com/Data/
Binary	Expenses-2017-Q3.txt	.txt	null	2/3/2018 8:09 AM	2/3/2018 8:09 AM	Record	https://cpt0205.sharepoint.com/Data/
Binary	LibertyPowerBISetup.docx	.docx	null	2/3/2018 7:54 AM	2/3/2018 7:54 AM	Record	https://cpt0205.sharepoint.com/Shared Documents/
Binary	RealtimeDashboards.pptx	.pptx	null	2/3/2018 7:54 AM	2/3/2018 7:54 AM	Record	https://cpt0205.sharepoint.com/Shared Documents/

Buttons at the bottom: Combine & Edit, Edit, Cancel.





**DEMO**

## **Importing Content from Files in SharePoint Online**

# Agenda

- ✓ Deciding What To Measure
- ✓ Query Design Fundamentals
- ✓ Designing Data Model using a Star Schema
- ✓ Working with the Query Editor Window
- ✓ Importing Content From SharePoint Online
- Understanding Parameters and Template Files
  - Designing with Function Queries





# Query Parameters

- What is a Query Parameter?
  - Configurable setting with project scope
  - Strongly-typed value to which you can apply restrictions
  - Can be referenced from a query
  - Can be referenced from DAX code in data model
- Where are Parameters commonly used
  - To parameterize data source connection details
  - To filter rows when importing data



# Creating Query Parameters

- Parameters can be created using **Manager Parameters** menu



- Parameter properties

- Name
- Description
- Required
- Allowed Values
- Default Value
- Current Value

Parameters

New

Customer State

Name

Customer State

Description

This parameter is used in the Customers query to filter the customer rows which are loaded into the dataset for the Power BI Desktop project.

☒ Required

Type

Text

Allowed Values

List of values

1	CA
2	OR
3	WA
4	AZ
5	TX
*	

Default Value

CA

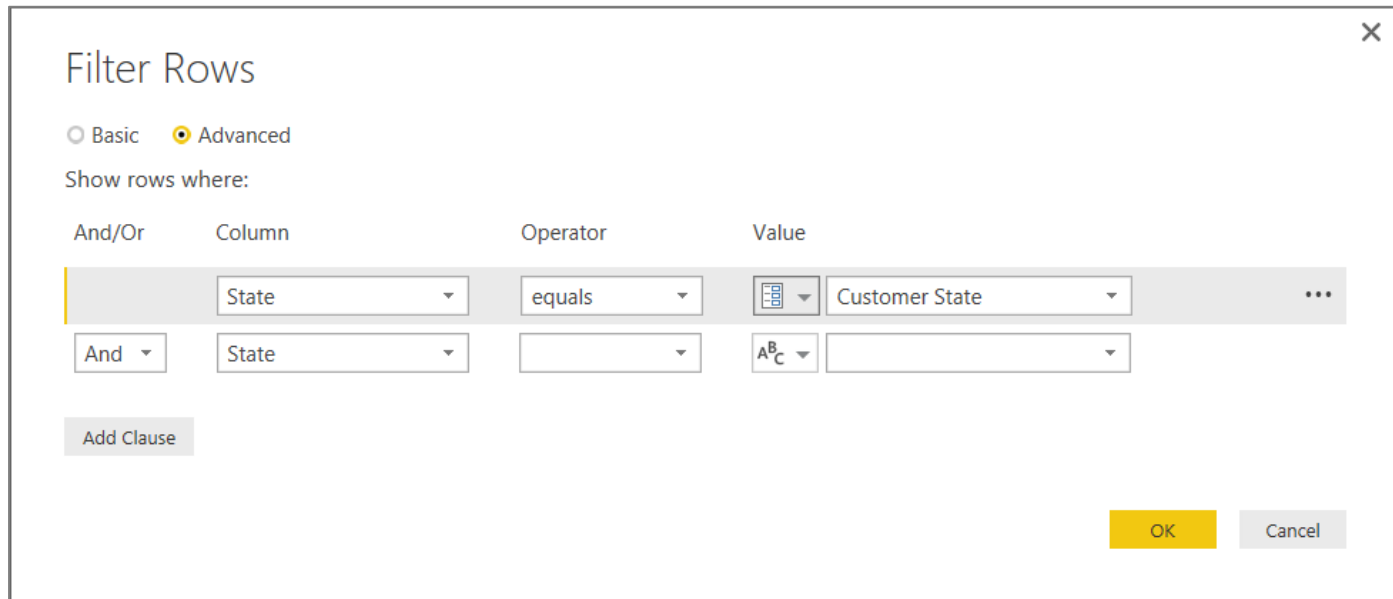
Current Value

CA

OK Cancel

# Referencing Parameters in a Query

- Parameters can be referenced inside query
  - Next query execution uses current parameter value



The screenshot shows a 'Filter Rows' dialog box with a close button (X) in the top right corner. It has two tabs: 'Basic' (unselected) and 'Advanced' (selected). Below the tabs, it says 'Show rows where:'. There are two rows of filter criteria. The first row has a column 'State', an operator 'equals', and a value 'Customer State'. The second row has a column 'State', an operator (empty), and a value 'A^B\_C'. There is an 'Add Clause' button at the bottom left and 'OK' and 'Cancel' buttons at the bottom right.

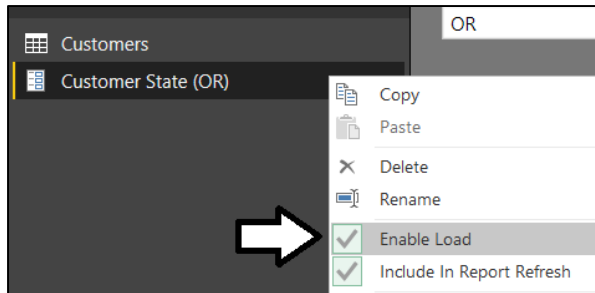
And/Or	Column	Operator	Value
	State	equals	Customer State
And	State		A^B_C



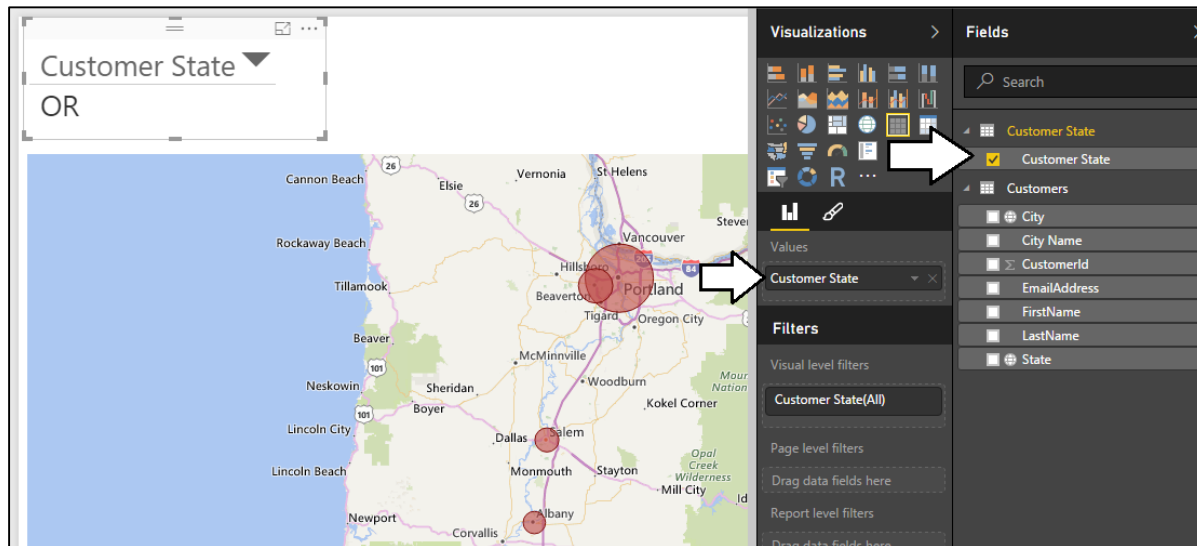


# Making Parameters Available to Data Model

- Configure parameter's Enable Load setting

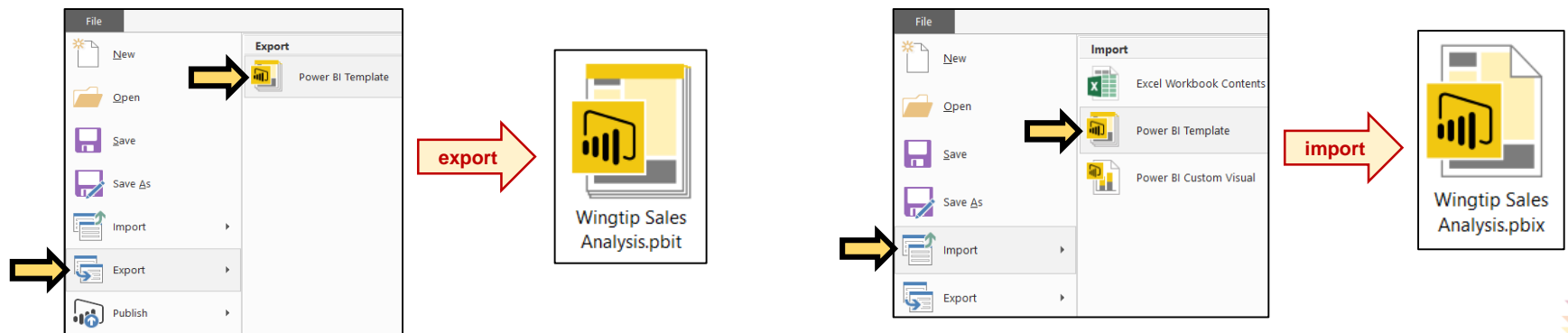


- Parameter becomes visible within fields list in report view



# Power BI Project Template Files

- PBIX project can be exported to project template file
  - Template file created with PBIT file extension
  - Generated template files contains everything except for the data
  - PBIT template file can be imported to create new PBIX projects
  - Template files are powerful when used together with parameters
- How are template files used?
  - Export PBIX project to create a PBIT template file
  - Import the PBIT template file to create a new PBIX project



# Agenda

- ✓ Deciding What To Measure
- ✓ Query Design Fundamentals
- ✓ Designing Data Model using a Star Schema
- ✓ Working with the Query Editor Window
- ✓ Importing Content From SharePoint Online
- ✓ Understanding Parameters and Template Files
- Designing with Function Queries



# Understanding Function Queries

- Query can be converted into reusable function
  - Requires editing query M code in Advanced Editor
  - Function query can be defined to accept parameters

```
GetExpensesFromFile

(FilePath as text) =>

let
    Source = Csv.Document(Web.Contents(FilePath))
    #"Changed Type" = Table.TransformColumnTypes
```

- Function query can't be edited with visual designer







**DEMO**

## Creating a Function Query

# Summary

- ✓ Deciding What To Measure
- ✓ Query Design Fundamentals
- ✓ Designing Data Model using a Star Schema
- ✓ Working with the Query Editor Window
- ✓ Importing Content From SharePoint Online
- ✓ Understanding Parameters and Template Files
- ✓ Designing with Function Queries

