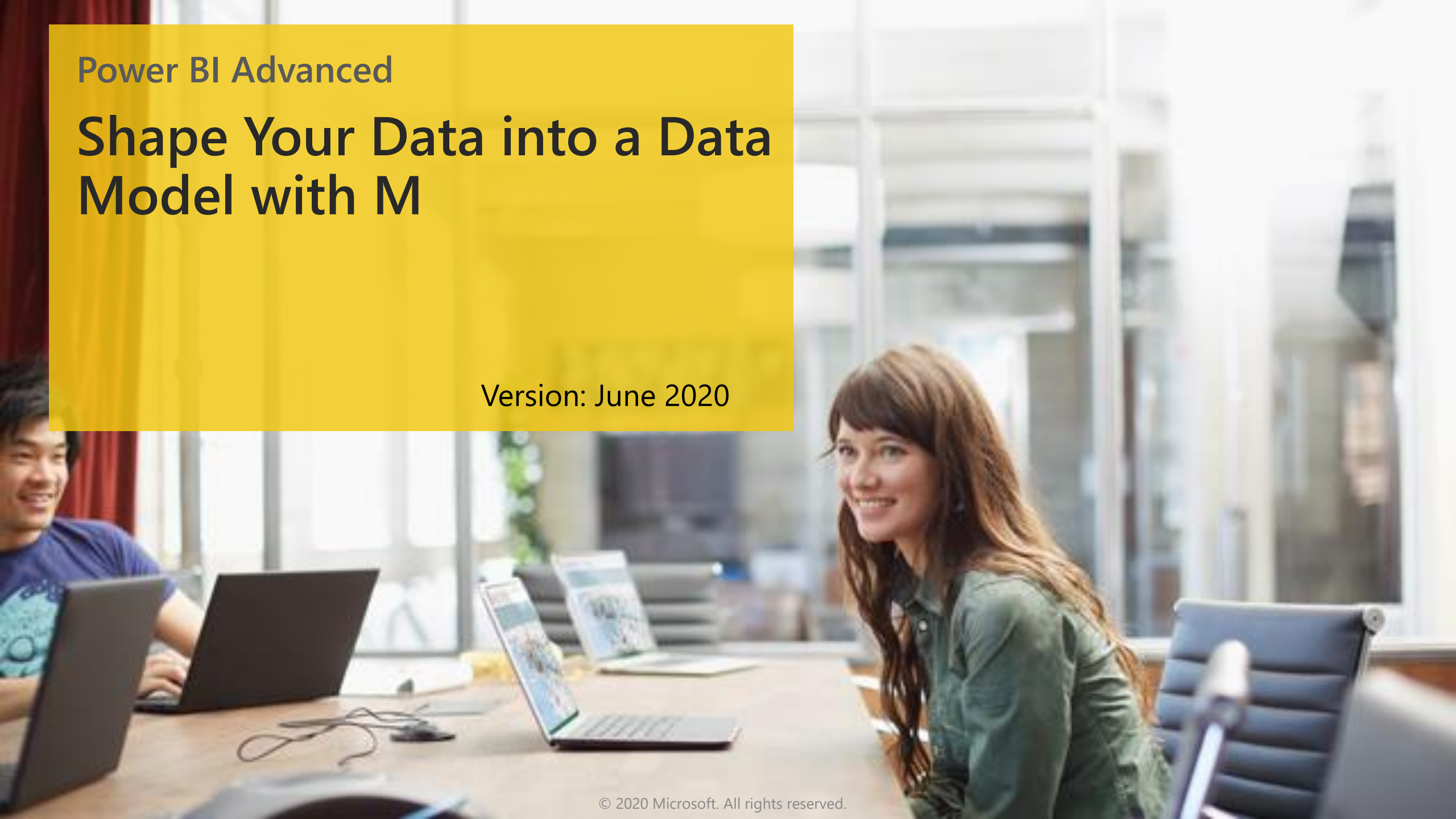
Power BI Advanced

Shape Your Data into a Data Model with M

Version: June 2020



Prerequisites and Setup Steps

Internet connectivity: You must be connected to the internet

- At minimum, a computer with 2-cores and 4GB RAM running Windows 8 / Windows Server 2008 R2 or later
- Microsoft Power BI Desktop requires Internet Explorer 10 or greater
- Verify if you have 32bit or 64bit operating system to decide if you need to install the 32bit or 64bit applications.
    - Search for computer on your PC, right click properties for your computer
    - You will be able to identify if your operating system is 64 or 32 bit based on "system type" as shown below

Download and install Power BI Desktop: Download and install Microsoft Power BI Desktop from http://www.microsoft.com/en-us/download/details.aspx?id=45331. Optionally, you can also install the Power BI Desktop tool from the **Power BI Desktop Install** folder on the flash drive that will be provided on the day of the session. Please choose appropriate 64-bit or 32-bit version depending on your platform. Microsoft Power BI Desktop is available for 32-bit (x86) and 64-bit (x64) platforms

Download Class Files:

Download files from: https://aka.ms/PBI-Shaping-inst

Open a browser to the Power Query Reference page:

https://docs.microsoft.com/en-us/powerquery-m/power-query-m-function-reference

NOTE: This lab is using real anonymized data and is provided by ObviEnce LLC. Visit their site to learn about their services: www.obvience.com.

This data is property of ObviEnce LLC and has been shared for the purpose of demonstrating PowerBI functionality with industry sample data. Any uses of this data must include this attribution to ObviEnce LLC.

Agenda *(times are approximate and will be fluid with the*

By the end of this course, you will be able to use *Power BI Desktop* to import and shape data from a variety of different sources. Specifically you will be able to:

- Understand the *Power BI Desktop* data model, its components and most effective schemas
- Gain an understanding of the Power Query M language
- Import data from a variety of sources (including XLS and CSV files)

- Create queries using toolbar navigations and Advanced Editor
- Understand parameters
- Organize queries using folders

Agenda *(times are approximate and will be fluid with the class)*

09:00 AM – 09:30 AM – Module 1
09:30 AM – 09:45 AM – Lab 01 – Import Data
09:45 AM – 10:15 AM – Module 2
10:15 AM – 11:30 AM – Lab 02 –Introduction to M
11:30 AM – 12:00 PM – Module 3 – Advanced M

12:00 – 12:30 – Break

12:30 PM – 01:15 PM – Lab 03 – Create Budget Fact table
01:15 PM – 01:45 PM – Module 4 – Variables, Parameters, and Functions
01:45 PM – 02:45 PM – Lab 04 – Variables, Parameters, and Functions
02:45 PM – 03:15 PM – Module 5
03:15 PM – 03:45 PM – Lab 05 - Optimization Techniques

MODULE 1:
Basic Data Modeling

MODULE OBJECTIVES

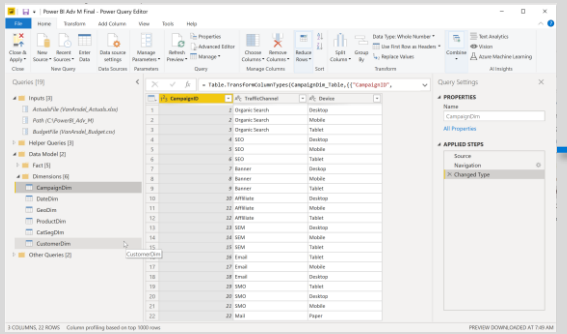
Objectives:

- Understand the basic architecture of PBI Desktop
- Understand Power BI modeling terminology

Power BI Desktop Data Flow

Data Sources

Power BI Desktop file (.PBIX)



Query Editor

Data Source Connections

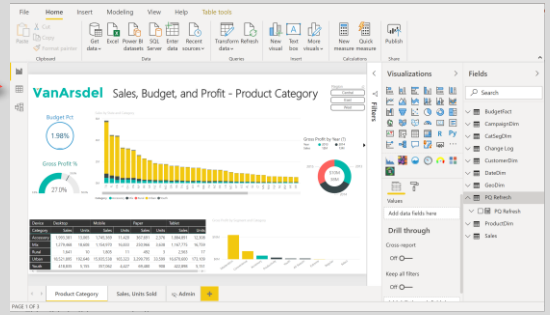
Data Transformations

(Prep data for Data Model)

Close & Apply

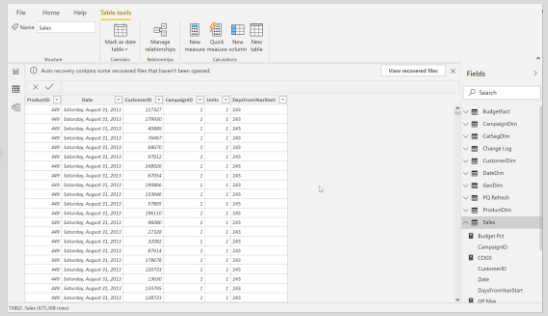
Close: Closes Query Editor

Apply: Loads data from sources to Data Model



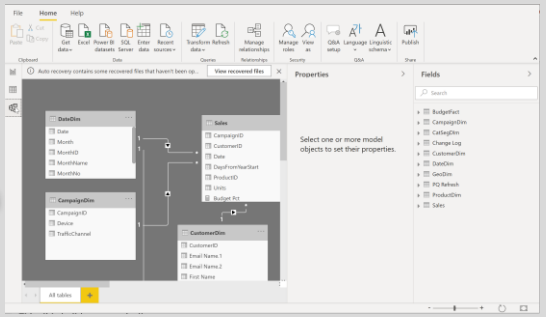
Report

Create Visuals



Data

View Tables



Relationships

See how Tables relate to each other

What is a Data model?

- Improves understandability of the data
- Increases performance of dependent processes and systems
- Increases resilience to change

Why is it important to have a Good Data model?

Basic Data Modeling

Components of a data model – Fact Table

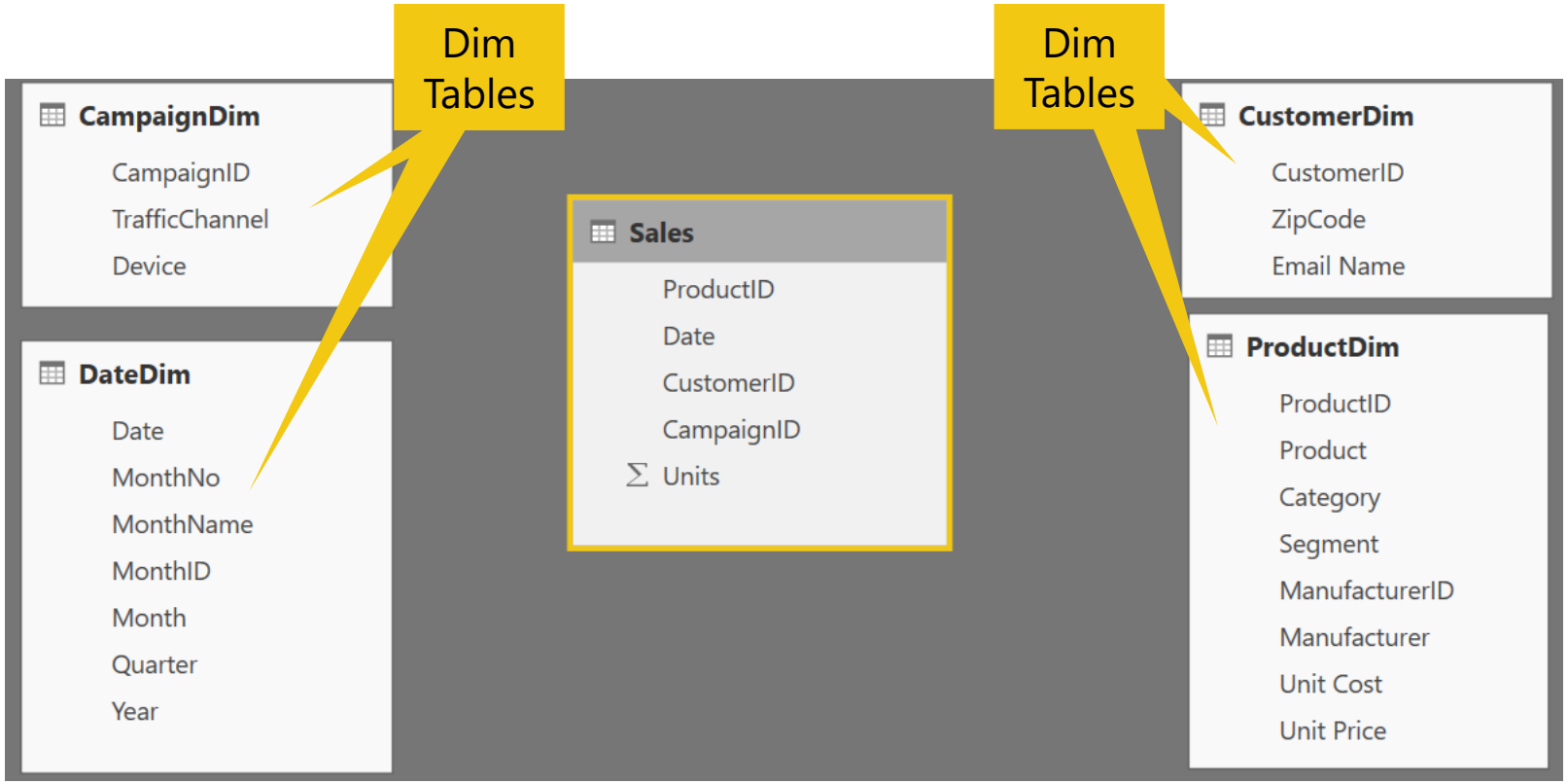
Fact Table

- Contains Measures
  (or items to be aggregated)
  of a business process
  Examples:
  - Transactions
  - Sales Revenue
  - Units
  - Cost

- Measures are usually sliceable.
  Examples:
  By Month, By Customer

Basic Data Modeling

Components of a data model – Dim Table



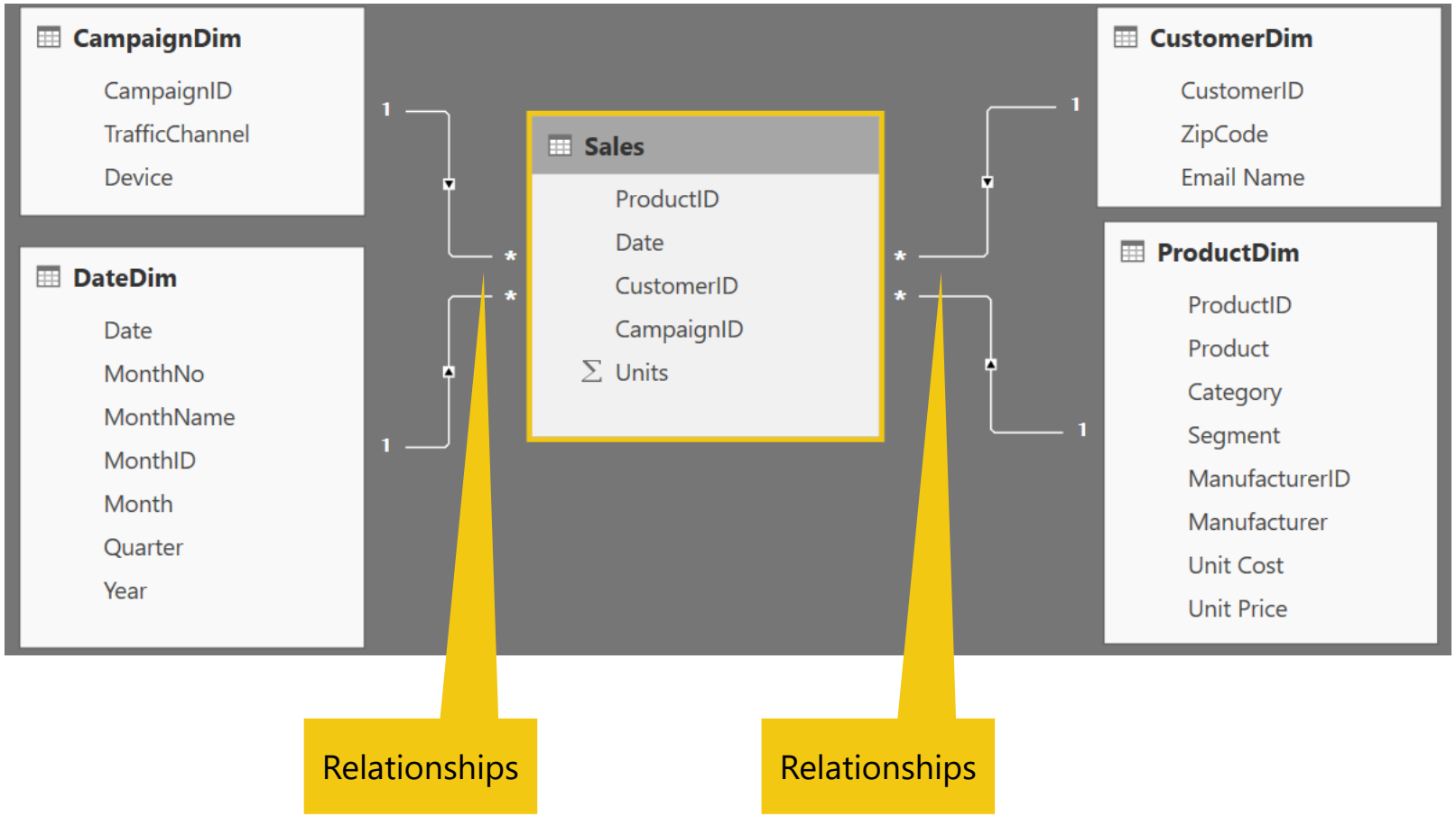
Dim Table

- A Dim (or Dimension) table contains descriptive attributes that define how a fact should roll up.

Examples:
By month,
By Customer, By Geo

Basic Data Modeling

Components of a data model - Relationships



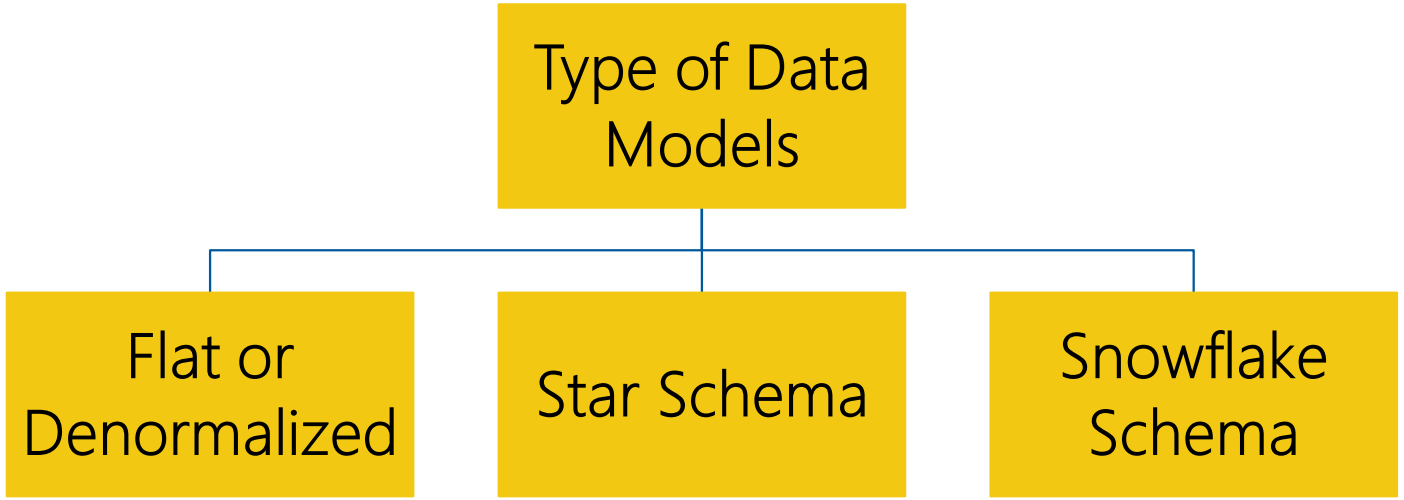
Relationships

- Connection between a 2 tables (usually fact & Dim tables) using columns from each

- 3 kinds of Relationships
  - 1 to Many
  - 1 to 1
  - Many to Many
    (with a bridge table)

Basic Data Modeling

Data Model Brings Facts and Dimensions Together



Note: This is not an exhaustive list, but are the most common model types used by Power BI.

Basic Data Modeling

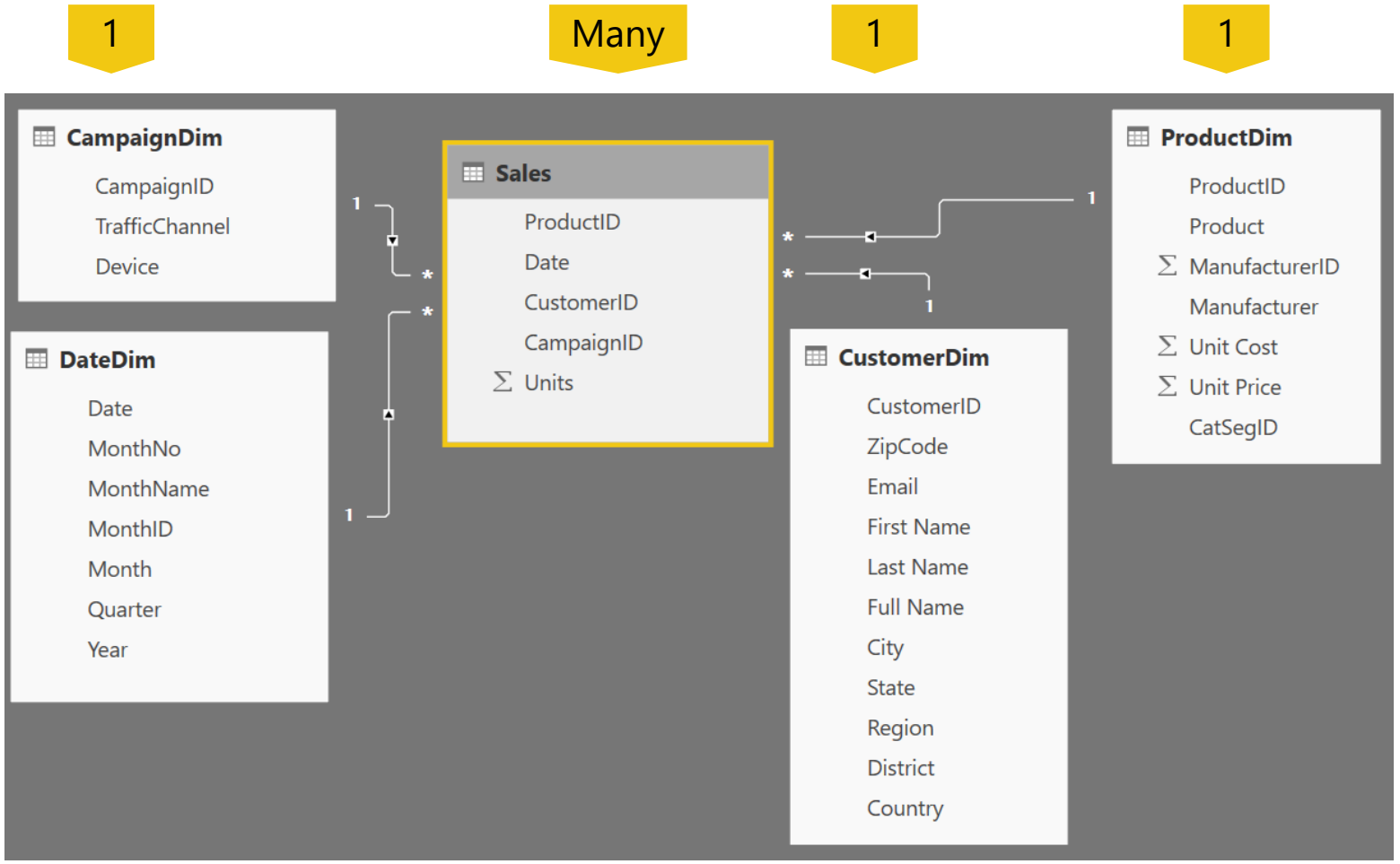
Flat or Denormalized schema

	ProductID	Product	Date	CustomerID	Email	Last Name	First Name	Full Name	CampaignID	Units	CatSegID
1	676	Maximus UC-41	9/25/2011	70283	Farrah.Kent@xyza.com	Kent	Farrah	Farrah Kent	22	1	10
2	585	Maximus UC-50	3/24/2014	70283	Farrah.Kent@xyza.com	Kent	Farrah	Farrah Kent	15	1	10
3	585	Maximus UC-50	11/30/2014	138334	Martha.Mcclain@xyza.…	Mcclain	Martha	Martha Mcclain	8	1	10
4	585	Maximus UC-50	6/21/2015	27193	Hedda.Mcintosh@xyza.…	Mcintosh	Hedda	Hedda Mcintosh	22	1	10
5	585	Maximus UC-50	1/6/2013	238970	Lunea.Walker@xyza.com	Walker	Lunea	Lunea Walker	21	1	10
6	585	Maximus UC-50	3/22/2013	182241	Upton.Page@xyza.com	Page	Upton	Upton Page	17	1	10
7	449	Maximus UM-54	9/25/2011	195385	Drake.Wells@xyza.com	Wells	Drake	Drake Wells	22	1	4
8	449	Maximus UM-54	9/30/2014	168009	Wallace.Bender@xyza.…	Bender	Wallace	Wallace Bender	17	1	4
9	449	Maximus UM-54	8/12/2014	110391	Astra.Erickson@xyza.…	Erickson	Astra	Astra Erickson	20	1	4
10	449	Maximus UM-54	4/16/2014	49327	Echo.Bradley@xyza.com	Bradley	Echo	Echo Bradley	7	1	4
11	449	Maximus UM-54	2/28/2013	65952	Yoko.Gross@xyza.com	Gross	Yoko	Yoko Gross	17	1	4
12	449	Maximus UM-54	6/6/2013	97	Yoshi.Grant@xyza.com	Grant	Yoshi	Yoshi Grant	10	1	4
13	449	Maximus UM-54	5/14/2013	56757	Brian.Carrillo@xyza.…	Carrillo	Brian	Brian Carrillo	10	1	4
14	449	Maximus UM-54	4/9/2015	248715	Mark.Hewitt@xyza.com	Hewitt	Mark	Mark Hewitt	19	1	4
15	449	Maximus UM-54	4/28/2013	248715	Mark.Hewitt@xyza.com	Hewitt	Mark	Mark Hewitt	8	1	4
16	449	Maximus UM-54	3/28/2014	240831	Oscar.Avila@xyza.com	Avila	Oscar	Oscar Avila	18	1	4
17	449	Maximus UM-54	2/26/2014	201004	Duncan.Mcintosh@xyza…	Mcintosh	Duncan	Duncan Mcintosh	19	1	4
18	615	Maximus UC-80	5/14/2012	212645	Jacob.Santiago@xyza.…	Santiago	Jacob	Jacob Santiago	22	1	10
19	615	Maximus UC-80	5/14/2012	70666	Hilary.Collier@xyza.…	Collier	Hilary	Hilary Collier	22	1	10
20	615	Maximus UC-80	5/14/2012	114459	Chester.Mitchell@xyz…	Mitchell	Chester	Chester Mitche…	22	1	10
21	615	Maximus UC-80	5/14/2012	221670	Sage.Yang@xyza.com	Yang	Sage	Sage Yang	22	1	10
22	615	Maximus UC-80	6/3/2012	168009	Wallace.Bender@xyza.…	Bender	Wallace	Wallace Bender	22	1	10
23	615	Maximus UC-80	6/3/2012	154439	Iliana.Dunlap@xyza.c…	Dunlap	Iliana	Iliana Dunlap	22	1	10
24	615	Maximus UC-80	6/4/2012	191391	Joelle.Lee@xyza.com	Lee	Joelle	Joelle Lee	22	1	10

- All attributes for model exist in a single table
- **Highly inefficient**
- Model has extra copies of data > slow performance
- Size of a flat table can blow up really quickly as data model becomes complex

Basic Data Modeling

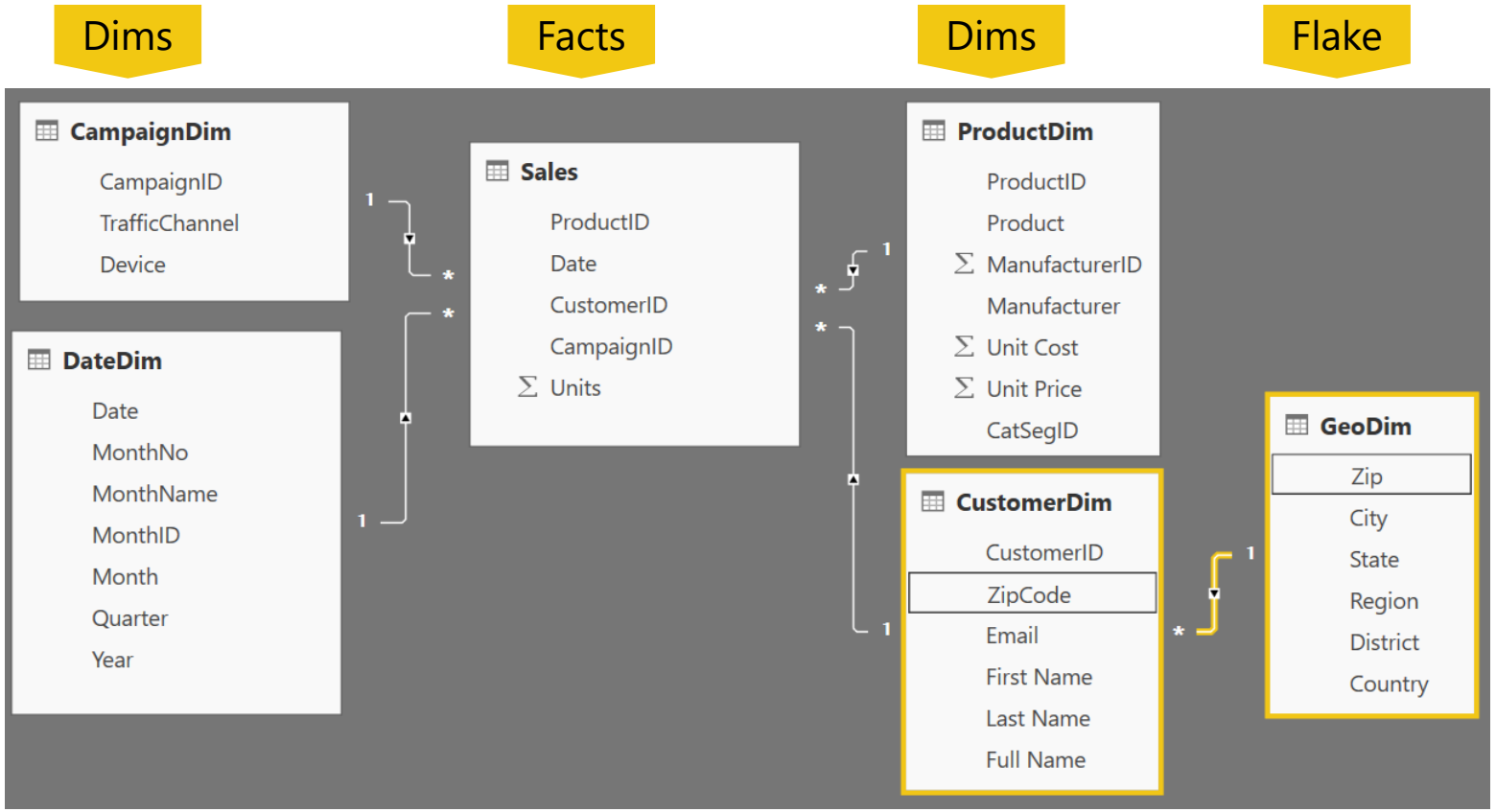
Star Schema



- **Fact** table in the middle
- Surrounded by **Dims**
- Looks like a 'Star'
- Fact table is the "Many" side of the (one to many) relationship

Basic Data Modeling

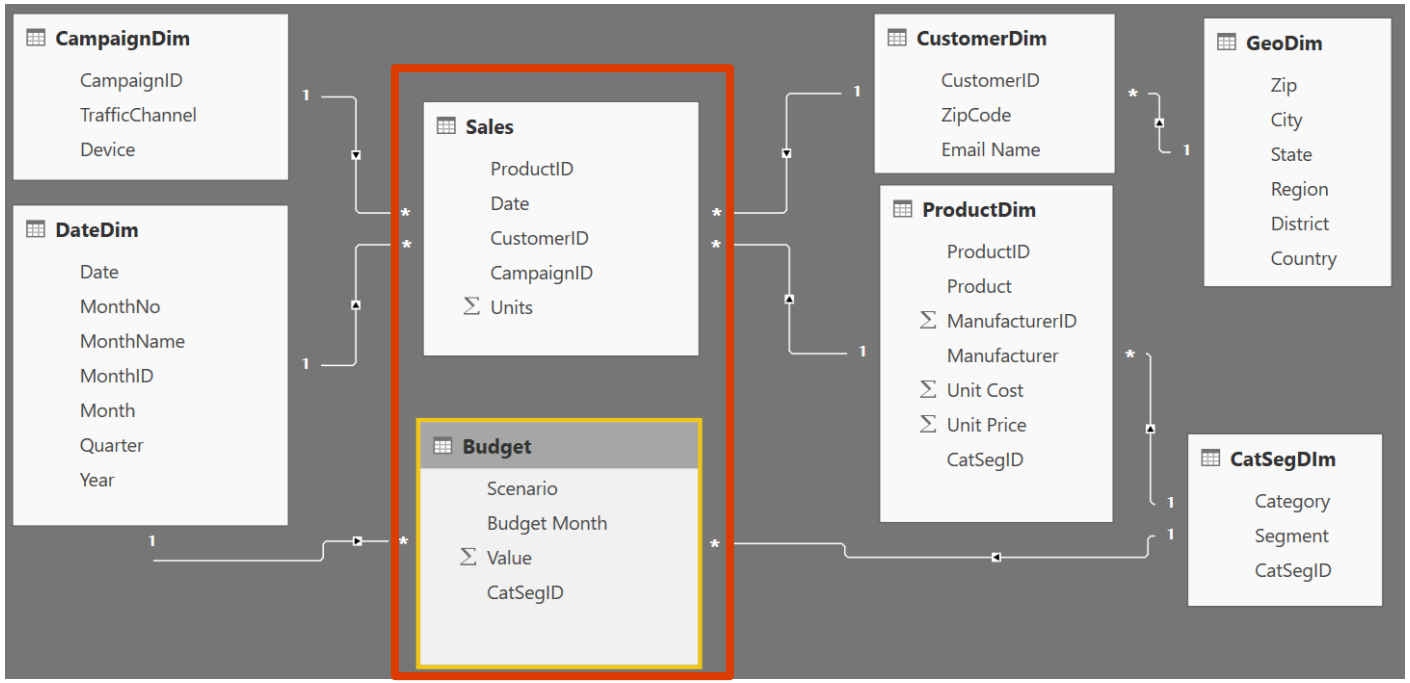
Snowflake Schema



- Center is a Star schema
- **Fact** table in middle
- Surrounded by **Dims**
- Dims "snowflake" off of other Dims
- If you have many, it looks like a 'Snowflake'
- Dim or Fact tables can be the "Many" side of the relationship

Basic Data Modeling

Granularity & Multiple Fact



SalesFact (Daily by Product)

BudgetFact (Monthly by Product Category & Product Segment)

- Grain (**granularity)** measures the level of detail in a table
- Example:
  One row per **order or per Item**
  Daily or Monthly date grain
- If your facts have **very different granularities**, split them into **Multiple Fact** tables & connect them to shared dimensions at the lowest common granularity.

Data Types

Numeric Data Types

- Whole Number
- Decimal Number
- Fixed Decimal Number (Floating point stored as integer)
- Boolean

Date/Time Data Types

- Date – Internally stored as an integer
- Time – Internally stored as a fraction between 0 and 1
- Date Time

Other Data Types

- Text
- **Any – You should never see this in a data model. Bad things can happen!!**

Set your
Data Types
in the
Query Editor

Set your
Data Formats
($ %, etc)
in the Data Model

Pro Tip: Data type is different from data format

Designing good data models

Star Schema – Go to structure for most Data Models

RAM is precious !!!!!

Some Tips and tricks to save RAM and increase speed of model

- If a fact table contains an ID field which is unique for each record, **remove it**
  - Ex. Transaction ID

- **Sort columns** before bringing them into a Power BI data model

- Change all of the ANY data types to an appropriate Data Type

KNOWLEDGE CHECK

1. What allow tables in the data model to “talk” to one another?
2. What type of schema has all attributes for model in a single table?
3. When would you use multiple Fact tables

Lab 01

Open Data Shaping Lab file and follow the instructions for Lab 01 - Import multiple tables from a single source

MODULE 2:
Getting Started with M
(Power Query Language)

MODULE OBJECTIVES

Objectives

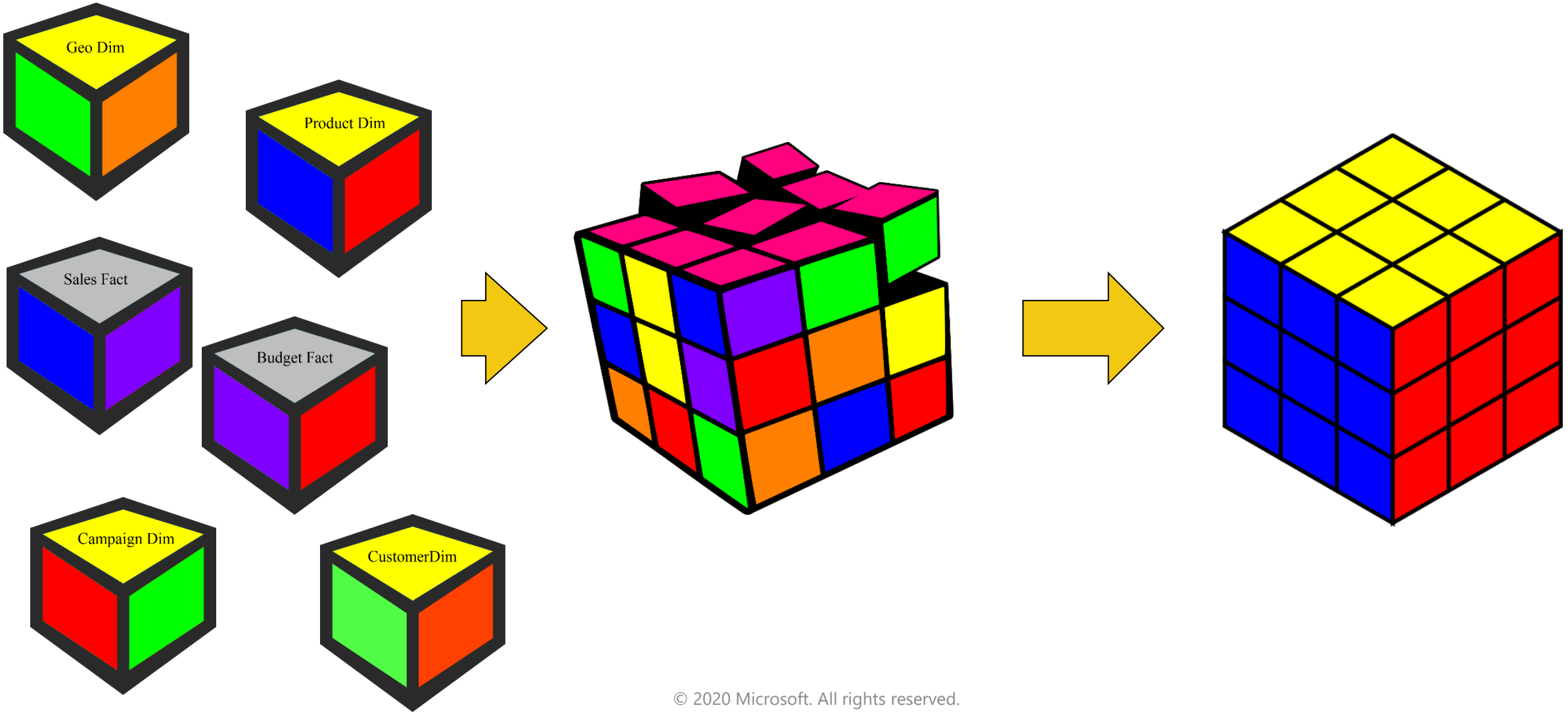
- Understand the basics syntax of the M Language
- Understand function & keyword structure
- Understand how to connect to data
- Understand the basic structure of the Advanced Editor

Agenda

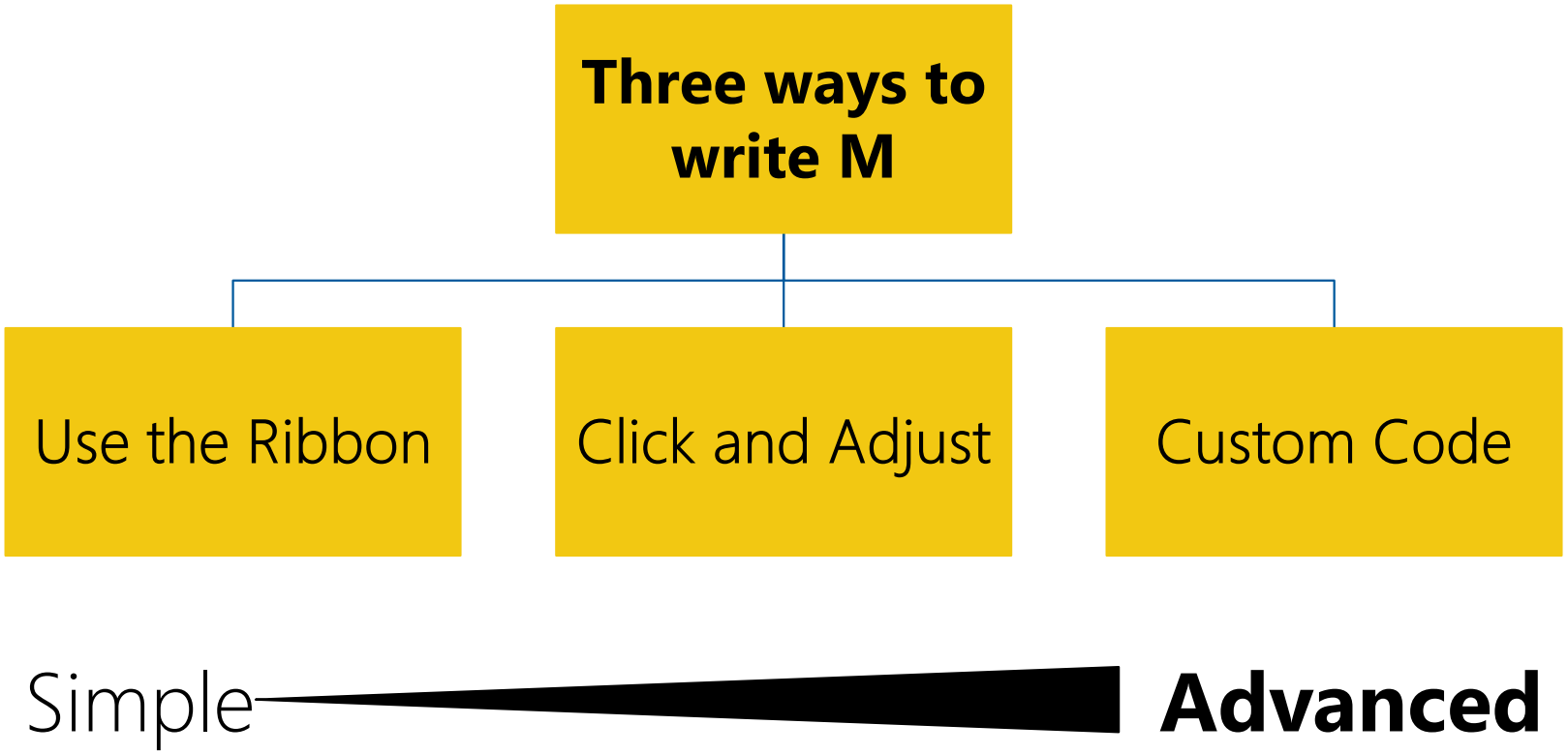
- Power BI M Language Overview
- Basic Data Import
- Introduction to Text functions
- Create and Apply Degenerate Dimensions

Why M?

M has amazing capabilities to transform data to optimize it for the data model



How do I use M?



Key Concepts of M syntax

M is the **key** to data transformation in Power BI

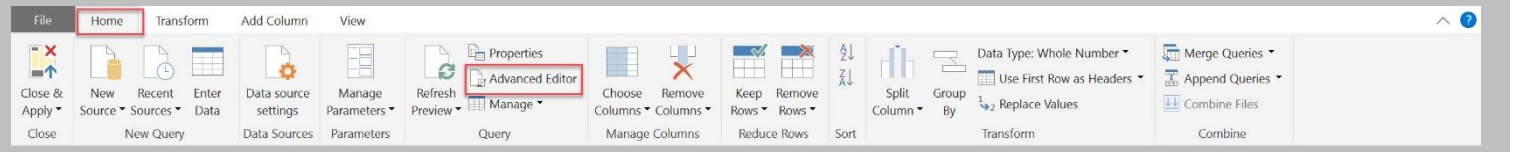
Many transforms are just a click away on the **Ribbons**

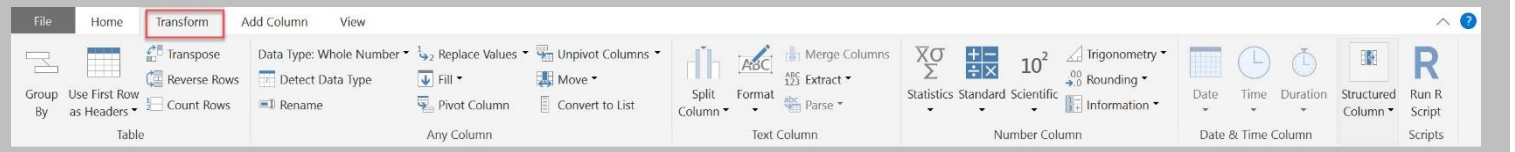
Turn on **Formula Bar** to see M syntax generated by clicks

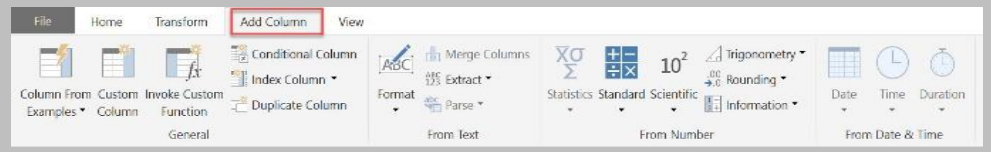
View full syntax in **Advanced Editor**

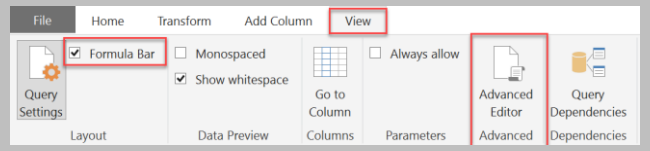
M is Column Based

- Many Transformations occur by adding columns









M is **CaSe SeNsItIvE!**

Key Concepts of M syntax

M Formulas

Foundation of transformations

- **Syntax:**
  Object.Action()
- **Objects:**
  Table, List, Record, Text, Number, Date, etc.
- **Actions (Camel Case):**
  ToList, DayOfWeek, ToNumber, FromText, etc.

M helper Words

little helper words – never capitalize these

- let in each
- if then else
- true false error
- and or not
- try otherwise
- as is meta section shared type

Key Punctuation

- [Column] - To reference a Column
- {List} – use {} to indicate a List

M # Key Words

Used to create new artifacts and literals

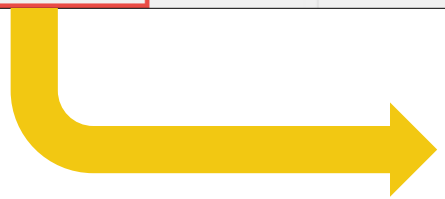
- #date(2016,01,01)
  #date([Year],[Month],[Day])
- #datetime(2016,02,26, 09,15,00)
- #time(09,15,00)
- #datetimezone(2013,02,26, 09,15,00, 09,00)
- #table({"X","Y"},{{0,1},{1,0}})
- #duration(0,1,30,0)
- #binary #infinity #nan #sections #shared

M is **CaSe SeNsItIvE!**

Power Query Formula Language

Text Functions

MyColumn	Text.Start	Text.End	Text.Range (Partial)	Text.Range (Full)	Text.StartsWith (abc)	Text.StartsWith (ABC)	Text.Length	Text.Contains (123)	Text.PostionOf (5)
ABC12345xyz	ABC	5xyz	234	2345xyz	FALSE	TRUE	11	TRUE	7



Position

My Column

M

0	1	2	3	4	5	6	7	8	9	10
A	B	C	1	2	3	4	5	x	y	z

Excel

1	2	3	4	5	6	7	8	9	10	11
A	B	C	1	2	3	4	5	x	y	z

Return "ABC"	Text.Start([MyColumn], 3)	LEFT([MyColumn], 3)
Return "234"	Text.Range([MyColumn], **4**, 3)	MID([MyColumn, **5**, 3)
Find position of "5"	PositionOf([MyColumn], "5") = **7**	SEARCH("5", [MyColumn]) = **8**

Text Functions

MyColumn	Text.Start	Text.End	Text.Range (Partial)	Text.Range (Full)	Text.StartsWith (abc)	Text.StartsWith (ABC)	Text.Length	Text.Contains (123)	Text.PostionOf (5)
ABC12345xyz	ABC	5xyz	234	2345xyz	FALSE	TRUE	11	TRUE	7

M Function Syntax	Excel Syntax	M Output
Text.Start([MyColumn], 3)	LEFT([MyColumn], 3)	ABC
Text.End([MyColumn], 4)	RIGHT([MyColumn], 4)	5xyz
Text.Range([MyColumn], 4, 3) *	MID([MyColumn], 5, 3) **	234
Text.Range([MyColumn], 4) *	MID([MyColumn], 5) **	2345xyz
Text.StartsWith ([MyColumn], "abc")	IF(LEFT([MyColumn], 3)="abc" , TRUE(), FALSE())	FALSE (Excel -> TRUE)
Text.StartsWith ([MyColumn], "ABC")	IF(LEFT([MyColumn], 3)="ABC", TRUE(), FALSE())	TRUE
Text.Length([MyColumn])	LEN([MyColumn])	11
Text.Contains ([MyColumn], "123")	IF(ISNUMBER(SEARCH("123", [MyColumn])),TRUE(), FALSE()) **	TRUE
Text.PositionOf ([MyColumn], "5") *	SEARCH("5", [MyColumn]) **	7 (Excel -> 8)

M is **CaSe SeNsItIvE!**

* M is 0 index based, so Text.Range & Text.PositionOf start counting at 0

** Excel MID & SEARCH are 1 index based – start counting at 1

if … then … else

- Basic Syntax:
  if <test if true> **then** <result if true> else <result if false>
- if's Can be chained or nested
- No parentheses () are *required, except*
- When if test or results have multiple conditions:
  (condition 1 and condition 2)
  (condition 1 or condition 2)
  (condition 1 or (condition 2 and condition 3))
- If multiple if's are used ensure you have the same number of "if", "then", "else"

M is **CaSe SeNsItIvE!**

Simple example (1 each of if, then & else)

```
=if [Column2] = "Key" then [Column2] else "Other"
```

Chained example (3 each of if, then & else)

```
=if Text.EndsWith([Column2],"Key") then [Column2] else
if [Column2] = "TotalCost" then "Cost $" else
if [Column2] = "SalesQuantity" then "Sales #" else null
```

Nested example (3 each of if, then & else)

```
=if Text.EndsWith([Column2],"Key") then
     if [Column2] = "TotalCost" then "Cost $" else
     if [Column2] = "SalesQuantity" then "Sales #" else null
  else "Item #"
```

Multiple conditions example (1 each of if, then & else)

```
=if  (Text.Length([Column2]) = 3 or Text.Length([Column2]) = 2 )
then  true else false
```

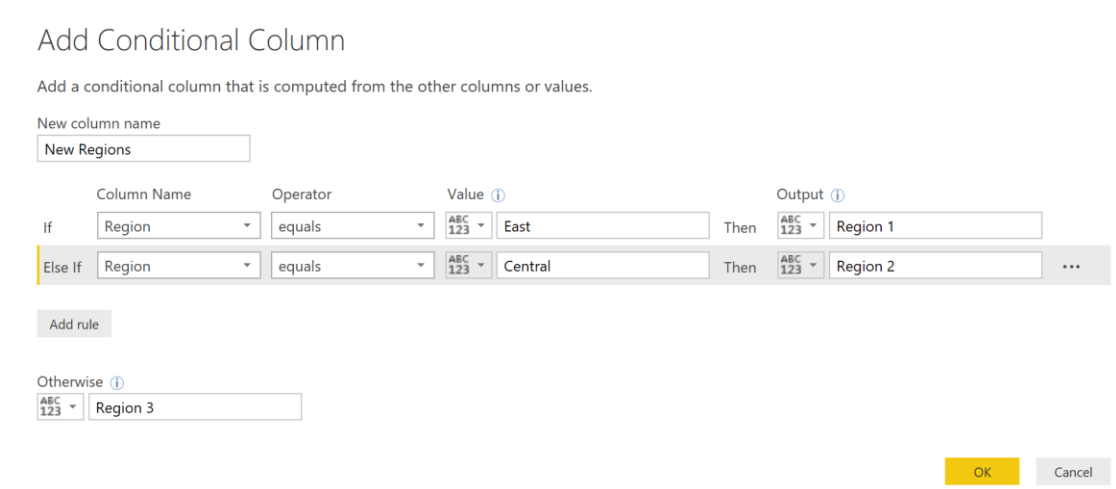
if … then … else using Conditional and Custom

Whether you use Conditional Column or Custom Column, the results are the same

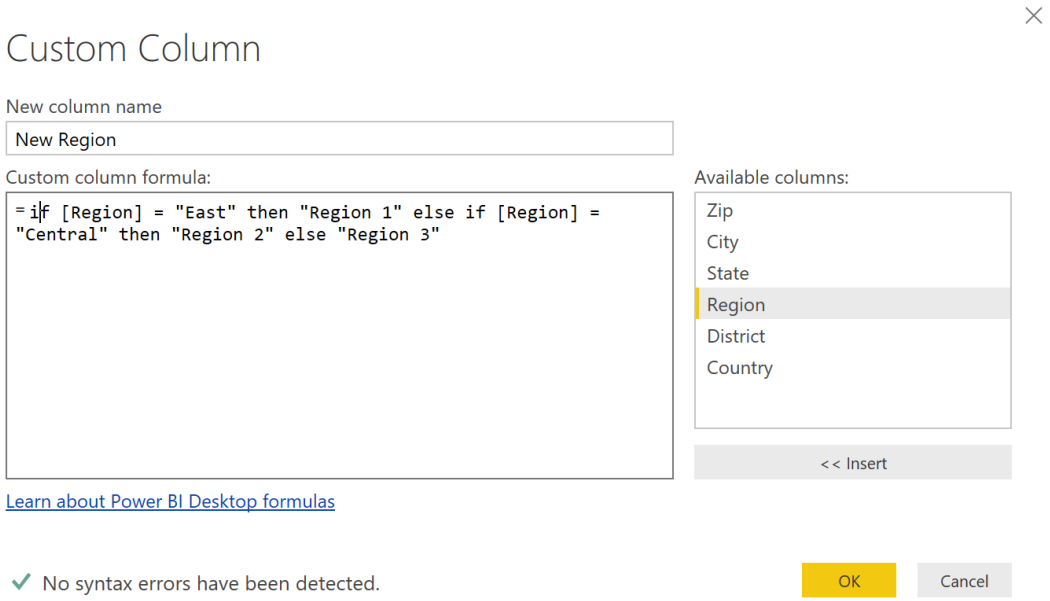
	Zip	City	State	Region	District
1	22654	Star Tannery, VA, USA	VA	East	District #07
2	22655	Stephens City, VA, USA	VA	East	District #07
3	22656	Stephenson, VA, USA	VA	East	District #07
4	22657	Strasburg, VA, USA	VA	East	District #07
5	22660	Toms Brook, VA, USA	VA	East	District #07
6	22663	White Post, VA, USA	VA	East	District #07

1. How it reads in the formula bar when using the Conditional Column format

```
= Table.AddColumn(#"Changed Type", "New Regions", each if [Region] = "East" then "Region 1"
else if [Region] = "Central" then "Region 2" else "Region 3" )
```

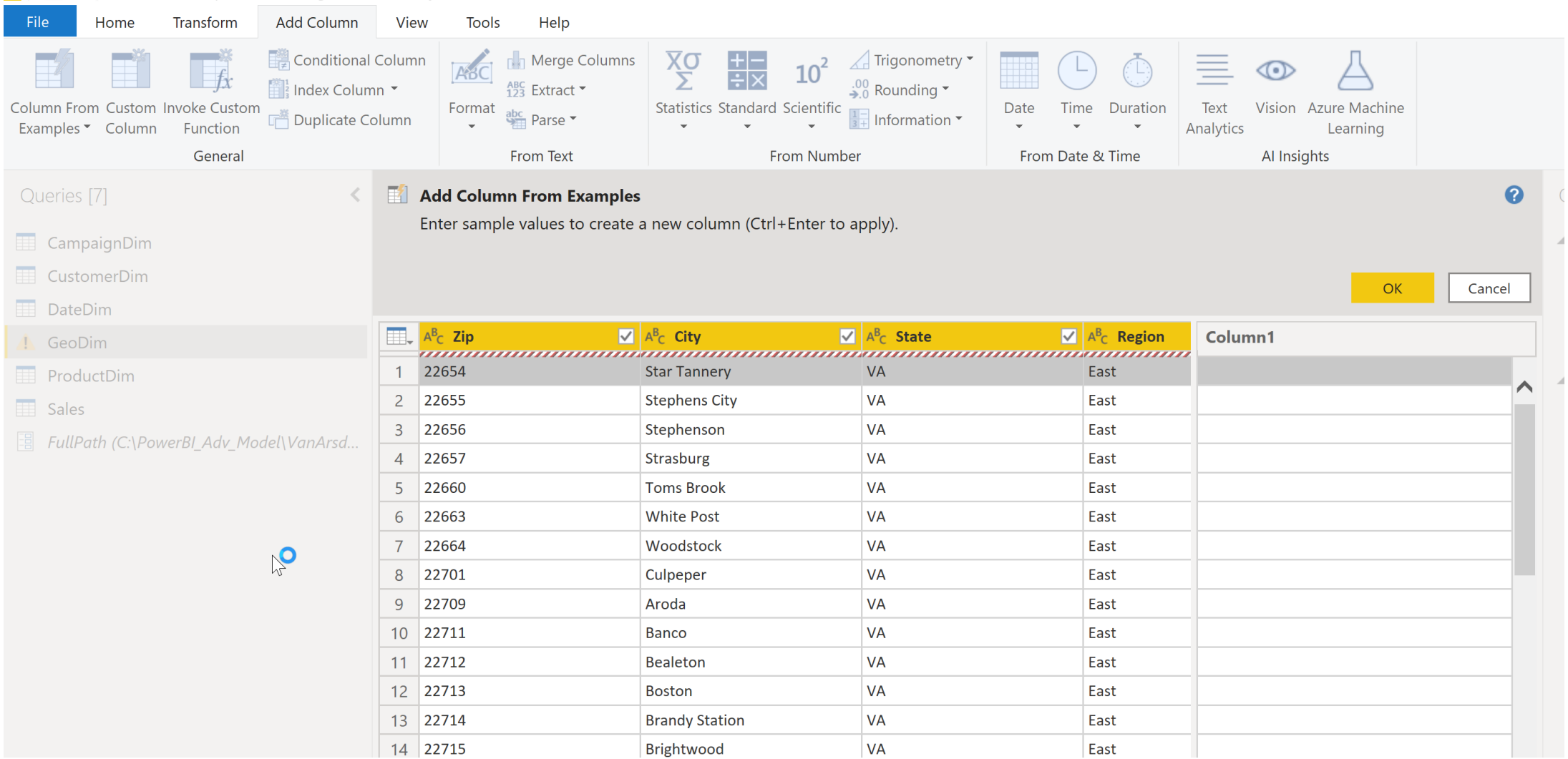


2. When using the Conditional Column format



if … then … else using Columns From Examples

You use Columns From Examples to create if else logic



Module 2 Lab: Import and basic transform

Who are we?

We report on Actual & Budget Revenue; Units, Cost & Gross Profit

- VtB – Actual Variance to Budget
- YoY – Year over Year
- MoM – Current Month vs Prior Month
- One Division percent of Total

VanArsdel

How we describe the data

- By Product Category
- By Product Segment
- By Campaign
- By Customer

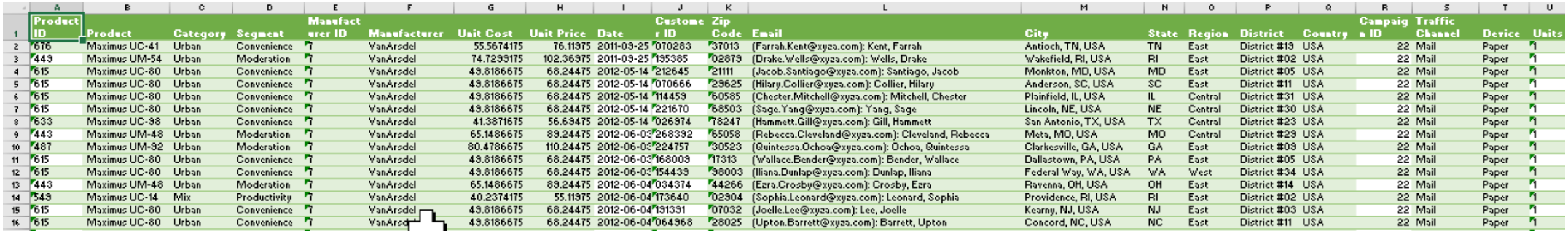
Denormalized to a Data Model

How do you turn a common Excel report into a Data Model

- One Excel sheet with multiple columns
- What am I counting or aggregating (FactTable)

How we describe the data

- By Product Category
- By Product Segment
- By Campaign
- By Customer

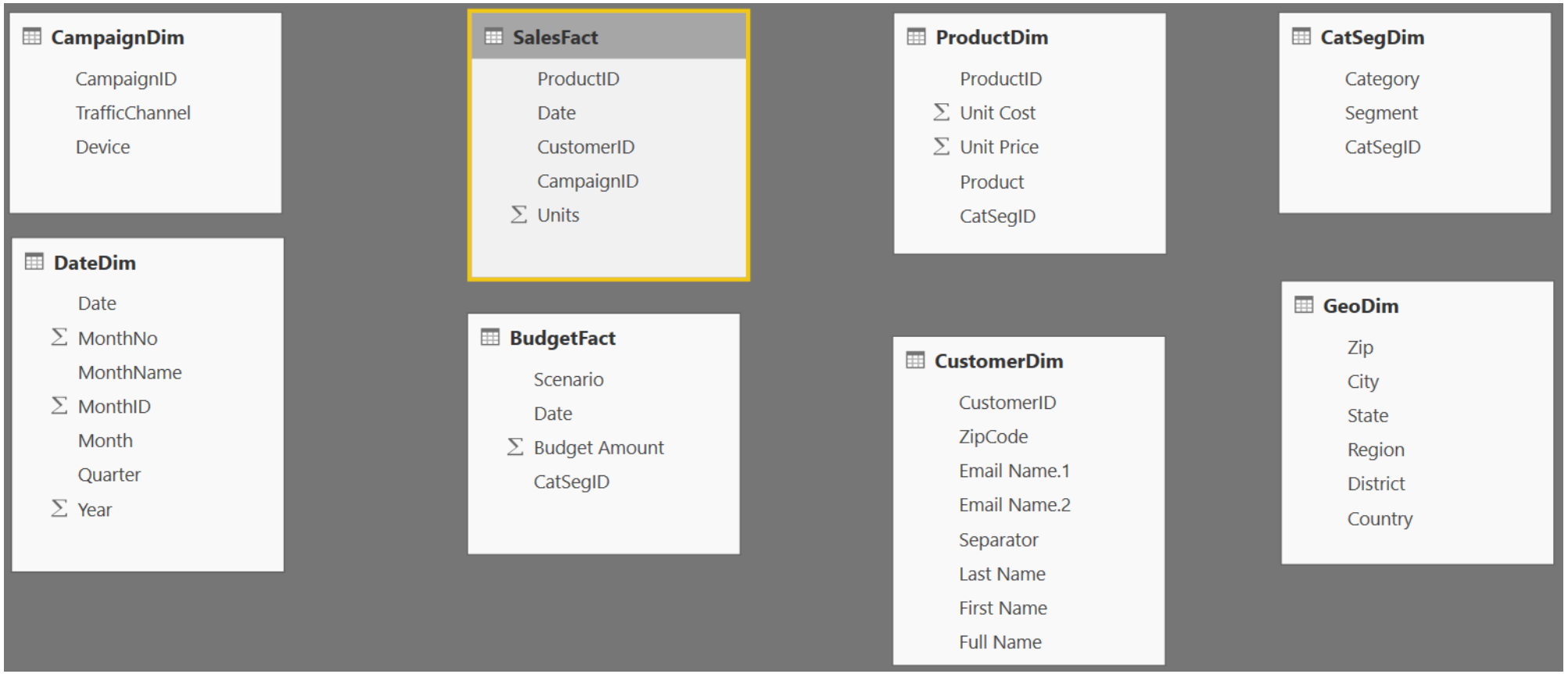


	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1	Product ID	Product	Category	Segment	Manufacturer ID	Manufacturer	Unit Cost	Unit Price	Date	Customer ID	Zip Code	Email	City	State	Region	District	Country	Campaign ID	Traffic Channel	Device	Units
2	676	Maximus UC-41	Urban	Convenience	7	VanArsdel	55.5674175	76.11975	2011-09-25	070283	37013	(Farrah.Kent@xyza.com): Kent, Farrah	Antioch, TN, USA	TN	East	District #19	USA	22	Mail	Paper	1
3	449	Maximus UM-54	Urban	Moderation	7	VanArsdel	74.7299175	102.36975	2011-09-25	195385	02879	(Drake.Wells@xyza.com): Wells, Drake	Wakefield, RI, USA	RI	East	District #02	USA	22	Mail	Paper	1
4	615	Maximus UC-80	Urban	Convenience	7	VanArsdel	49.8186675	68.24475	2012-05-14	212645	21111	(Jacob.Santiago@xyza.com): Santiago, Jacob	Monkton, MD, USA	MD	East	District #05	USA	22	Mail	Paper	1
5	615	Maximus UC-80	Urban	Convenience	7	VanArsdel	49.8186675	68.24475	2012-05-14	070666	29625	(Hilary.Collier@xyza.com): Collier, Hilary	Anderson, SC, USA	SC	East	District #11	USA	22	Mail	Paper	1
6	615	Maximus UC-80	Urban	Convenience	7	VanArsdel	49.8186675	68.24475	2012-05-14	114459	60585	(Chester.Mitchell@xyza.com): Mitchell, Chester	Plainfield, IL, USA	IL	Central	District #31	USA	22	Mail	Paper	1
7	615	Maximus UC-80	Urban	Convenience	7	VanArsdel	49.8186675	68.24475	2012-05-14	221670	68503	(Sage.Yang@xyza.com): Yang, Sage	Lincoln, NE, USA	NE	Central	District #30	USA	22	Mail	Paper	1
8	633	Maximus UC-38	Urban	Convenience	7	VanArsdel	41.3871675	56.69475	2012-05-14	026974	78247	(Hammett.Gill@xyza.com): Gill, Hammett	San Antonio, TX, USA	TX	Central	District #23	USA	22	Mail	Paper	1
9	443	Maximus UM-48	Urban	Moderation	7	VanArsdel	65.1486675	89.24475	2012-06-03	268332	65058	(Rebecca.Cleveland@xyza.com): Cleveland, Rebecca	Meta, MO, USA	MO	Central	District #29	USA	22	Mail	Paper	1
10	487	Maximus UM-32	Urban	Moderation	7	VanArsdel	80.4786675	110.24475	2012-06-03	224757	30523	(Quintessa.Ochoa@xyza.com): Ochoa, Quintessa	Clarkesville, GA, USA	GA	East	District #09	USA	22	Mail	Paper	1
11	615	Maximus UC-80	Urban	Convenience	7	VanArsdel	49.8186675	68.24475	2012-06-03	168003	17313	(Wallace.Bender@xyza.com): Bender, Wallace	Dallastown, PA, USA	PA	East	District #05	USA	22	Mail	Paper	1
12	615	Maximus UC-80	Urban	Convenience	7	VanArsdel	49.8186675	68.24475	2012-06-03	154439	98003	(Iliana.Dunlap@xyza.com): Dunlap, Iliana	Federal Way, WA, USA	WA	West	District #34	USA	22	Mail	Paper	1
13	443	Maximus UM-48	Urban	Moderation	7	VanArsdel	65.1486675	89.24475	2012-06-04	034374	44266	(Ezra.Crosby@xyza.com): Crosby, Ezra	Ravenna, OH, USA	OH	East	District #14	USA	22	Mail	Paper	1
14	549	Maximus UC-14	Mix	Productivity	7	VanArsdel	40.2374175	55.11975	2012-06-04	173640	02904	(Sophia.Leonard@xyza.com): Leonard, Sophia	Providence, RI, USA	RI	East	District #02	USA	22	Mail	Paper	1
15	615	Maximus UC-80	Urban	Convenience	7	VanArsdel	49.8186675	68.24475	2012-06-04	131391	07032	(Joelle.Lee@xyza.com): Lee, Joelle	Kearny, NJ, USA	NJ	East	District #03	USA	22	Mail	Paper	1
16	615	Maximus UC-80	Urban	Convenience	7	VanArsd	49.8186675	68.24475	2012-06-04	064968	28025	(Upton.Barrett@xyza.com): Barrett, Upton	Concord, NC, USA	NC	East	District #11	USA	22	Mail	Paper	1

Import Data Walkthrough

You have access to the data for your model, but it is not in the right "Shape"

- Start with basic Get Data queries to bring data in to the Query Editor
- Use a combination of the UI and M to transform data into a streamlined data model



Module 2 Lab: Import Multiple Tables from a Single Source

If the Source mirrors the Table required by the Model, import multiple tables at once

CampaignDim

	CampaignID	TrafficChannel	Device
1	1	Organic Search	Desktop
2	2	Organic Search	Mobile
3	3	Organic Search	Tablet

GeoDim

	Zip	City	State	Region	District
1	22654	Star Tannery, VA, USA	VA	East	District #07
2	22655	Stephens City, VA, USA	VA	East	District #07
3	22656	Stephenson, VA, USA	VA	East	District #07

ProductDim

	ProductID	CatSegID	Product	Unit Price
1	392	1	Maximus RP-01	51.05625
2	393	1	Maximus RP-02	51.05625
3	394	2	Maximus RS-01	164.05725

DateDim

	Date	MonthNo	MonthName	MonthID	Month	Quarter
1	1/1/2011	1	Jan	201101	1/11/2020	Q1
2	1/2/2011	1	Jan	201101	1/11/2020	Q1
3	1/3/2011	1	Jan	201101	1/11/2020	Q1

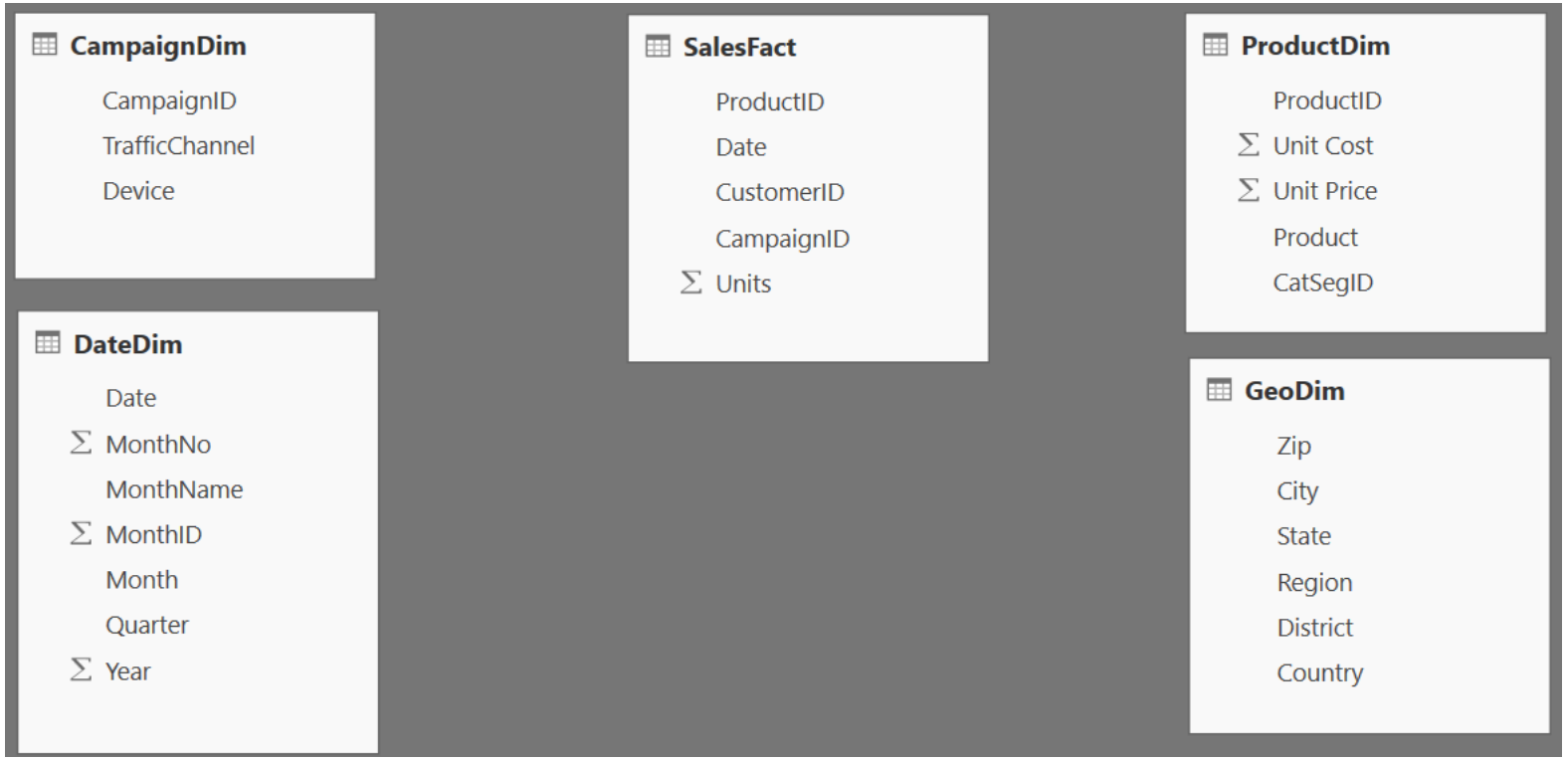
Sales

	ProductID	Date	CustomerID	CampaignID	Units
1	676	9/25/2011	70283	22	1
2	449	9/25/2011	195385	22	1
3	615	5/14/2012	212645	22	1

Module 2 Lab Exercise

This is the current state of our data model

- Relationships will be “drawn” when the model is loaded



Module 2 Lab: Create CatSegDim (Category Segment)

Extract unique combinations of Categories & Segments from the Product Dimension

ProductID	Product	Category	Segment
392	Maximus RP-01	Rural	Productivity
393	Maximus RP-02	Rural	Productivity
394	Maximus RS-01	Rural	Select

Add Index

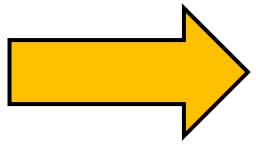
CatSegID	Category	Segment
1	Rural	Productivity
2	Rural	Select
3	Accessory	Accessory
4	Urban	Moderation
5	Urban	Regular
6	Urban	Extreme
7	Mix	All Season
8	Mix	Productivity
9	Youth	Youth
10	Urban	Convenience

CatSegDim is a **Degenerate Dimension**
It is extracted from the data in the fact table.

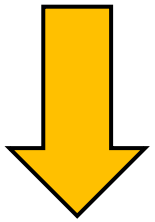
Module 2 Lab: Update Product Dim

Apply new CatSegID field to the Product Dimension

ProductID	Product	Category	Segment	ManufacturerID
392	Maximus RP-01	Rural	Productivity	7
393	Maximus RP-02	Rural	Productivity	7
394	Maximus RS-01	Rural	Select	7
396	Maximus UM-01	Accessory	Accessory	7
397	Maximus UM-02	Accessory	Accessory	7
398	Maximus UM-03	Accessory	Accessory	7
400	Maximus UM-05	Accessory	Accessory	7
402	Maximus UM-07	Accessory	Accessory	7



Replace with Index

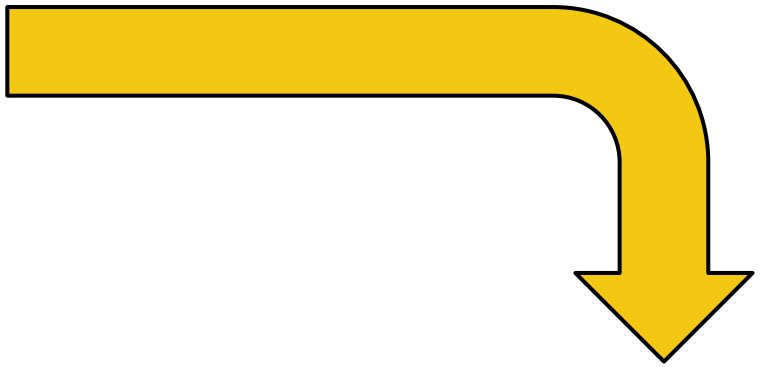


ProductID	Product	CatSegID	Unit Cost	Unit Price
392	Maximus RP-01	1	37.2710625	51.0562
393	Maximus RP-02	1	37.2710625	51.0562
394	Maximus RS-01	2	119.7617925	164.0572
396	Maximus UM-01	3	66.2830875	90.7987
397	Maximus UM-02	3	109.2224175	149.6197
398	Maximus UM-03	3	114.9711675	157.4947
400	Maximus UM-05	3	61.6840875	84.4987
402	Maximus UM-07	3	74.7299175	102.3697
403	Maximus UM-08	3	83.1154275	113.8567
404	Maximus UM-09	3	136.0499175	186.3697

Module 2 Lab: Create Customer Dimension

Using a *garbage* column to fill out CustomerDim

CustomerID	ZipCode	Email
1	90250	Meghan.Alexander@xyza.com
2	90250	Leah.Kemp@xyza.com
3	90250	Tamekah.Stevens@xyza.com
4	90250	Dexter.Haney@xyza.com
5	90250	Jonah.Moon@xyza.com
6	90250	Brock.Burnett@xyza.com
7	90250	Lamar.Daugherty@xyza.com
8	90250	Dorian.Turner@xyza.com
9	90250	Olympia.Rodriguez@xyza.com
10	90250	Colby.Snow@xyza.com



Key Formulas:

Text.PositionOf()

Text.Start()

Text.Range()

Email	First Name	Last Name	Full Name
Meghan.Alexander@xyza.com	Alexander	Meghan	Alexander Meghan
Leah.Kemp@xyza.com	Kemp	Leah	Kemp Leah
Tamekah.Stevens@xyza.com	Stevens	Tamekah	Stevens Tamekah
Dexter.Haney@xyza.com	Haney	Dexter	Haney Dexter
Jonah.Moon@xyza.com	Moon	Jonah	Moon Jonah
Brock.Burnett@xyza.com	Burnett	Brock	Burnett Brock
Lamar.Daugherty@xyza.com	Daugherty	Lamar	Daugherty Lamar

KNOWLEDGE CHECK

1. You cannot import multiple tables from a single source (T/F)?
2. Why would we need a snowflake dimension?
3. What syntax would you use to find out if a column has “Abc” as its first three character?
4. Or Last three?

Module 2 Lab: Advanced Editor

CustomerDim

```
let
    Source = Actuals_Path,
    CustomerDim_Table = Source{[Item="CustomerDim",Kind="Table"]}[Data],
    #"Changed Type" = Table.TransformColumnTypes(CustomerDim_Table,{{"CustomerID", Int64.Type}, {"ZipCode", type text}, {"Email Name", type text}}),
    #"Split Column by Delimiter" = Table.SplitColumn(#"Changed Type","Email Name",Splitter.SplitTextByDelimiter(": ", QuoteStyle.Csv),{"Email Name.1",
    #"Changed Type1" = Table.TransformColumnTypes(#"Split Column by Delimiter",{{"Email Name.1", type text}, {"Email Name.2", type text}}),
    #"Replaced (" = Table.ReplaceValue(#"Changed Type1","(","",Replacer.ReplaceText,{"Email Name.1"}),
    #"Replaced )" = Table.ReplaceValue(#"Replaced (",")","",Replacer.ReplaceText,{"Email Name.1"}),
    #"Added Separator" = Table.AddColumn(#"Replaced )", "Separator", each Text.PositionOf([Email Name.2], ",")),
    #"Added Last Name" = Table.AddColumn(#"Added Separator", "Last Name", each Text.Start([Email Name.2],[Separator])),
    #"Added First Name" = Table.AddColumn(#"Added Last Name", "First Name", each Text.Range([Email Name.2], [Separator]+2 )),
    // First Name had a trailing Space - removed it
    #"Trimmed First Name" = Table.TransformColumns(#"Added First Name",{{"First Name", Text.Trim}}),
    #"Full Name" = Table.AddColumn(#"Trimmed First Name", "Full Name", each [First Name] & " " & [Last Name]),
    #"Removed Columns" = Table.RemoveColumns(#"Full Name",{"Separator"}),
    #"Changed Type2" = Table.TransformColumnTypes(#"Removed Columns",{{"Last Name", type text}, {"First Name", type text}, {"Full Name", type text}})
in
    #"Changed Type2"
```

Query Settings

PROPERTIES

Name

CustomerDim

All Properties

APPLIED STEPS

- Source
- Navigation
- Changed Type
- Split Column by Delimiter
- Changed Type1
- Replaced (
- Replaced)
- Added Separator
- Added Last Name
- Added First Name
- Trimmed First Name
- Full Name
- Removed Columns
- Changed Type2

Lab 02

Open Data Shaping Lab file and follow the instructions for Lab 02 – exercises A – C

a. Create new dimension - Create CatSegDim

b. Update Product Dimension

c. Create Customer Dimension

Module 3: Advanced M Transformations

MODULE OBJECTIVES

Objectives

- Understand M transformations
- Understand Merge and other combining queries
- Understand how to use multiple queries in an advanced transformation
- Using Data Profiling

Agenda

- Power BI M Key Transformations
- Power BI M Merge Types
- Create BudgetFact using multi-query approach

Key Transformations

Transpose

Transpose

Convert Rows to Columns and Columns to Rows

- **Syntax:**
  Table.Transpose(Previous)
- Column headers are lost
- To transpose column headers, demote them to data prior to transpose

	Color	Letter	Value
1	Red	A	1
2	Red	B	2
3	Blue	C	3
4	Blue	D	4
5	Green	E	5

	Column1	Column2	Column3	Column4	Column5
1	Red	Red	Blue	Blue	Green
2	A	B	C	D	E
3	1	2	3	4	5

Pivot Column

Pivot Column

Convert Selected column of values into Column Headers

- **Syntax:**

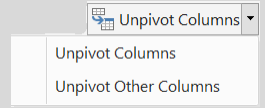
Table.Pivot(Previous,List.Distinct(Previous[Color]), "Color", "Value", List.Sum)

- The values in [Color] column are converted to headers
- The values in the [Value] column are filled in where applicable

	Color	Letter	Value
1	Red	A	1
2	Red	B	2
3	Blue	C	3
4	Blue	D	4
5	Green	E	5

	Letter	Red	Blue	Green
1	A	1	null	null
2	B	2	null	null
3	C	null	3	null
4	D	null	4	null
5	E	null	null	5

Unpivot Columns



Convert Selected Column Headers into a column of values

- **Syntax:**
- Table.UnpivotOtherColumns(Previous, {"Letter"}, "Attribute", "Value")
- The unpivoted column headers become the column [Attribute]
- The unpivoted values become the column [Value]

	Letter	Red	Blue	Green
1	A	1	null	null
2	B	2	null	null
3	C	null	3	null
4	D	null	4	null
5	E	null	null	5

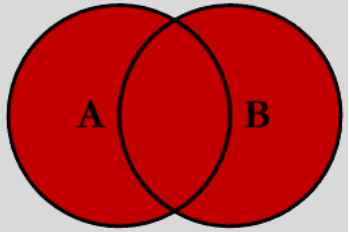
	Letter	Attribute	Value
1	A	Red	1
2	B	Red	2
3	C	Blue	3
4	D	Blue	4
5	E	Green	5

Note: Previous => Name of the previous step in the query

Join Kinds – Merge types

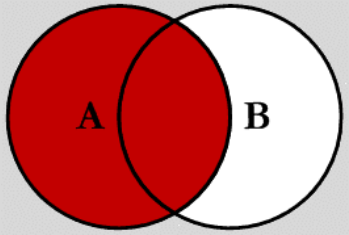
Full Outer (JoinKind.FullOuter)

Shows **all of the rows from both queries.** Like SQL "Union All"

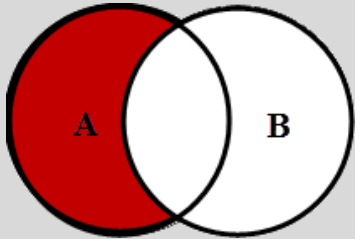


Left Outer (JoinKind.LeftOuter)
Default for Merge

Shows all rows in 1st query (**A**) and "paints" attributes from second query (**B**) based on matches in joined column(s).

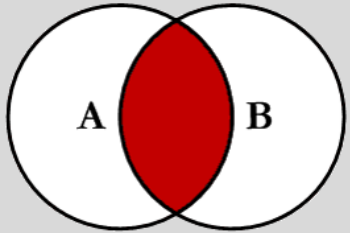


Left Anti (JoinKind.LeftAnti)
Shows ONLY rows in 1st (**A**) query where there is NO MATCH to 2nd query (**B**) based on joined column(s)

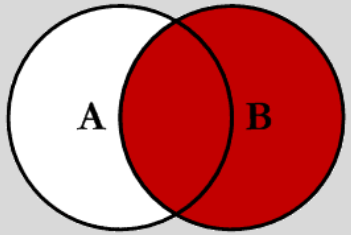


Inner (JoinKind.Inner)

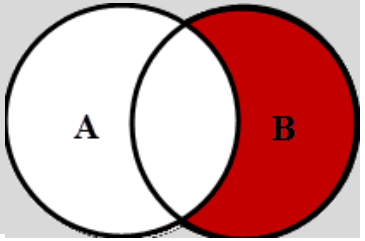
Limits the rows to ONLY those which Match on joined column(s)



Right Outer (JoinKind.RightOuter)
Shows all rows in 2nd (**B**) query and "paints" attributes from 1st query (**A**) based on matches in joined column(s)



Right Anti (JoinKind.RightAnti)
Shows ONLY rows in 2nd (**B**) query where there is NO MATCH to 1st query (**A**) based on joined column(s)



Note: One or Multiple columns can be used to create the join

Join Kinds – Other Combining Queries

Merge

Combines attributes (columns) of two queries based on a joined columns(s)

- Column(s) used for the join must be in both queries (can be one or more)
- Both queries rerun with each execution
- In Database terms, this is a JOIN

Append

"Stack" records of two (or more) queries

- Both (all) queries rerun with each execution
- Like column headers must be named the same
- Non-matching columns will be added to the right as extra colunns

Duplicate

Copies a query at a point in time (Save As)

- Duplicate query can be modified independently without altering the original query
- Executing the 2nd query (duplicate) does NOT execute the 1st (original) query

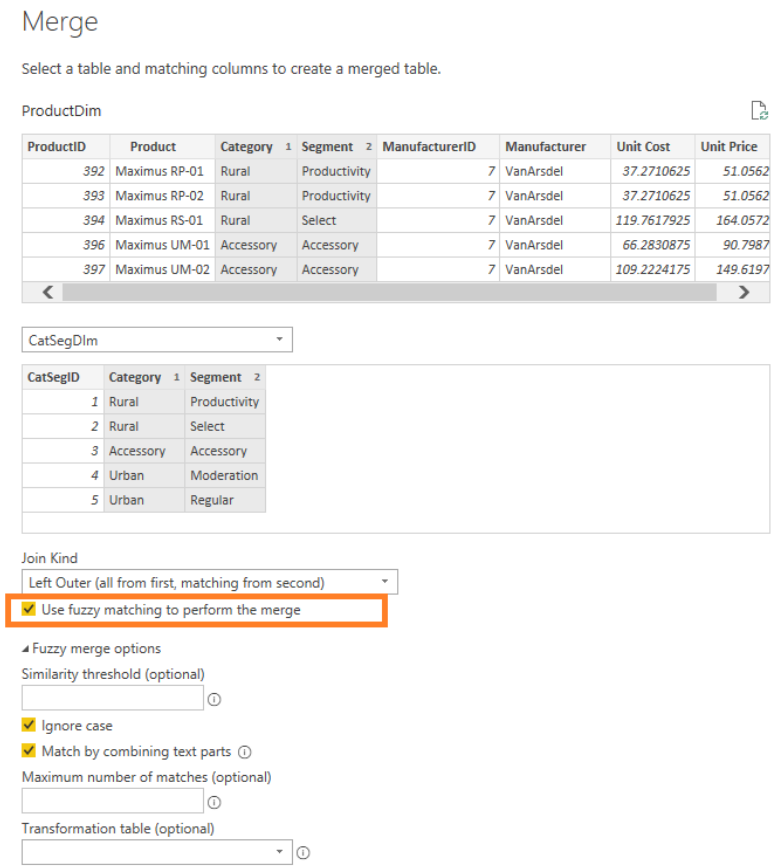
Reference

Uses the output of one query as the INPUT in another query. (Link in a chain)

- All subsequent changes made to 1st query affect the 2nd during refresh
- When the 2nd query executes, it re-executes all the steps in the 1st query

Fuzzy Merge

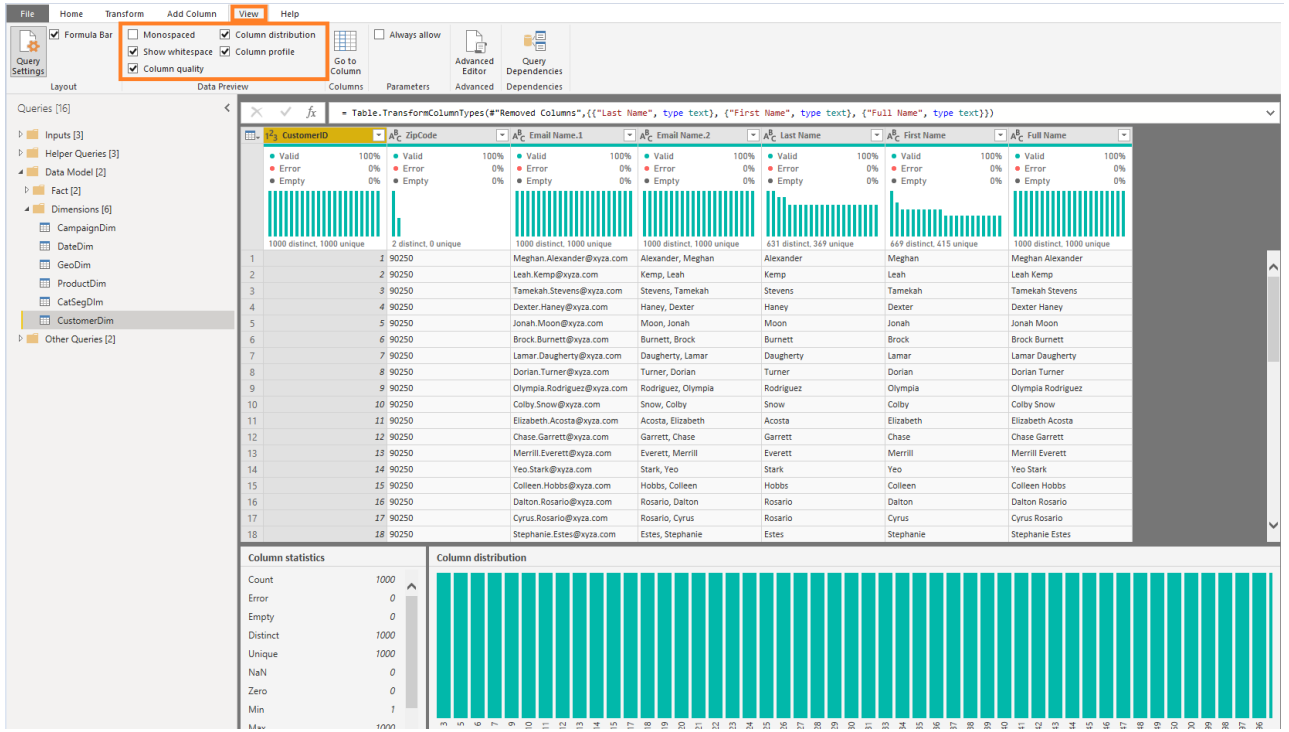
Fuzzy Merge allows you to apply Fuzzy Matching algorithms when comparing columns and try to find matches across tables being merged.



Data Profiling

Data Profiling allows you to easily and quickly understand data distribution

- **Column statistics** – # of errors, empty, valid, duplicated and unique values. Value distribution measures such as Min/Max/Average/Median, etc.
- **Column distribution –** Larger size version of the inline value distribution histograms, also including the ability to Keep or Remove values, which will generate the corresponding Filter Rows step in your query (“Equals” / “Does Not Equal” filters).

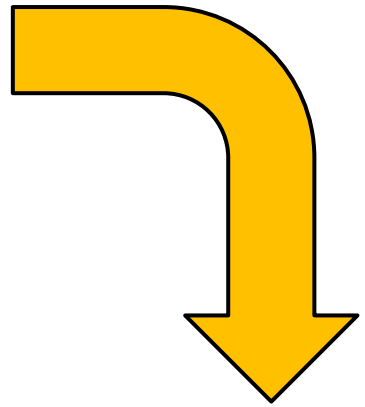


Module 3 Lab: Complex Transformation

Module 3 Lab: Create Budget Fact

Objective: Transform a "wide" Budget file with three extra rows on top and a three row header into a usable BudgetFact table

Column1	Column2	Column3	Column4	Column5
Budget Spreadsheet for VanArsdel				
		Forecast	Forecast	Forecast
		2016	2016	2016
Category	Segment	Dec	Nov	Oct
Accessory	Accessory	44190.57888	50598.81566	54740.5709
Mix	All Season	11442.14474	14120.78693	18109.64804
Mix	Productivity	19538.89812	17597.55926	22835.18396
Rural	Select	311.708775	172.2601125	662.79129
Urban	Convenience	120710.4406	129923.2814	169468.7696
Urban	Extreme	20868.84072	46971.33037	70793.02886
Urban	Moderation	251155.7122	322984.2215	362385.6466
Urban	Regular	689.7969225	427.4372025	2989.28376
Youth	Youth	3931.03074	2891.005425	5397.748965



CatSegID	Scenario	Date	Budget Amount
3	Forecast	12/1/2016	44,190.58
3	Forecast	11/1/2016	50,598.82
3	Forecast	10/1/2016	54,740.57
3	Forecast	9/1/2016	64,442.91
3	Forecast	8/1/2016	98,285.91
3	Forecast	7/1/2016	61,545.41

Module 3 Lab: Create Budget Fact

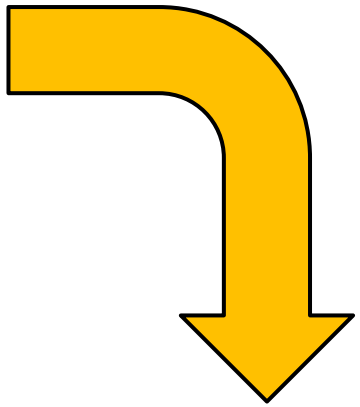
Import CSV, remove blank rows, and rename query to BudgetFact_Data

Column1	Column2	Column3	Column4	Column5
		Forecast	Forecast	Forecast
		2016	2016	2016
Category	Segment	Dec	Nov	Oct
Accessory	Accessory	44190.57888	50598.81566	54740.5709
Mix	All Season	11442.14474	14120.78693	18109.64804
Mix	Productivity	19538.89812	17597.55926	22835.18396
Rural	Select	311.708775	172.2601125	662.79129
Urban	Convenience	120710.4406	129923.2814	169468.7696
Urban	Extreme	20868.84072	46971.33037	70793.02886
Urban	Moderation	251155.7122	322984.2215	362385.6466
Urban	Regular	689.7969225	427.4372025	2989.28376
Youth	Youth	3931.03074	2891.005425	5397.748965

Module 3 Lab: Create Budget Fact

Create a Duplicate, and Extract just the Header Rows into a separate query

Column1	Column2	Column3	Column4	Column5
		Forecast	Forecast	Forecast
		2016	2016	2016
Category	Segment	Dec	Nov	Oct
Accessory	Accessory	44190.57888	50598.81566	54740.5709
Mix	All Season	11442.14474	14120.78693	18109.64804
Mix	Productivity	19538.89812	17597.55926	22835.18396
Rural	Select	311.708775	172.2601125	662.79129
Urban	Convenience	120710.4406	129923.2814	169468.7696
Urban	Extreme	20868.84072	46971.33037	70793.02886
Urban	Moderation	251155.7122	322984.2215	362385.6466
Urban	Regular	689.7969225	427.4372025	2989.28376
Youth	Youth	3931.03074	2891.005425	5397.748965



Column1	Column2	Column3	Column4	Column5
		Forecast	Forecast	Forecast
		2016	2016	2016
Category	Segment	Dec	Nov	Oct

Module 3 Lab: Create Budget Fact

Transpose headers, combine into a single column, and transpose back

Column1	Column2	Column3	Column4	Column5
		Forecast	Forecast	Forecast
		2016	2016	2016
Category	Segment	Dec	Nov	Oct

Column1	Column2	Column3
		Category
		Segment
Forecast	2016	Dec
Forecast	2016	Nov
Forecast	2016	Oct
Forecast	2016	Sep
Forecast	2016	Aug
Forecast	2016	Jul
Forecast	2016	Jun
Forecast	2016	May
Forecast	2016	Apr
Forecast	2016	Mar
Forecast	2016	Feb
Forecast	2016	Jan

Fully Combined Header
Category
Segment
Forecast~12/1/2016
Forecast~11/1/2016
Forecast~10/1/2016
Forecast~9/1/2016
Forecast~8/1/2016
Forecast~7/1/2016
Forecast~6/1/2016
Forecast~5/1/2016
Forecast~4/1/2016
Forecast~3/1/2016
Forecast~2/1/2016
Forecast~1/1/2016
Budget~12/1/2016

Column1	Column2	Column3	Column4
Category	Segment	Forecast~12/1/2016	Forecast~11/1/2016

Module 3 Lab: Create Budget Fact

Append BudgetFact_Data to the bottom of the headers, then remove the extraneous header rows, finally set first row as header

Column1	Column2	Column3	Column4
Category	Segment	Forecast~12/1/2016	Forecast~11/1/2016

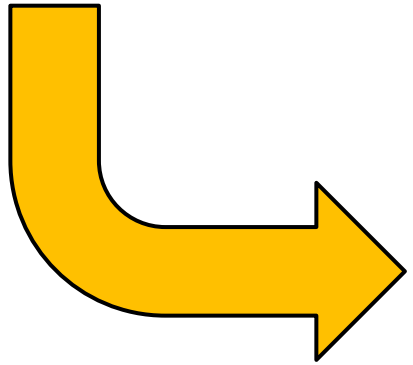
Column1	Column2	Column3	Column4	Column5
Category	Segment	Forecast~12/1/2016	Forecast~11/1/2016	Forecast~10/1/2016
		Forecast	Forecast	Forecast
		2016	2016	2016
Category	Segment	Dec	Nov	Oct
Accessory	Accessory	44190.57888	50598.81566	54740.5709
Mix	All Season	11442.14474	14120.78693	18109.64804
Mix	Productivity	19538.89812	17597.55926	22835.18396
Rural	Select	311.708775	172.2601125	662.79129

Category	Segment	Forecast~12/1/2016	Forecast~11/1/2016	Forecast~10/1/2016
Accessory	Accessory	44190.57888	50598.81566	54740.5709
Mix	All Season	11442.14474	14120.78693	18109.64804
Mix	Productivity	19538.89812	17597.55926	22835.18396
Rural	Select	311.708775	172.2601125	662.79129
Urban	Convenience	120710.4406	129923.2814	169468.7696

Module 3 Lab: Create Budget Fact

Unpivot Forecast Columns, Split and Rename them, then set data types

Category	Segment	Forecast~12/1/2016	Forecast~11/1/2016	Forecast~10/1/2016
Accessory	Accessory	44190.57888	50598.81566	54740.5709
Mix	All Season	11442.14474	14120.78693	18109.64804
Mix	Productivity	19538.89812	17597.55926	22835.18396
Rural	Select	311.708775	172.2601125	662.79129
Urban	Convenience	120710.4406	129923.2814	169468.7696

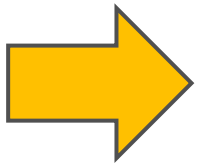


Category	Segment	Attribute	Value
Accessory	Accessory	Forecast~12/1/2016	44190.57888
Accessory	Accessory	Forecast~11/1/2016	50598.81566
Accessory	Accessory	Forecast~10/1/2016	54740.5709
Accessory	Accessory	Forecast~9/1/2016	64442.9079
Accessory	Accessory	Forecast~8/1/2016	98285.91328
Accessory	Accessory	Forecast~7/1/2016	61545.41064
Accessory	Accessory	Forecast~6/1/2016	91299.65402
Accessory	Accessory	Forecast~5/1/2016	120579.7261
Accessory	Accessory	Forecast~4/1/2016	103436.9095
Accessory	Accessory	Forecast~3/1/2016	93558.89718
Accessory	Accessory	Forecast~2/1/2016	58793.20125
Accessory	Accessory	Forecast~1/1/2016	51771.46338
Accessory	Accessory	Budget~12/1/2016	44650.89741

Module 3 Lab: Create Budget Fact

Merge in CatSegID and remove extraneous columns

Category	Segment	Scenario	Date
Accessory	Accessory	Forecast	12/1/2016
Accessory	Accessory	Forecast	11/1/2016
Accessory	Accessory	Forecast	10/1/2016
Accessory	Accessory	Forecast	9/1/2016
Accessory	Accessory	Forecast	8/1/2016
Accessory	Accessory	Forecast	7/1/2016
Accessory	Accessory	Forecast	6/1/2016
Accessory	Accessory	Forecast	5/1/2016



CatSegID	Scenario	Date	Budget Amount
3	Forecast	12/1/2016	44,190.58
3	Forecast	11/1/2016	50,598.82
3	Forecast	10/1/2016	54,740.57
3	Forecast	9/1/2016	64,442.91
3	Forecast	8/1/2016	98,285.91
3	Forecast	7/1/2016	61,545.41
3	Forecast	6/1/2016	91,299.65
3	Forecast	5/1/2016	120,579.73

KNOWLEDGE CHECK

1. Which transformation type turns all rows into columns and columns into rows?
2. Which Merge type would you use to get all rows from two queries, even if the joined column(s) did not match?
3. What happens in an Append if the column names do not match?

Lab 03

Open Data Shaping Lab file and follow the instructions for Lab 03 – Create Budget Fact table

Module 4:
Variable, Parameters and Functions

MODULE OBJECTIVES

Objectives

- Understand what Variables and Parameters
- Understand Custom Functions
- Understand the benefits to parameterizing your queries
- Understand what “Enter Data” is and how it can be used
- Understand how to organize your queries using folder groups

Agenda

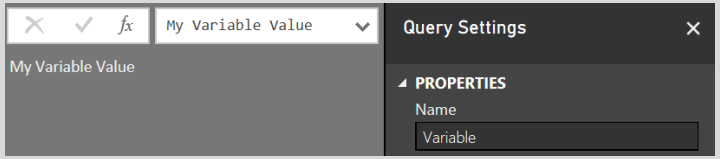
- Power BI M Variables & Parameters
- Enter Data
- Leveraging Variables & Parameters
- Create Change Log

Variable and Parameters

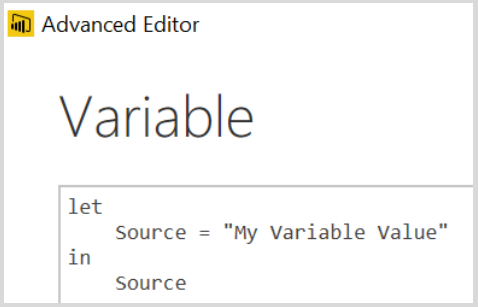
M Variables

Query with a constant value, table or list result

- Create with a new blank query
- Query name is the Variable name to use in other queries
- Type the variable value in the formula bar



- Advanced Editor would look like this:

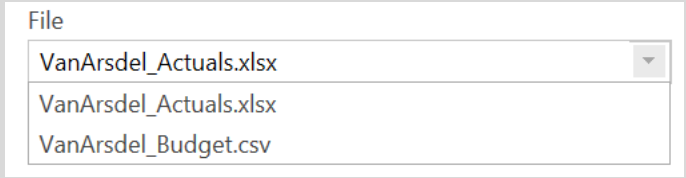


M Parameters

Parameters take Variables to the next level.

- Parameters can be any VALID data type
- The most common are hard keyed single, switchable values

- The values list for a parameter can also come from a query that is formatted as a **LIST**



- Parameter used as Source Input

```
= Excel.Workbook(File.Contents( Path & ActualsFile ), null, true)
```

Parameter Usage

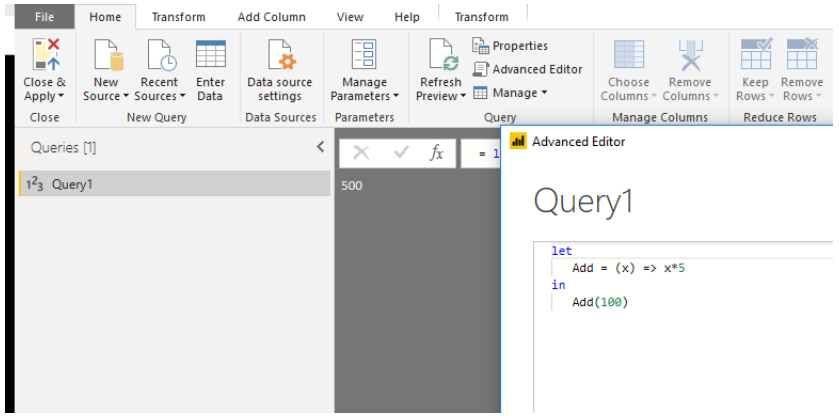
Be Creative!

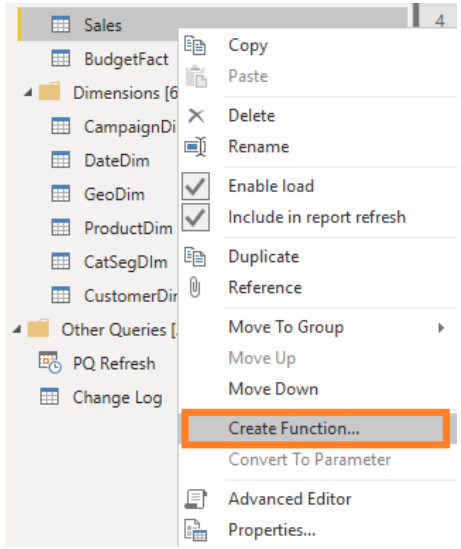
- SQL Server
  - For Database Name
  - For Server (Switch Dev, Test, Prod)
- "Where Clauses" in SQL Queries
  - To pull all Orgs, or just a single one
- SharePoint Team site names
  - If they are based on a template, you can easily share queries
- File Paths and File Names
  - Have a single place where you need to make a change

Custom Functions

Custom Functions help you re-use code

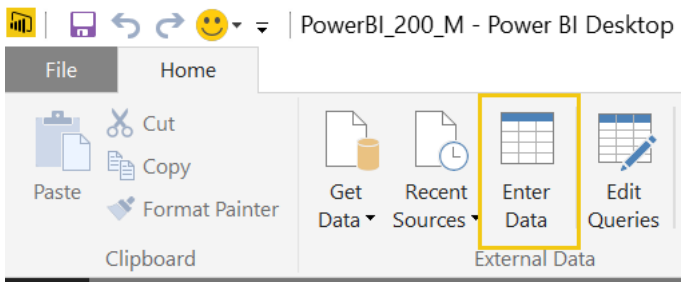
1. Create function using Blank query
2. Create function using Baseline query

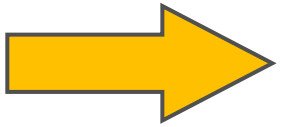


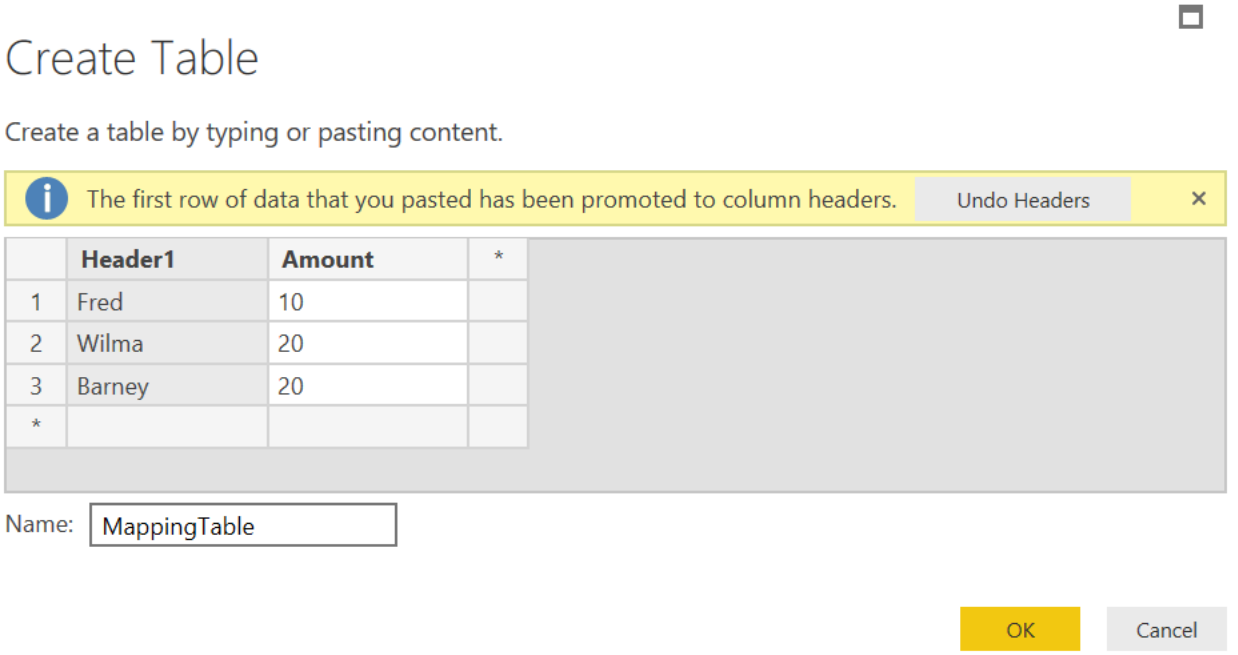


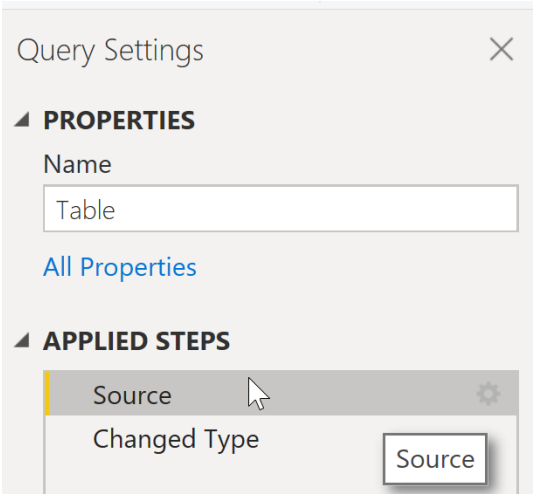
Enter Data

User **Enter Data** to add a table on the fly. They make great mapping tables.









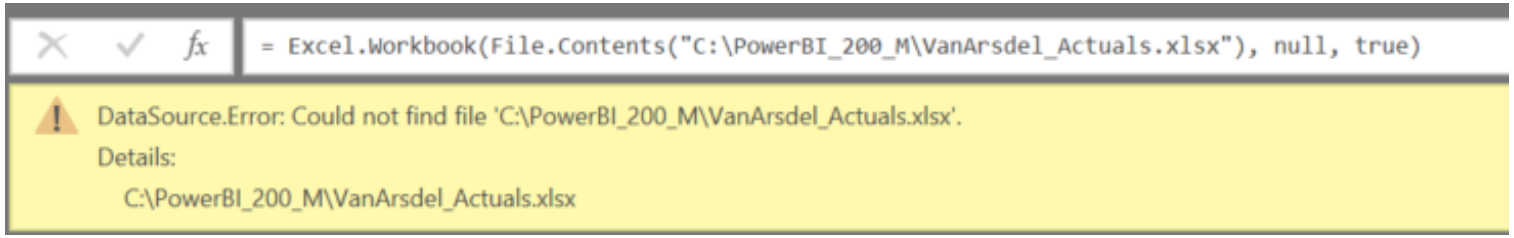
<- To add additional rows or columns later, click the gear icon.

Module 4 Lab:
Creating Parameters and Paths

Module 4 Lab: File Source considerations

All of the queries in our file are dependent on the file being in a specified path

- If you move the source file and try to refresh the queries you will receive an error
  - C:\ to File Share
  - Different folder location
  - To SharePoint
  - To OneDrive



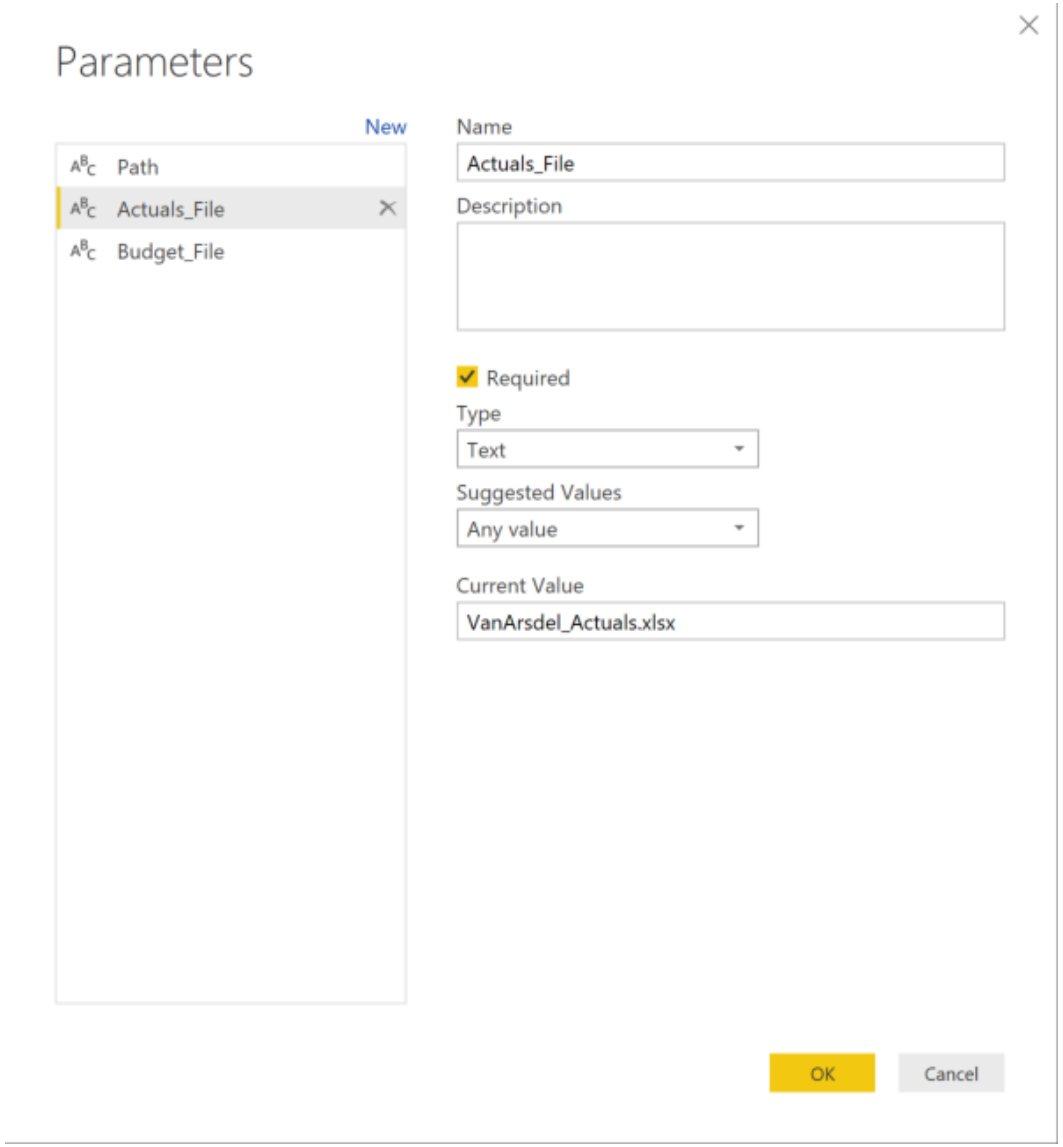
- Each query will need to be changed EACH time the file is moved

Creating Parameters with Dynamic Paths resolves this issue

Module 4 Lab: Create Parameters

Create Parameters

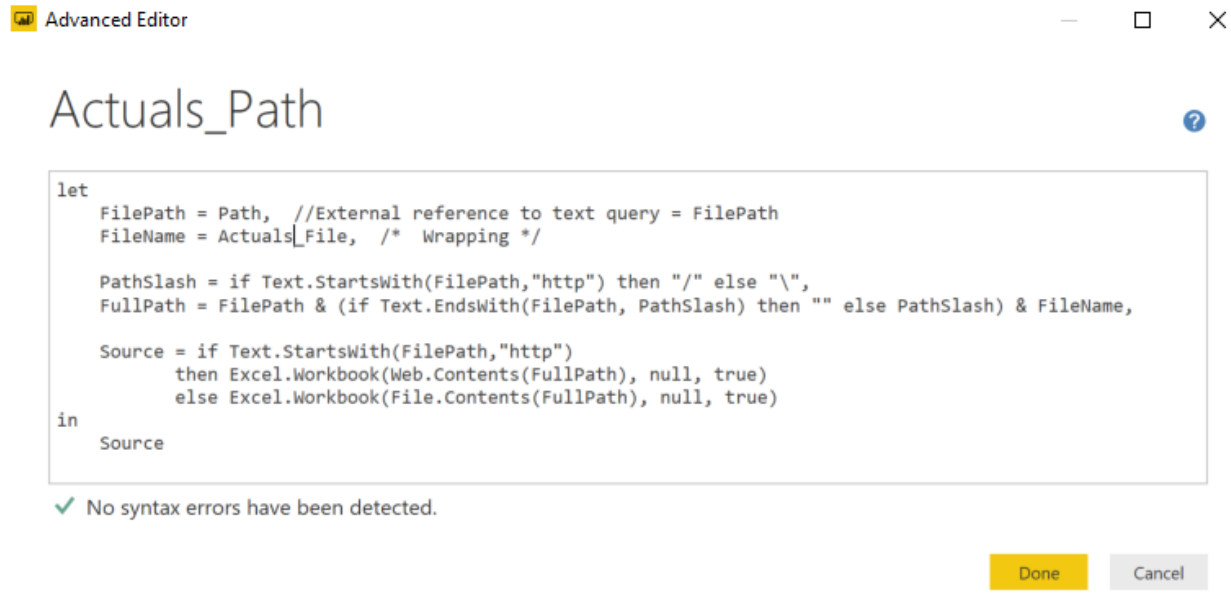
- Actual File
- Budget File
- Path

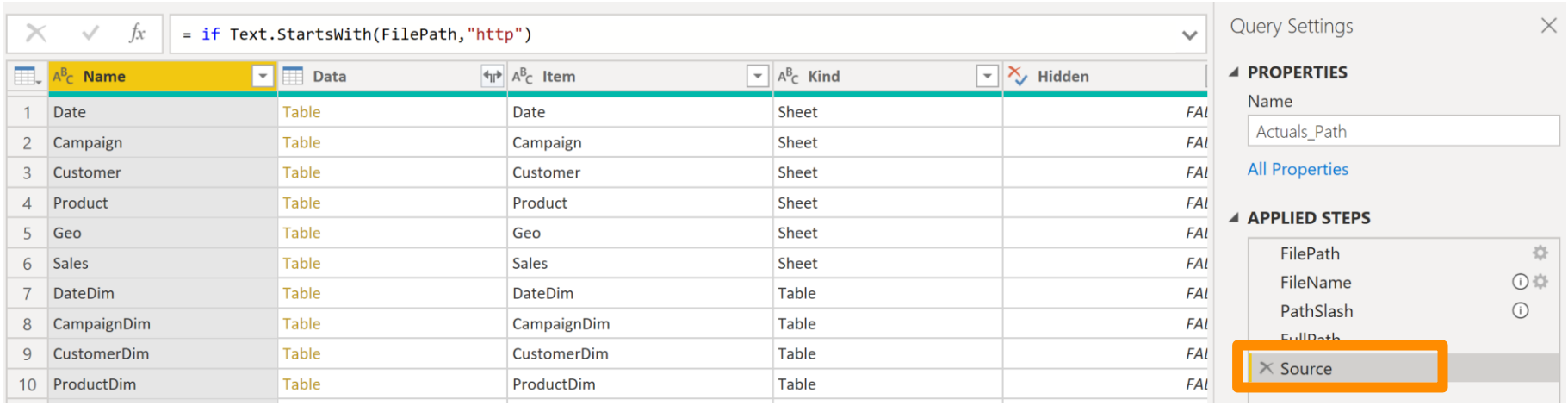


Module 4 Lab: Dynamic Path to Excel Source File

Use blank query to create Dynamic Path

- Uses Blank Query
- Populate Advanced Editor of query with text from file provided
- For each query which uses the Excel source, update the Source (applied step) with the new variable name ***=Actuals_Path***





Module 4 Lab: Dynamic Path to Excel Source File

Use the **Actuals_Path** query as a variable in other queries

- In Advanced Editor, update the Excel source to the new Actuals_Path source.

CampaignDim — Original

```
let
    Source = Excel.Workbook(File.Contents("C:\PowerBI_200_M\PowerBI_200_M_Data.xlsx"), null, true),

    CampaignDim_Table = Source{[Item="CampaignDim",Kind="Table"]}[Data],
    #"Changed Type" = Table.TransformColumnTypes(CampaignDim_Table,{{"CampaignId", type text}, {"Traffic Chann
in
    #"Changed Type"
```

CampaignDim — Updated

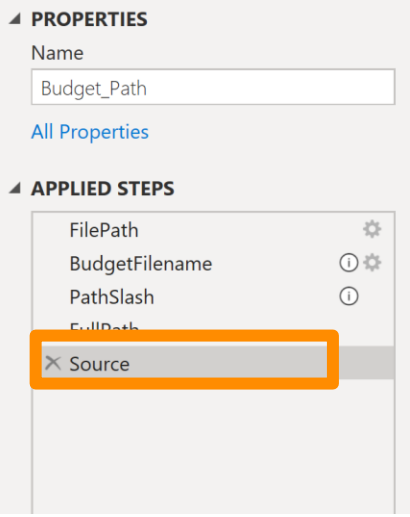
```
let
    Source = Actuals_Path,
    Campaign_Sheet = Source{[Item="CampaignDim",Kind="Table"]}[Data],
    #"Changed Type" = Table.TransformColumnTypes(Campaign_Sheet,{{"CampaignID", Int64.Type}, {"Traffic
in
    #"Changed Type"
```

Module 4 Lab: Dynamic Path to CSV Source File

Follow a similar patter to make the Budget CSV file dynamic

- Uses Blank Query
- Populate Advanced Editor of query with text from file provided
- For each query which uses the Excel source, update the Source (applied step) with the new variable name
  =Budget_Path

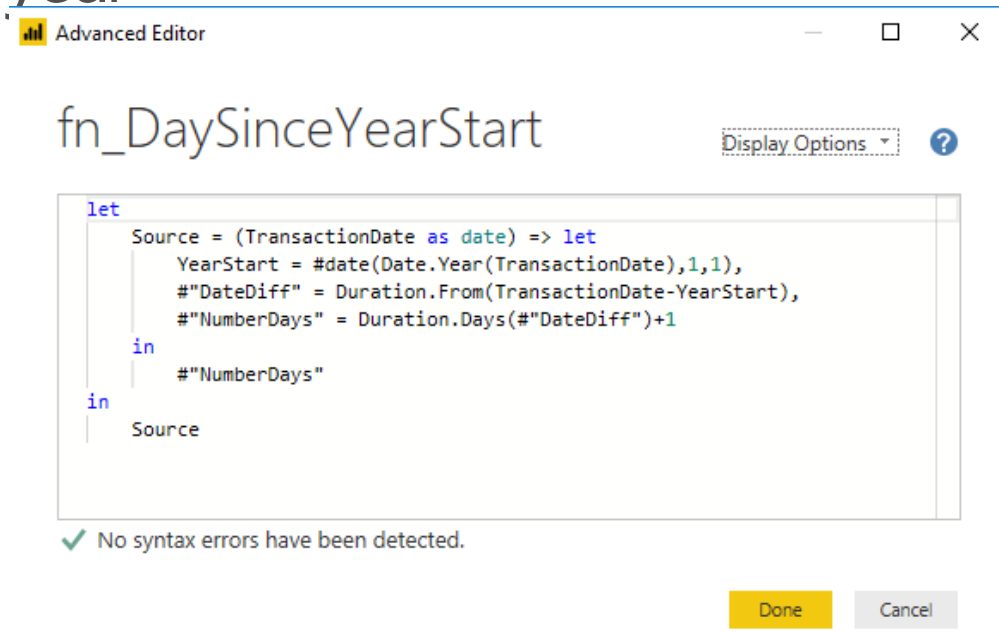
	Column1	Column2	Column3	Column4	Column5
1	Budget Spreadsheet for VanArsdel				
2					
3					
4			Forecast	Forecast	Forecast
5			2016	2016	2016
6	Category	Segment	Dec	Nov	Oct
7	Accessory	Accessory	44190.57888	50598.81566	54740.5709
8	Mix	All Season	11442.14474	14120.78693	18109.64804
9	Mix	Productivity	19538.89812	17597.55926	22835.18396
10	Rural	Select	311.708775	172.2601125	662.79129
11	Urban	Convenience	120710.4406	129923.2814	169468.7696
12	Urban	Extreme	20868.84072	46971.33037	70793.02886
13	Urban	Moderation	251155.7122	322984.2215	362385.6466

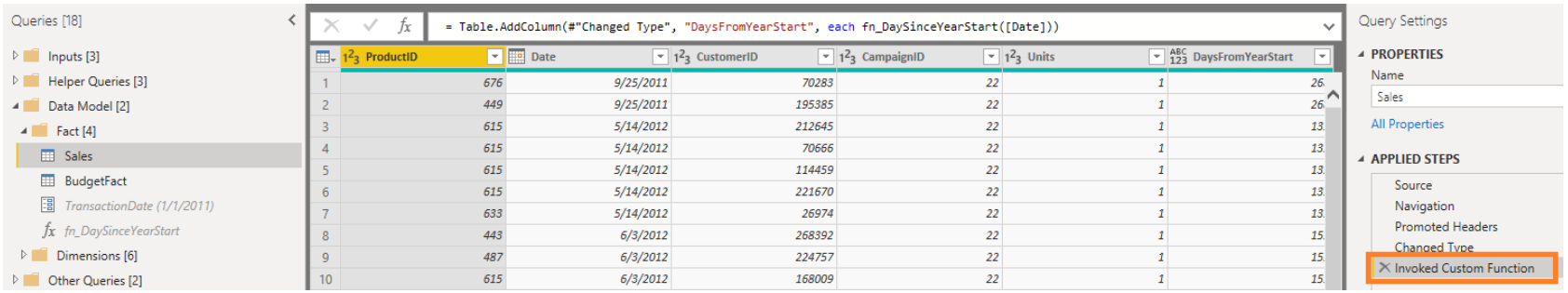


Module 4 Lab: Custom Function

Create a function to get number of days from start of year

- Uses Blank Query
- Populate Advanced Editor of query with text from file provided
- Invoke the custom function from Sales table. This will provide the number of days from start of year for each transaction

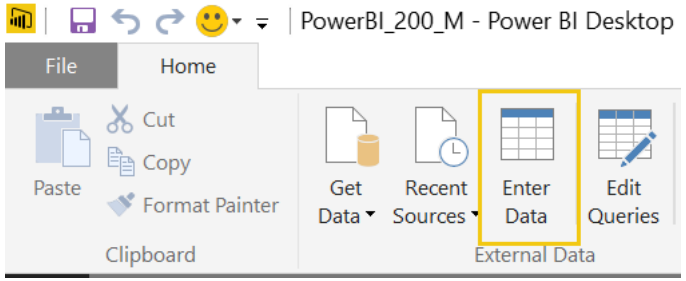




Module 4 Lab: Create a Change Log

Use Enter Data to create a Change Log

- Create a table within query to house the change log
- Name the query **Change Log**
- Update table each time a change is made to the code
- If you decide to load file to data model, then you have the ability to report on current version and last update date



Create Table

Create a table by typing or pasting content.

	Version	Date	Made By	Change	*
1	2.0	12/15/2016	V-Barran (Barbara	Version 2 Updates	
2	2.01	1/3/2017	V-Barran (Barbara	Lab Updates	
*					

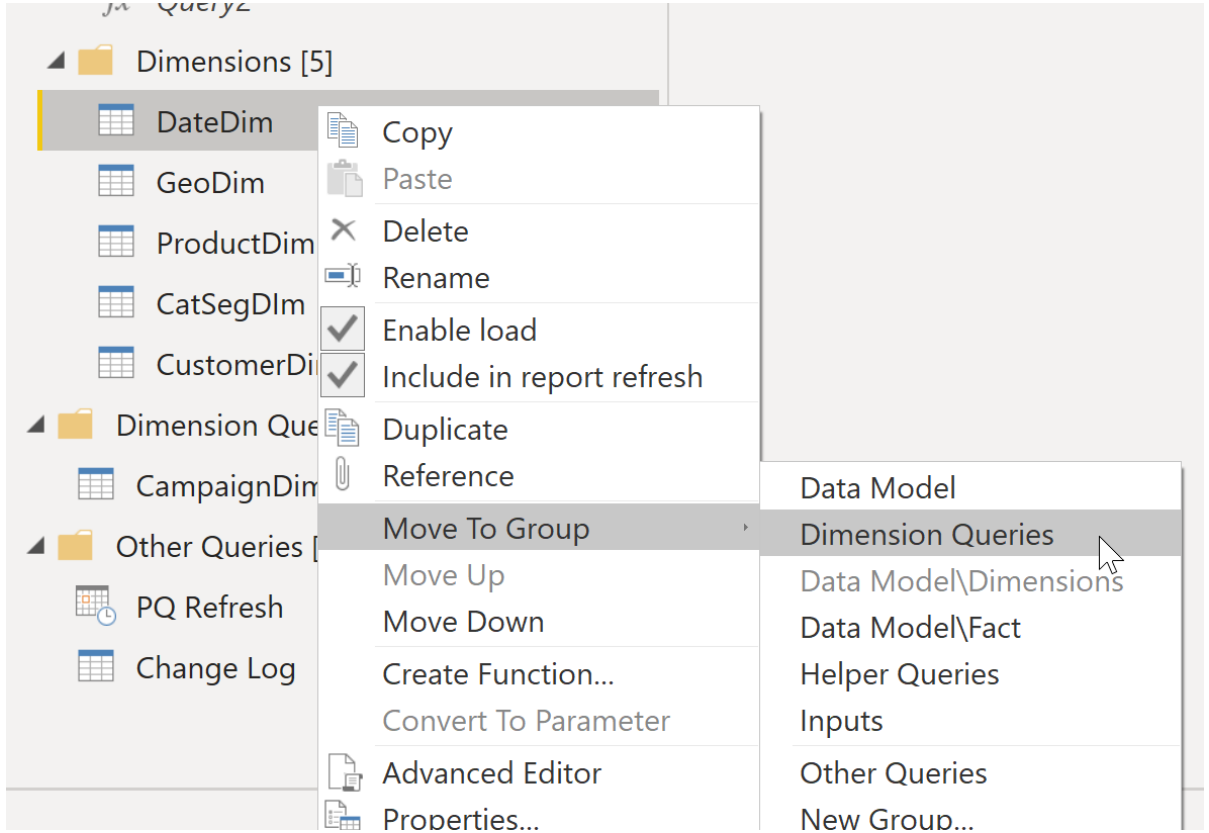
Report

Version	Change	Date
2.00	Version 2 Updates	12/15/16
2.01	Lab Updates	1/3/17

Module 4 Lab: Organize your Queries

Use Folders to Group Queries

- Create logical groups to Manage your queries
  - Can create sub-groups
- Use Drag and Drop to move queries to folders



Custom Connectors

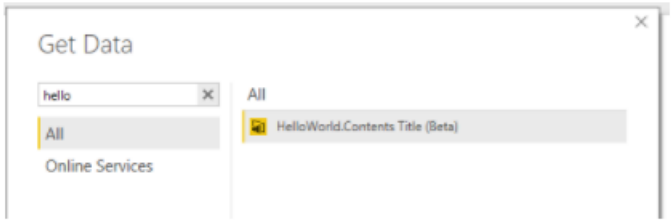
Use the M Language to create your own Custom Connector!

- Samples and SDK on github

https://github.com/Microsoft/DataConnectors

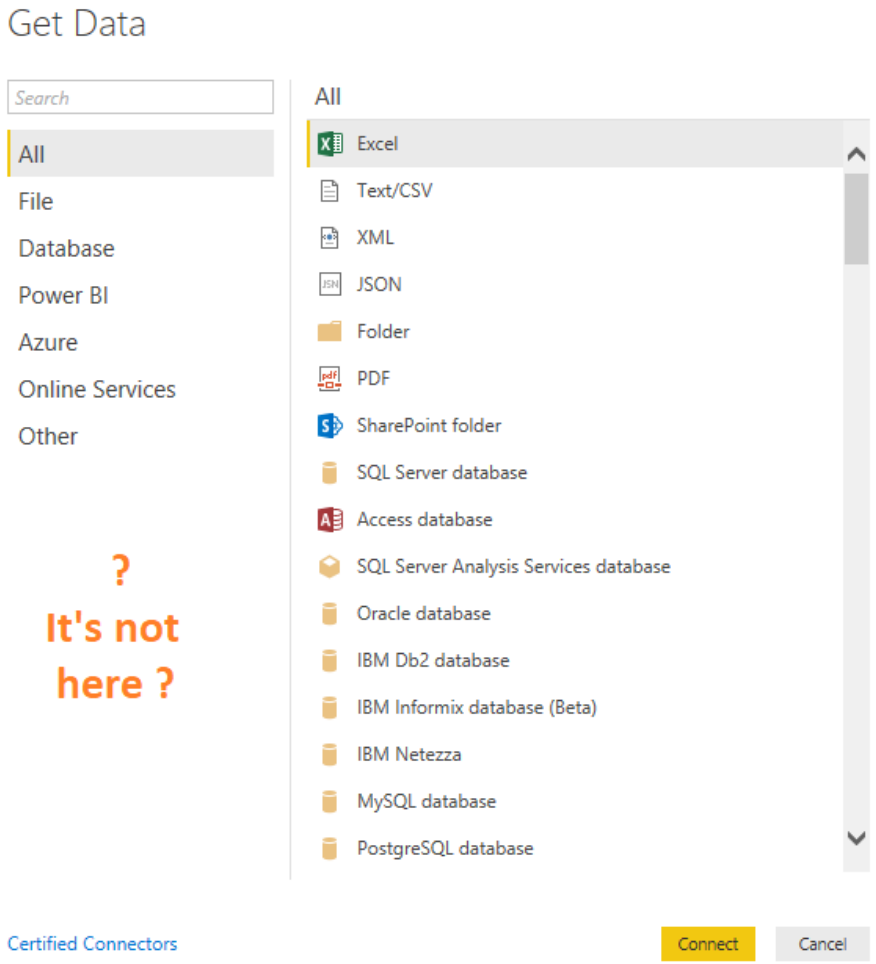
Getting Started with Data Connectors

Data Connectors for Power BI enables users to connect to and access data from your application, service, or data source, providing them with rich business intelligence and robust analytics over multiple data sources. By integrating seamlessly into the Power Query connectivity experience in Power BI Desktop, Data Connectors make it easy for power users to query, shape and mashup data from your app to build reports and dashboards that meet the needs of their organization.



Data Connectors are created using the M language. This is the same language used by the Power Query user experience found in Power BI Desktop and Excel 2016. Extensions allow you to define new functions for the M language, and can be used to enable connectivity to new data sources. While this document will focus on defining new connectors, much of the same process applies to defining general purpose M functions. Extensions can vary in complexity, from simple wrappers that essentially just provide "branding" over existing data source functions, to rich connectors that support Direct Query.

Please see the Data Connector technical reference for more details.



KNOWLEDGE CHECK

1. A Parameter can be used to easily switch your queries to a different Server (T/F)?
2. Provide one example for usage of **Enter Data**?
3. Advantages to putting queries in Query Groups (folders)?

Lab 04

Open Data Shaping Lab file and follow the instructions for Lab 04 – Exercises A – D

a. Create Lab Parameters

b. Create dynamic path to excel

c. Create dynamic path to csv

d. Create a custom function

Module 5: Optimization Techniques

MODULE OBJECTIVES

Objectives

- Understand best practices when working with large data volumes
- Understand best practices to use the power of the underlying data sources
- Understand best practices in Direct Query mode

Data Loading/Refresh: High Level

- Power Query/M has a lot of point and click functions
- Sub-optimal step ordering and configuration can slow down development in Desktop and refresh once published

Goals

- Pushing work to the data source
- Avoiding memory/cpu intensive operations, or spreading them out
- These also reduce Desktop slowdowns/crashes

Tip: Disable Background Data in Desktop

Scenario

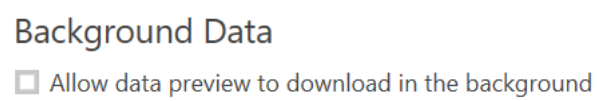
- Slow sources can take a while to return data
- Many tables/parameterized queries slow/failed refresh

Why is it undesired?

- Frequent waiting in Power Query, cpu/mem

Proposed solutions

- Disable in Power Query Editor Options



- Use parameter flags to return smaller subsets of data when still editing queries

```
#"Removed Columns" = Table.RemoveColumns(DaxBook_Sales,{"PromotionKey", "CurrencyKey", "Unit Discount", "Order Line Number", "Order
#"Filtered Rows" = Table.SelectRows(#"Removed Columns", each ([Order Number] = "200701022CS425" or [Order Number] = "200701023CS425"
FilterToggle = if DBFilterToggle=1 then #"Removed Columns" else #"Filtered Rows",
```

Tip: Leverage Query Folding

Scenario

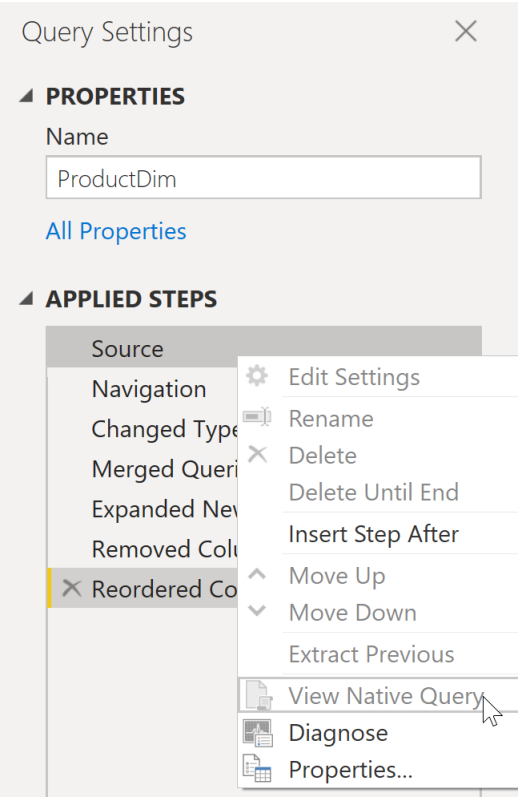
- Some transforms can be converted to native source language

Why is it desired?

- Operations pushed down to source are often
- much faster

Proposed solutions

- Right click step, View Native Query – folded
- Ensure all foldable steps happen first, together



Tip: Disable Privacy Settings

Scenario

- Using data from one source to filter another
- Default Privacy settings prevent data being leaked between sources

Why is it undesired?

- Entire table needs to be loaded to memory before applying filter

Proposed solutions

- If safe, disable privacy or set both sources to Organizational

Privacy Settings can be confusing, see Chris Webb

Tip: Avoid Transforms that Scan Whole Tables

Scenario

- Grouping/aggregations/merges performed in M

Why is it undesired?

- If not folded, the entire table needs to be loaded to memory before moving to next step

Proposed solutions

- Consider DAX measure instead
- Customize source query using Advanced Editor

Tip: Avoid joining large tables in Query editor

Scenario

- Merge used in Query Editor across 2 large tables (perhaps Millions of rows)

Why is it undesired?

- Entire table needs to be loaded to memory to perform join

Proposed solutions

- Join at source using custom query, then manipulate further in Query Editor

Tip: Don’t load intermediate queries to memory

Scenario

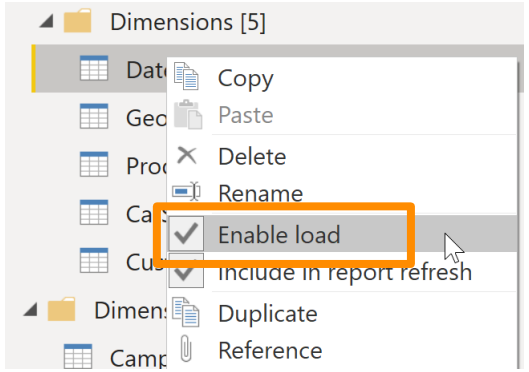
- Model contains M queries that are only used as intermediate tables in other queries. Author hides these tables to prevent user access.

Why is it undesired?

- Consumes memory when model is being used, but table never touched interactively
- Hiding alone isn’t secure

Proposed solutions

- Disable load in Query Editor



Tip: Turn off Auto Date/Time

Scenario

- Source data has many date columns with broad ranges
- Default setting is On
- System uses 1/1/1900 or similar for blank date

Why is it undesired?

- Creates many internal date tables that can be a significant size and can bloat smaller models

Direct Query Optimization: High Level

- By design, DQ issues parallel queries for every interaction that affects report results
- Sources can be overwhelmed by many/expensive queries
- In some cases, gateway or PBI can be the limiting factor

Goals

- Exploit the data sources strengths
- Reduce concurrency and queueing
- In general, all tips to reduce no./size of queries will benefit DQ greatly

Tip: Push Transformations to DQ Source

Scenario

- Performing filters, derived calcs and other transforms in M

Why is it undesired?

- In many cases, predicates won't be folded unless in specific order.
- DQ sources can be highly optimized for joins and filters so not folding often slower in Power BI

Proposed solutions

- Avoid transformations in M
- Ensure all foldable operations are performed together, first
- Write native code to source (e.g. TSQL, PL/SQL)

Lab 05

Open Data Shaping Lab file and follow the instructions for Lab 05 – Performance Best Practices

Questions?

Power BI Support Resources

Contact Support

Report Errors, Issues – Support.PowerBI.com

Resources *use presentation mode to click the hyperlinks*

- *Community.PowerBI.com – Community Forum*
- *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*
- *Visuals.PowerBI.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*
- *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*

Instructors:

Appendix A: Knowledge Check Answers

KNOWLEDGE CHECK Answers – Module 1

1. What allow tables in the data model to “talk” to one another?
    1. *Relationships*
2. What type of schema has all attributes for model in a single table?
    1. *Denormalized or “Flat” – the Kitchen Sink Schema*
3. When would you use multiple Fact tables
    1. *Multiple Reasons*
        1. *Different Time Grains*
        2. *Different dimensional attributes*
        3. *One dimension at different grain: Product v. Product Category*
        4. *Facts are sourced from different systems with different refresh cadences.*

KNOWLEDGE CHECK Answers – Module 2

1. You cannot import multiple tables from a single source (T/F)?
    1. *FALSE – you CAN import multiple tables from a single source*
2. Why would we need a snowflake dimension?
    1. *A snowflake dimension can be useful to remove duplicating fields from a large dimension. Or to use the snowflake for a primary dimension off a second fact at a higher grain.*
3. What syntax would you use to find out if a column has "Abc" as its first three character?
    1. *Text.StartsWith([Column], "Abc")*
4. Or Last three?
    1. *Text.EndsWith([Column], "Abc")*

KNOWLEDGE CHECK Answers – Module 3

1. Which transformation type turns all rows into columns and columns into rows?
    1. *Transpose*
2. Which Merge type would you use to get all rows from two queries, even if the joined column(s) did not match?
    1. *Full Outer Join*
3. What happens in an Append if the column names do not match?
    1. *Extra Columns are added to the right of the Joining columns. If the field did not exist in the source for the row, then the field is null*

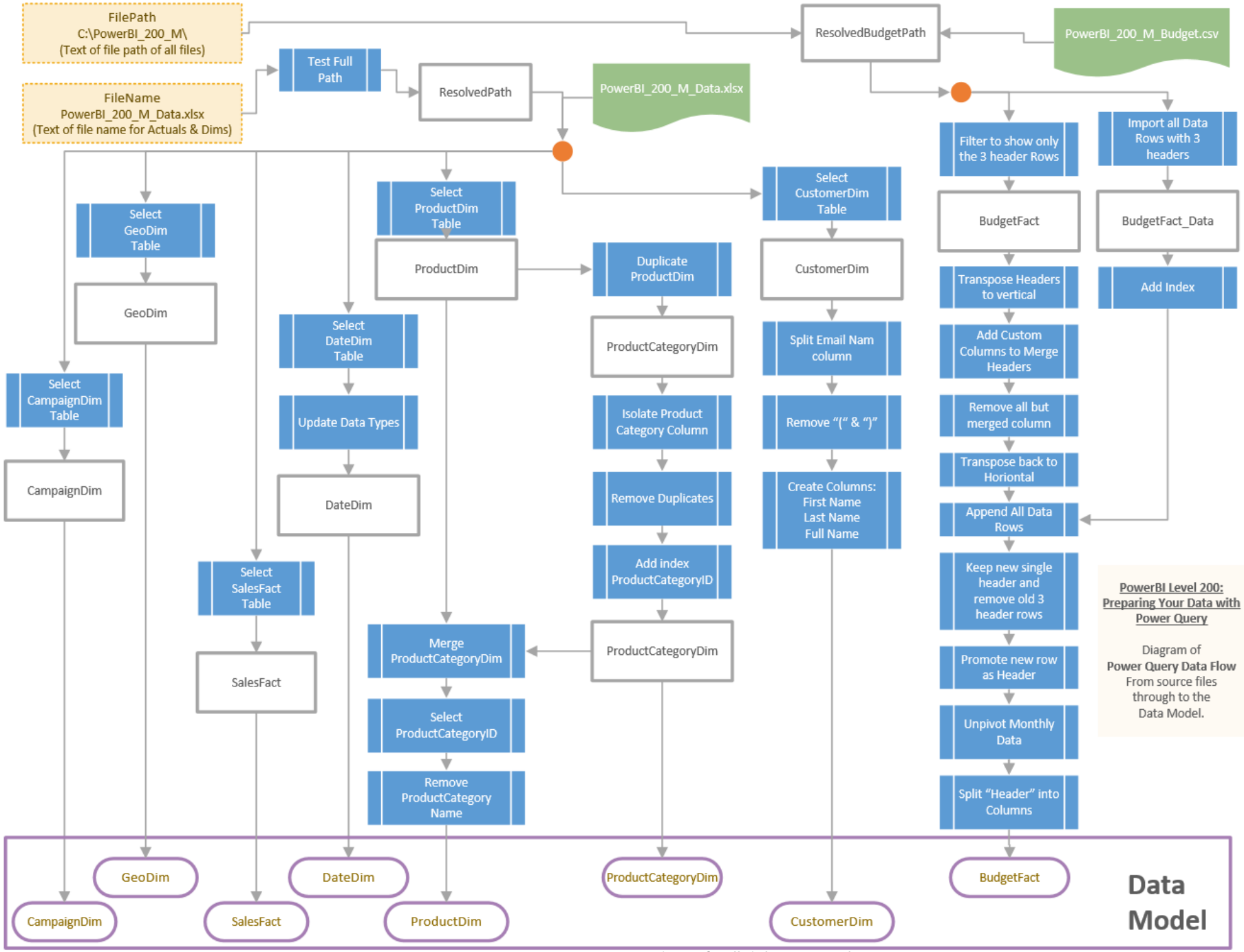
KNOWLEDGE CHECK Answers – Module 4

1. A Parameter can be used to easily switch your queries to a different Server (T/F)?
    1. *True – the Parameter makes this easy as all of the possible values can be populated, so the user just has to choose.*

2. Provide one example for usage of **Enter Data**?
    1. *Change Log or Mapping Table*

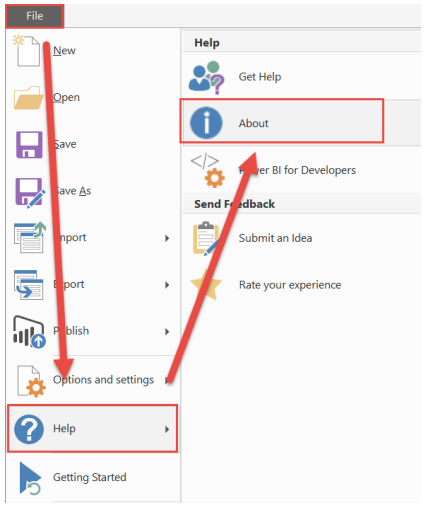
3. Advantages to putting queries in Query Groups (folders)?
    1. *Easier to keep the code organized.*

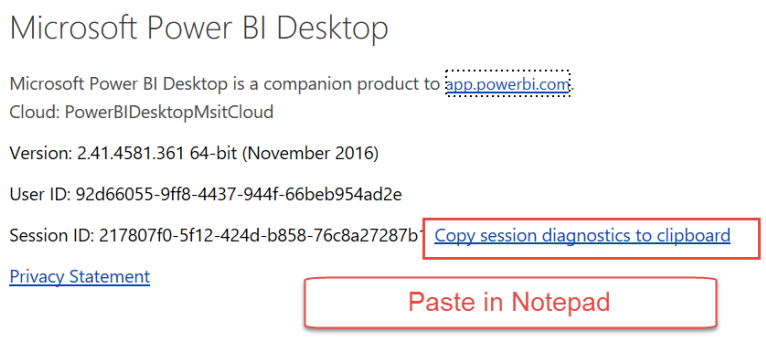
Appendix B: Example of Data Flow diagram using Visio



Appendix C:
Extract all M queries in a PBIX

Extract M Queries






```
section Section1;

shared ToolsVSO = let
    Source = Excel.Workbook(Web.Contents("
    https://microsoft.sharepoint.com/teams/o365-sfb/Shared%20Documents/Reports/Skype%20at%20Mic
    VSO_Sheet = Source{[Item="VVC Tools VSO",Kind="Sheet"]}[Data],
    #"Removed Top Rows" = Table.Skip(VSO_Sheet,1),
    #"Promoted Headers" = Table.PromoteHeaders(#"Removed Top Rows"),
    #"Removed Columns0" = Table.RemoveColumns(#"Promoted Headers",{"Title 2", "Title 4"}),
    #"Added Task Name" = Table.AddColumn(#"Removed Columns0", "Task", each if [Title 9] <> null
if [Title 8] <> null then [Title 8] else
if [Title 7] <> null then [Title 7] else
if [Title 6] <> null then [Title 6] else
"Other"),
    #"Added Work Item Type" = Table.AddColumn(#"Added Task Name", "Work_Item_Type", each if [Ta
if [Title 5] <> null then "Story" else
(if [Title 3] <> null then "Feature" else
"Epic")),
    #"Reordered Columns" = Table.ReorderColumns(#"Added Work Item Type",{"ID", "Work Item Type"
    "Title 6", "Title 7", "Title 8", "Title 9", "Assigned To", "State", "Tags", "Area Path", "C
    "Story Points", "Target Date", "Completed Work", "Remaining Work", "Stack Rank", "Task"}),
    #"Split Column by Delimiter" = Table.SplitColumn(#"Reordered Columns","Iteration Path",Spli
    true),{"Iteration Path.1", "Iteration Path.2"}),
    #"Changed Type1" = Table.TransformColumnTypes(#"Split Column by Delimiter",{{"Iteration Pat
    #"Added Custom1" = Table.AddColumn(#"Changed Type1", "Iteration Path", each if [Iteration P
    Path.2]<>"" then ([Iteration Path.2] & "-2") else null)),
    #"Renamed Columns" = Table.RenameColumns(#"Added Custom1",{{"Iteration Path", "Sprint Plann
    #"Removed Columns" = Table.RemoveColumns(#"Renamed Columns",{"Iteration Path.1", "Iteration
    #"Added Epic ID" = Table.AddColumn(#"Removed Columns", "Epic ID", each if [Work_Item_Type]
    #"Added Feature ID" = Table.AddColumn(#"Added Epic ID", "Feature ID", each if [Work Item Ty
```

Appendix D: Other M Topics

Date and Time Literals

Hard code a Date or time literal into your query

```
= Table.AddColumn(Source, "Result", each Expression.Evaluate([#"Example (as text)"]))
```

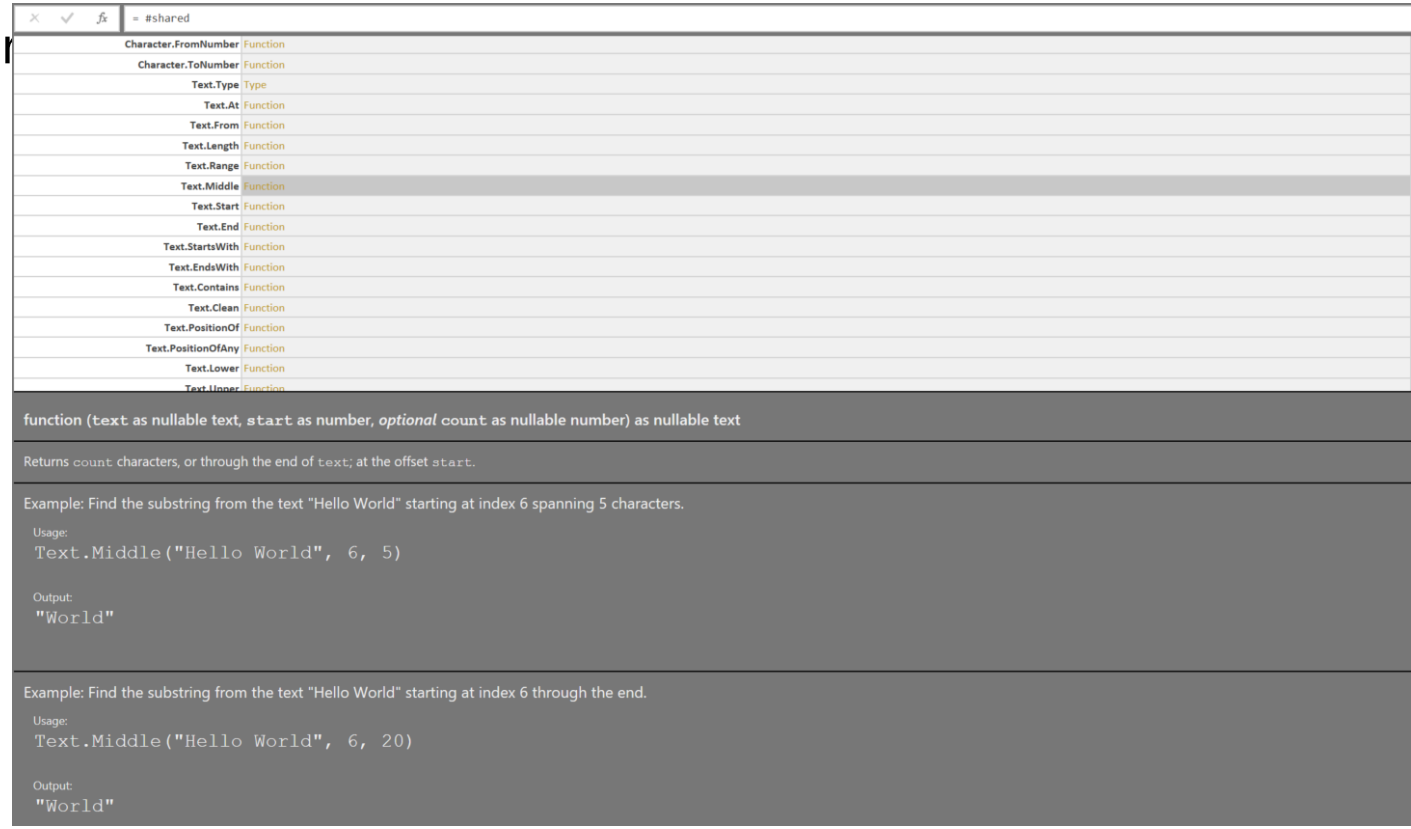
	Function	Example (as text)	Components	Result
1	#date	#date(2015, 4, 22)	year, month, day	*4/22/2015*
2	#time	#time(09, 15, 00)	hour, minute, second	*9:15:00 AM*
3	#datetime	#datetime(2015, 4, 22, 09, 15, 00)	year, month, day, hour, minute, second	*4/22/2015 9:15:00 AM*
4	#datetimezone	#datetimezone(2015, 4, 22, 09,15,00, 09, 00)	year, month, day, hour, minute, second, timezone	*4/22/2015 9:15:00 AM +09:00*
5	#duration	#duration(2, 1, 30, 45)	day, hour, minute, second	*2.01:30:45*
6	#infinity	#infinity	no extra components	*Infinity*

#shared

Access function documentation from within Edit Queries

- Type **=#shared** in Power Query formula bar
- Convert list to table
- Sort list on function name
- Click in whitespace next to function to see full *UPDATED* Help Text & Examples

- If you click on the “Function” word, it adds a step to the Power Query focusing on that function



#table

Access function documentation from within Edit Queries

- #table({ “title1”, ”title2”} , { { 0,1 }, { 1,0 }, { 0, 0 } })

```
= #table({"Name", "Rank", "ID"},{{"Fred",10,99},{"George",3,42},{"Mary",5,64},{"Erik", 1, 99}})
```

	Name	Rank	ID
1	Fred	10	99
2	George	3	42
3	Mary	5	64
4	Erik	1	99

Use to append a null id field to existing pull which does not currently have the value

```
= #table({"ID", "Name", "Description"},{{-1,"na",null}})
```

	ID	Name	Description
1	-1	na	null

M Object Types

Recap of Object Types

- **Formula Expression –** Evaluated expression which is computed to yield a result. The result can be a **List**, **Record**, **Table** or single (scalar) **Value**
- **List –** An ordered sequence of values i.e. {1, 2, 3}
- **Field** - A field is a name/value pair
- **Record** - A set of fields. A field is a name/value pair where the name is a text value that is unique within the field's record. For example: [CustomerID = 1, Name = "Bob"]
- **Operators** M supports common Operators including >, >=, <, <=, =, <>, or, and, not, +, -, *, /, &

try...otherwise

Example of Error handling in a formula

```
= Table.AddColumn(Source, "Round Number Down", each try Number.Round([Value to Convert] , 0) otherwise null )
```

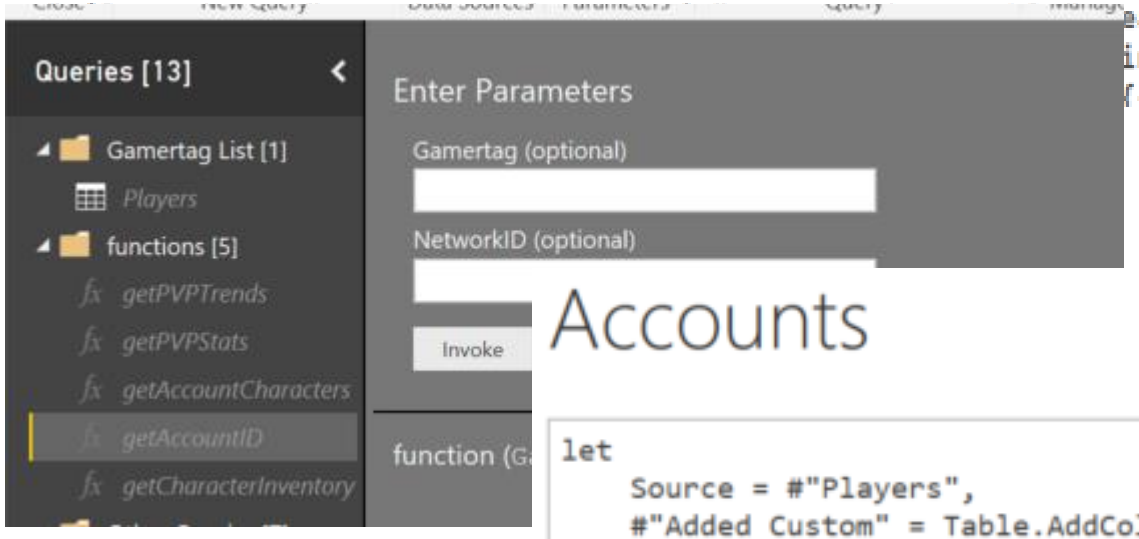
	Value to Convert	Round Number Do...
1	*3.141593*	*3*
2	*42*	*42*
3	nan	*null*
4	Fred	*null*
5	*10000*	*10000*

functions

Advanced feature! Parameterize a query, then call it for each row in other query

getAccountID

```
(Gamertag, NetworkID) => let
    Source = Json.Document(Web.Contents("http://www.bungie.net/platform/destiny/SearchDestinyPlayer/"&NetworkID&"/" & Gamertag,
    Response = Source[Response],
    Response1 = Response{0},
    #"Converted to Table" = Record.ToTable(Response1),
```



Accounts

```
let
    Source = #"Players",
    #"Added Custom" = Table.AddColumn(Source, "AccountID", each getAccountID([Gamertag],[NetworkID])),
    #"Expanded AccountID" = Table.ExpandTableColumn(#"Added Custom", "AccountID", {"accountID"}, {"Acco
```