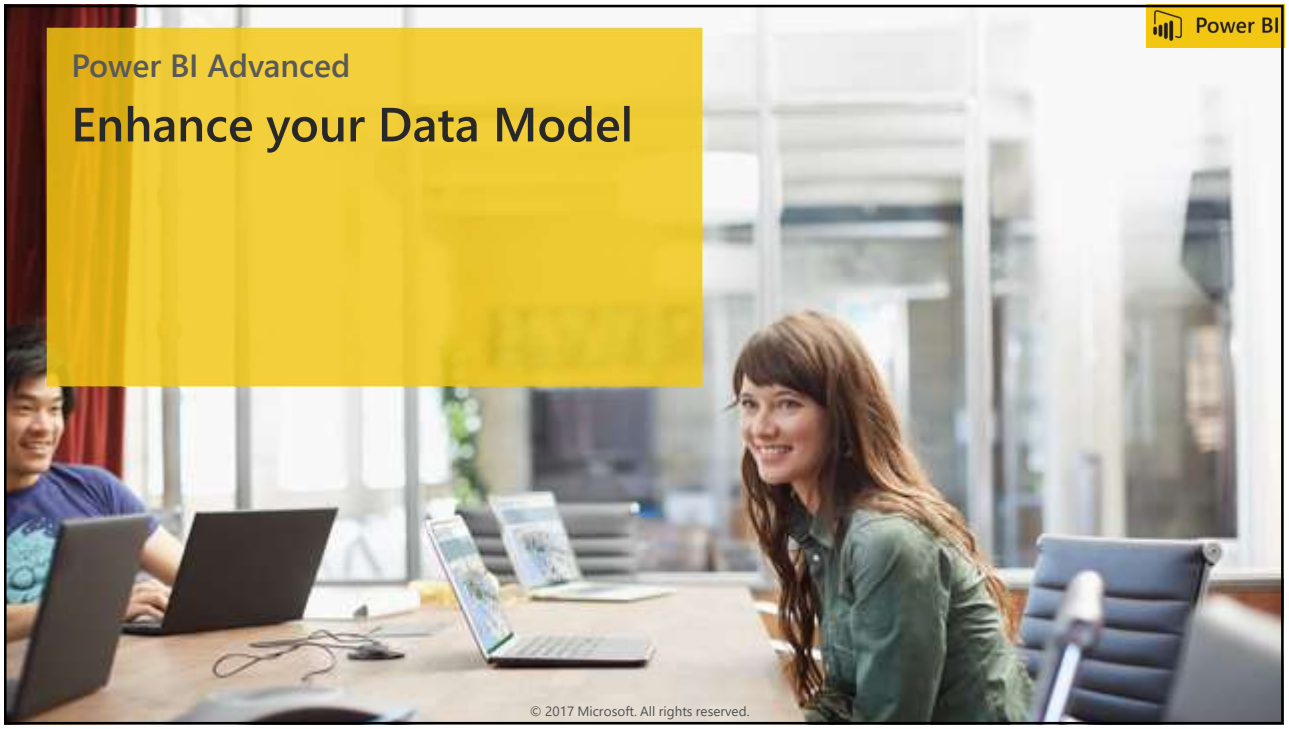


Prerequisites and Setup Steps

Power BI

Internet connectivity: You must be connected to the internet

- At minimum, a computer with 2-cores and 4GB RAM running one of the following version of Windows: Windows 7, Windows 8, (64-bit preferred), Windows 8.1 or Windows 10 or Windows Server 2008 R2 or Windows Server 2012/R2
- Microsoft Power BI Desktop requires Internet Explorer 9 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:

- Copy Files from your USB to **C:\Power BI_Adv_Model** (Please return the USBs)

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.

COURSE OBJECTIVES

Power BI

By the end of this course, you will be able to use DAX to create calculations in a *Power BI Desktop* data model. Specifically you will be able to:

- Understand basic concepts of Data Modeling
- Understand the consequences of data model design decisions
- Understand concepts of calculated columns and measures
- Gain familiarity with standard DAX patterns & CALCULATE
- Understand evaluation contexts and their impact on calculations
- Gain ability to parse data modeling formulas

COURSE AGENDA

Power BI

	Introductions and Overview
Module 1	Data Modeling Basics & Power BI Desktop Internals
Module 2	DAX Calculated Columns & Measures
Module 3	CALCULATE
Module 4	DAX Evaluation Contexts
Module 5	Data Modeling: Time Intelligence Functions
	Wrap-up & Questions

Module 1
Data Modeling Basics & *Power BI Desktop* Internals

MODULE 1 OBJECTIVES

Power BI

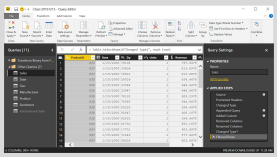
- Understand what is meant by *data model* in the context of Power BI
- Understand the consequences of data model design decisions
- Understand Power BI's data storage architecture and use this knowledge to optimize performance
- Understand consequences of Power BI's data type handling

Power BI Desktop Data Flow

Power BI

Data Sources

Power BI Desktop file (.PBIX)



Close & Apply

Close: Closes Query Editor

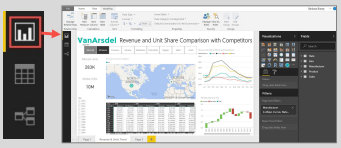
Apply: Loads data from sources to Data Model

Query Editor (M)

Data Source Connections

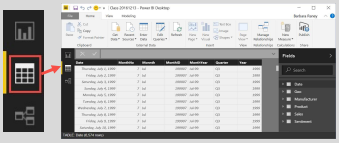
Data Transformations

(Prep data for Data Model)



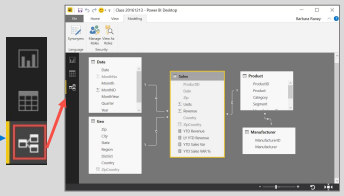
Report

Create Visuals



Data (DAX)

View Tables



Relationships

See how Tables relate to each other

What is a Data model?

Power BI

A Power BI **Data Model** is a **collection of tables with relationships** which enable your business users to easily understand and explore their data to get business insights.

Why is it important to have a Good Data model?

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

Components of a data model – Fact Table

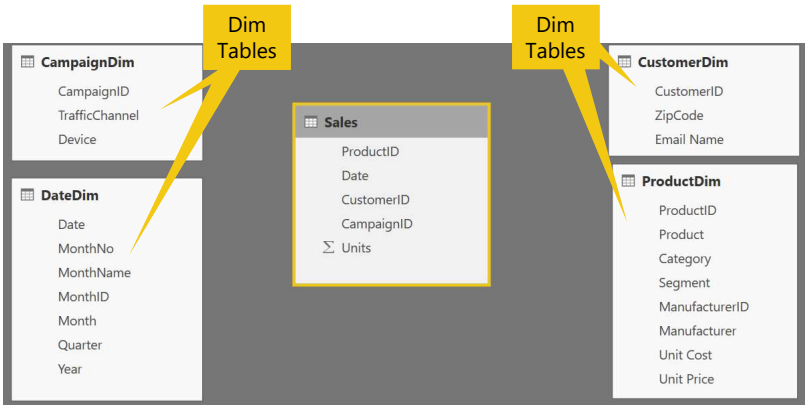
Power BI

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

Components of a data model – Dim Table

Power BI



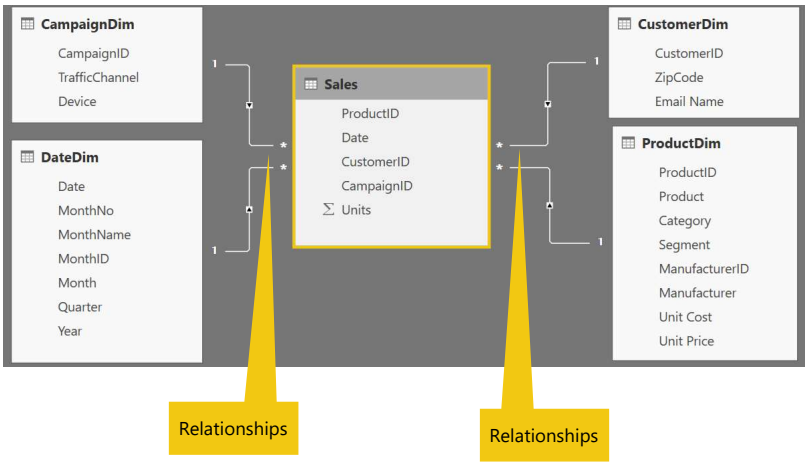
Dim Table

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

Examples:
By month, By Customer, By Geo

Components of a data model - Relationships

Power BI



Relationships

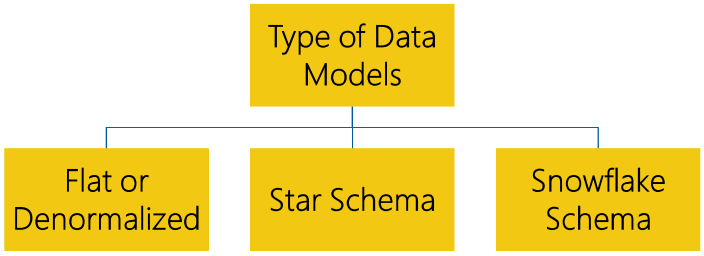
- Connection between a 2 tables (usually fact & Dim tables) using columns from each

- 3 kinds of Relationships
  - 1 to Many
  - 1 to 1
  - Many to Many

(with a bridge table)

Data Model Brings Facts and Dimensions Together

Power BI



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

Flat or Denormalized Schema

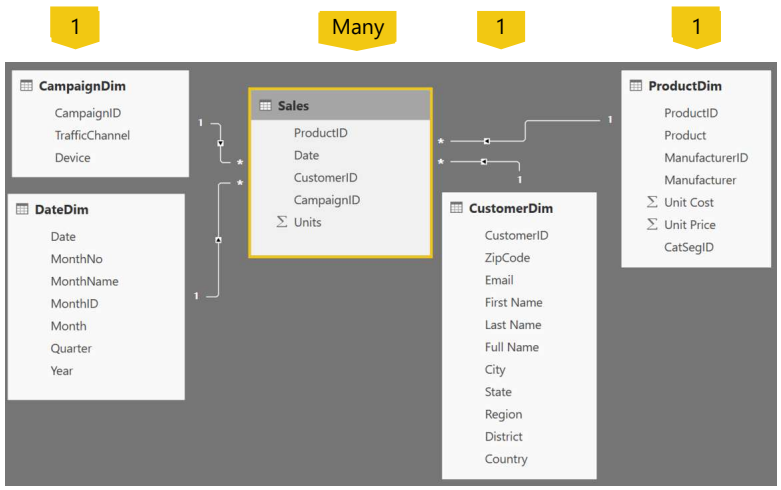
Power BI

	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	27183	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	2/26/2013	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

Star Schema

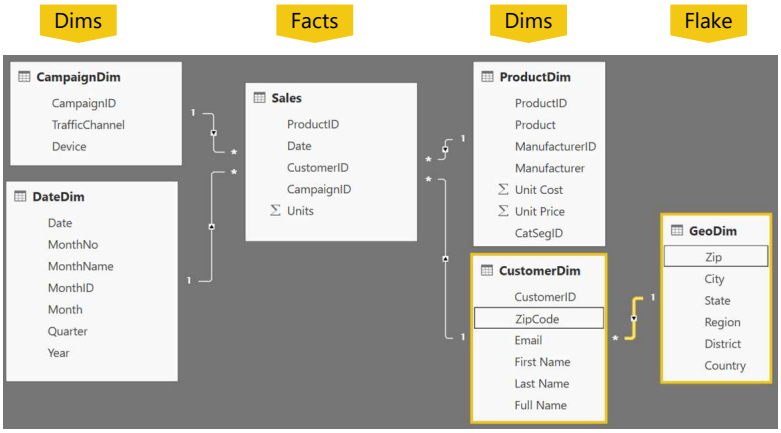
Power BI



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

Snowflake Schema

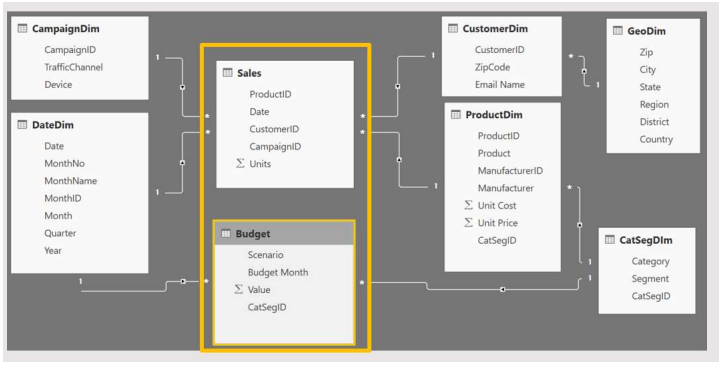
Power BI



- 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

Granularity & Multiple Fact Tables

Power BI



Sales (Daily by Product)

Budget (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 Mode Types in Power BI

Power BI

How can I tell what Data Model Type I have?

- Live Connect to SQL Analysis Services (SSAS) tabular
  - Report view only available

- DirectQuery to SQL or other relational source
  - Report & Relationship views available

- Import data into Power BI (creates a copy of the data)
  - Report, Data and Relationship views available

Connection: Live Connect

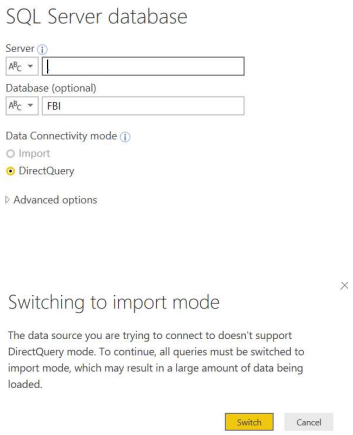
Power BI

- Live Connect to Multidimensional or Tabular
  - On Premise or Azure
- Only a single connection will be made and all modeling is done in the cube
- You can not add relationships or additional data source
- If allowed, you can add DAX measures

Connection: DirectQuery to Relational Source

Power BI

- Direct Query to SQL or other relational source
  - On Premise or Azure
- All Data Sources are required to be DirectQuery, you cannot "Mashup" with Import sources or you get this message
- You can add relationships and DAX



Import Mode

Power BI

What is unique about Power BI Desktop in Import Mode?

- Columnar database
- In-memory database

Let us understand some of the internals of Power BI Desktop !!

Columnar Database

Power BI

Row Based Database

First Name	Last Name	Sales
John	Smith	$10
Jane	Doe	$25
Hardy	B	$35

- Stores **each row separately** (like a separate file)
- Retrieving multiple columns from a single row is fast
- Retrieving multiple rows from a single column is slower

PBI - Columnar Database

First Name	Last Name	Sales
John	Smith	$10
Jane	Doe	$25
Hardy	B	$35

- Stores **each column separately** (like a separate file)
- Retrieving multiple columns from a single row is slow
- Retrieving multiple rows from a single column is faster
- **Columnar databases are well suited for analytics**

In-Memory Database

Power BI

PBI – In-Memory Database

- Data stored in **RAM (in memory)**
- RAM is all electronic – **Read/Write is fast**
- Laptops have smaller **RAM space (~8GB)**

Power BI compresses data to conserve space in RAM

Compressing Data – Dictionary Encoding

Power BI

How Power BI Compresses Data – Dictionary Encoding

Sale Id	Color	Sales Amount
390a30e0-dc37	Red	$10
390a30e1-dc37	Green	$25
390a30e2-dc37	Red	$35
390a30e3-dc37	Red	$15
390a30e4-dc37	Red	$25
390a30e5-dc37	Green	$30
390a30e6-dc37	Blue	$10
390a30e7-dc37	Blue	$12
390a30e8-dc37	Blue	$15
390a57f0-dc37	Blue	$18
390a57f1-dc37	Green	$25

Red = 1 Green = 2 Blue = 3

- Create a Dictionary to create an integer value for text string
- Storing 1,2,3 instead of "Red", "Green", "Blue" saves memory
- **Dictionary encoding is powerful when there are few unique values** in a column
  - Ex. Color column – Good for dictionary encoding
  - Ex. Sale ID – Bad for dictionary encoding

Compressing Data – Run Length Encoding

Power BI

How Power BI Compresses Data – Run Length Encoding

Sale Id	Color	Sales Amount	
390a30e0-dc37	Red	$10	1
390a30e1-dc37	Green	$25	2
390a30e2-dc37	Red	$35	1
390a30e3-dc37	Red	$15	1
390a30e4-dc37	Red	$25	1
390a30e5-dc37	Green	$30	2
390a30e6-dc37	Blue	$10	3
390a30e7-dc37	Blue	$12	3
390a30e8-dc37	Blue	$15	3
390a57f0-dc37	Blue	$18	3
390a57f1-dc37	Green	$25	2

Run Length Encoding in Power BI

Where Red = 1 Green = 2 Blue = 3

- Instead of storing - 1, 2, 1, 1, 1, 2, 3, 3, 3, 3, 2
- It Stores:

1 – 1	(1 instance of One)
1 – 2	(1 instance of Two)
3 – 1	(3 instances of One)
1 – 2	(1 instance of Two)
4 – 3	(4 instances of Three)
1 - 2	(1 instance of Two)

- **Run length encoding** is **very powerful when data is sorted** well and has few unique values

Compression

Power BI

Practical Example of Compression

Dashboard in a Day Class Data

Sales Fact	420.0 MB
Dimensions	4.4 MB
Int'l Sales	32.4 MB
Total Data	**456.8 MB**

Queries ONLY – No Data Loaded

Query Metadata	**113 KB**

DIAD Complete Data Model

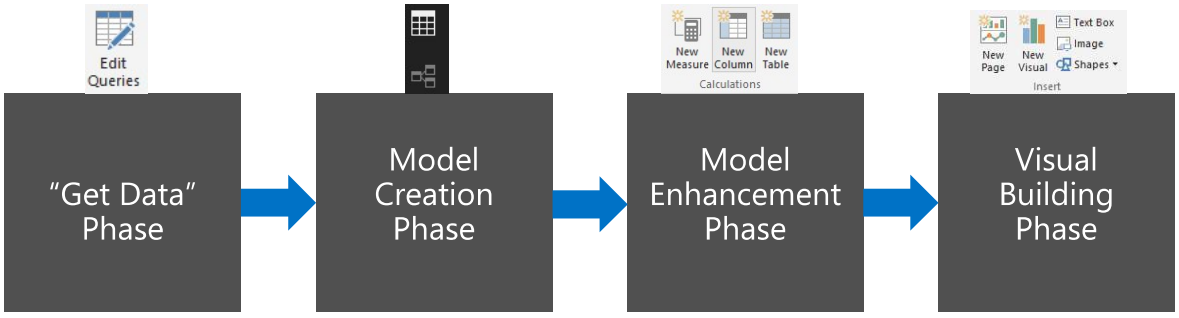
Data Model	**59.4 MB**

Almost 8X Compression!!

Power BI Desktop Files

Power BI

Phases in Building a Power BI Desktop File



Create Query in "M"

Compress data, auto detect relationships

(Automatically done by Tool)

Add calculated columns, measures, add missing relationships

Evaluate Measures and build each visual

Designing good data models

Power BI

Key takeaways to design a good Power BI Desktop data model

- **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
- The DateTime data type is usually not needed, unless you are specifically using the Time component
  - If you really need Time, **try splitting Date & Time** into two columns - Reduces # of unique values

Star Schema – Good for most Data Models

Data Types

Power BI

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*

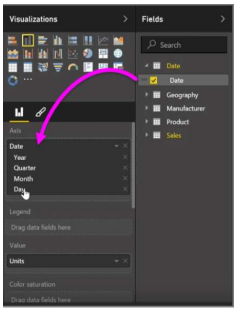
Set your ***Data Formats*** *($ %, etc) in the Data Model*

Pro Tip: Data type is different from data format

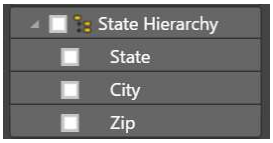
Hierarchies

Power BI

- Power BI generates Date hierarchies when dates are added to visuals, this allows the end user to drill from Year, Quarter, Month & Day.



- Users can also create custom hierarchies in the model by dragging a lower level field onto the parent.

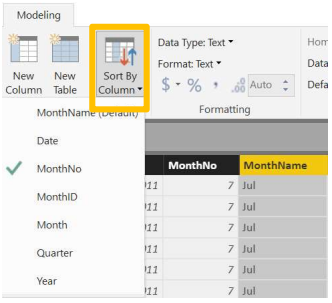


Sort By Column

Power BI

- Enables sorting one text field by another (numeric) field

- Both columns must have the same number of distinct values



Module 1 Lab

1. Open up the file **Student Modeling Pre-class.pbix**
2. Create the relationships between the tables!
   HINT: You may need to preview some of the tables to see what is in them

   Think about: What sort of data model are you creating?

KNOWLEDGE CHECK Module 1

Power BI

1. What is a *data model* in the context of Power BI?
2. What are some advantages of a star schema over a flat or denormalized model?
3. How might you improve the performance of a Power BI model?
4. How does Power BI store DateTime information? What are some consequences of this?

Module 2
DAX Calculated Columns & Measures

MODULE 2 OBJECTIVE

Power BI

- Understand differences between calculated columns and measures (uses, evaluation, performance, etc.)

34

DAX Level Set

Power BI

- DAX looks similar to Excel functions but they have key differences
- DAX is a very deep and elegant…
- This class provides a solid base in DAX, but don't expect to leave being able to write the most complex DAX patterns--they take practice.

35

DAX Foundations

Power BI

Path to DAX Expertise

Evaluation Contexts

CALCULATE

Calculated Columns and Measures

DAX Foundations

Power BI

Calculated Column

Measure

What is a Calculated Column?

Power BI

```
Price Band = If(ProductDim[Unit Price] <=25, "Low",If(ProductDim[Unit Price] <=50, "Medium", "High"))
```

ProductID	Product	Category	Segment	ManufacturerID	Manufacturer	Unit Cost	Unit Price	Price Band
577	Maximus UC-42	Urban	Convenience	7	VanArsdel	74.73	102.37	High
578	Maximus UC-43	Urban	Convenience	7	VanArsdel	57.48	78.74	High
579	Maximus UC-44	Urban	Convenience	7	VanArsdel	96.96	132.82	High
580	Maximus UC-45	Urban	Convenience	7	VanArsdel	60.92	83.45	High
581	Maximus UC-46	Urban	Convenience	7	VanArsdel	101.54	139.10	High
582	Maximus UC-47	Urban	Convenience	7	VanArsdel	26.06	35.69	Medium
583	Maximus UC-48	Urban	Convenience	7	VanArsdel	40.18	55.05	High
584	Maximus UC-49	Urban	Convenience	7	VanArsdel	45.22	61.94	High

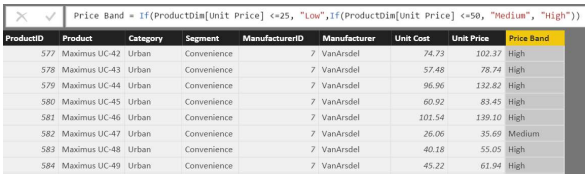
Calculated Column

Pro Tip: Always refer to a calculated column by its full name -> **TableName[ColumnName]**

Calculated Column

Power BI

Calculated Column in DAX



ProductID	Product	Category	Segment	ManufacturerID	Manufacturer	Unit Cost	Unit Price	Price Band
577	Maximus UC-42	Urban	Convenience	7	VanArsdel	74.73	102.37	High
578	Maximus UC-43	Urban	Convenience	7	VanArsdel	57.48	78.74	High
579	Maximus UC-44	Urban	Convenience	7	VanArsdel	96.96	132.82	High
580	Maximus UC-45	Urban	Convenience	7	VanArsdel	60.92	83.45	High
581	Maximus UC-46	Urban	Convenience	7	VanArsdel	101.54	139.10	High
582	Maximus UC-47	Urban	Convenience	7	VanArsdel	26.06	35.69	Medium
583	Maximus UC-48	Urban	Convenience	7	VanArsdel	40.18	55.05	High
584	Maximus UC-49	Urban	Convenience	7	VanArsdel	45.22	61.94	High

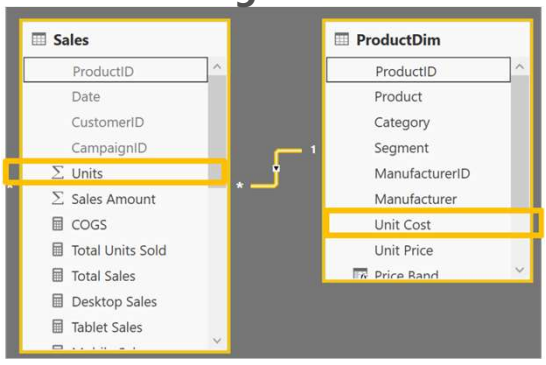
Custom Column in "Query Editor"

Note: If given a choice, creating the column in "M" or "Query Editor" will give you better compression.

Calculated Column

Power BI

Calculated Column – Accessing columns from other Tables in model



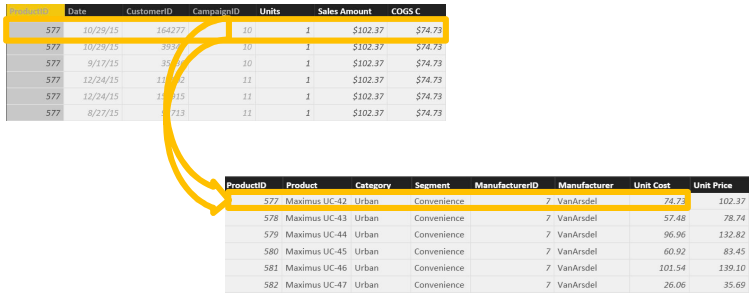
- **Often you want to access columns from multiple tables to create a Calculated Columns**
- **Let us say you want to calculate COGs, which is Units * Units Cost**
- **Units Cost is in another Table**

RELATED Function

Power BI

Row Context and Multiple Tables – RELATED Function

```
Sales[COGS] = RELATED(ProductDim[Unit Cost]) * Sales[Units]
```



ProductID	Date	CustomerID	CampaignID	Units	Sales Amount	COGS C
577	10/29/15	164277	10	1	$102.37	$74.73
577	10/29/15	3984	10	1	$102.37	$74.73
577	9/17/15	35	10	1	$102.37	$74.73
577	12/24/15	11	11	1	$102.37	$74.73
577	12/24/15	15	11	1	$102.37	$74.73
577	8/27/15	3	11	1	$102.37	$74.73

ProductID	Product	Category	Segment	ManufacturerID	Manufacturer	Unit Cost	Unit Price
577	Maximus UC-42	Urban	Convenience	7	VanArsdel	74.73	102.37
578	Maximus UC-43	Urban	Convenience	7	VanArsdel	57.48	78.74
579	Maximus UC-44	Urban	Convenience	7	VanArsdel	96.96	132.82
580	Maximus UC-45	Urban	Convenience	7	VanArsdel	60.92	83.45
581	Maximus UC-46	Urban	Convenience	7	VanArsdel	101.54	139.10
582	Maximus UC-47	Urban	Convenience	7	VanArsdel	26.06	35.69

- **RELATED is just like VLOOKUP in Excel**

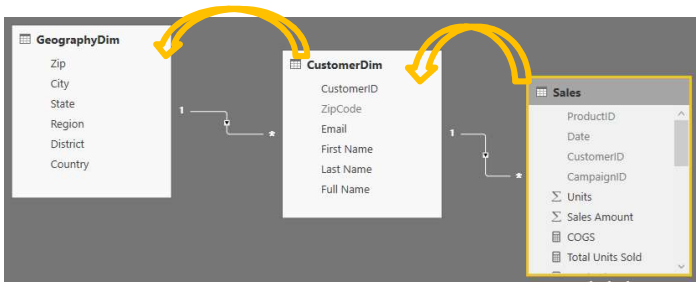
RELATED Function

Power BI

RELATED Function Example

- You *could* follow a chain of relationship from Many side to 1 side using RELATED

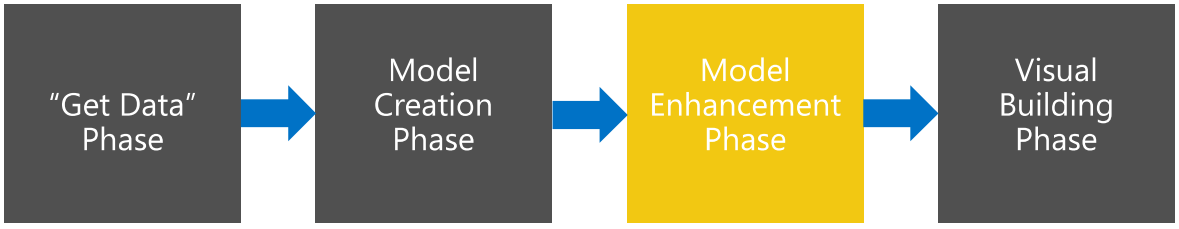
```
Sales [City State]= RELATED(GeographyDim[City]) & ", " & RELATED(GeographyDim[State])
```



DAX Foundations

Power BI

When is a Calculated Column Evaluated?



- Create Query in "M"
- Compress data
- Auto detect relationships
- Add calc. columns, Measures
- Add missing relationships
- Evaluate Measures and build each visual

Best Practices – Calculated Columns

Power BI

Best Practices with DAX Calculated Columns

- Whenever possible, DAX helper columns should be avoided. Each "Helper Column" will consume RAM
- Create a calculated column in the Dim Table as opposed to in the Fact Table
- Move calculated columns to "M" if you can

DAX Foundations

Power BI

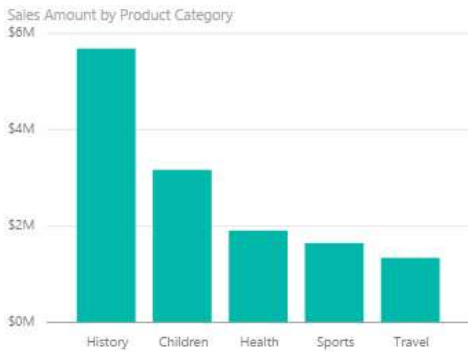
Calculated Column

Measure

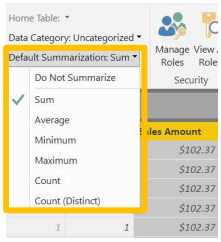
Default Summarization

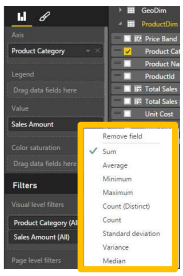
Power BI

What is a Default Summarization?



On the Modeling ribbon





For an individual visualization

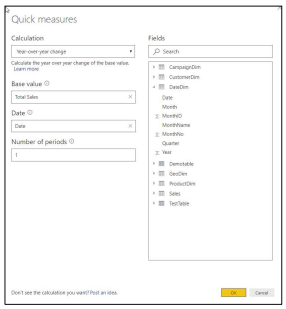
- Sales Amount is automatically summed for each category
- You should set this summarization for all numeric fields in the **Modeling** ribbon

Quick Measures

Power BI

Quick Measures are wizard driven DAX calculations

- Right+Click a table and select **Quick Measures**
  - Select calculation type and fill-in parameters
  - DAX is generated automatically
  - Great way to learn DAX




```
Total Sales YoY% =
IF(
    ISFILTERED('DateDim'[Date]),
    ERROR("Time intelligence quick measures can only be grouped or filtered by the Power BI-provided date hierarchy or primary date
column."),
    VAR __PREV_YEAR = CALCULATE([Total Sales], DATEADD('DateDim'[Date].[Date], -1, YEAR))
    RETURN
        DIVIDE([Total Sales] - __PREV_YEAR, __PREV_YEAR)
)
```

Category	Total Sales	Total Sales YoY%
Urban	$54,427,851	9.98%
Accessory	$5,991,334	9.06%
Mix	$3,853,181	14.15%
Youth	$1,268,274	3.37%
Rural	$6,500	38.43%
Total	**$65,547,141**	**10.00%**

See Quick Measure gallery: https://community.powerbi.com/t5/Quick-Measures-Gallery/bd-p/QuickMeasuresGallery

Measures

Power BI

What is a Measure?

ProductID	Date	CustomerID	CampaignID	Units	Sales Amount
666	2/24/12	58642	3	1	$81.37
666	2/25/12	208515	3	1	$81.37
666	7/12/12	164032	3	1	$81.37
666	7/12/12	243676	3	1	$81.37
406	6/12/16	31036	16	1	$191.62
406	6/17/16	44688	16	1	$191.62
406	6/17/16	108991	16	1	$191.62

[Total Sales]=SUM(Sales[Sales Amount])

- Measures are created using DAX
- Place your Measures on a Fact table for best results

Pro Tip: When referring to a measure in other calculations, refer to it without a Table name: **[MeasureName]**

Measures

Power BI

Measure, Use Case 1: Using One Measure in Another

Instead of writing this:

```
[Profit] = SUM(Sales[Sales Amount])-SUM(Sales[COGS])
```

Write this:

```
[Profit] = [Total Sales]- [Total COGS]
```

- Allows re-use of measures
- Formulas are much simpler to read

Measures

Power BI

Measure, Use Case 2: More Complex Calculations

```
[Profit Margin %] = [Profit] / [Total Sales]
```

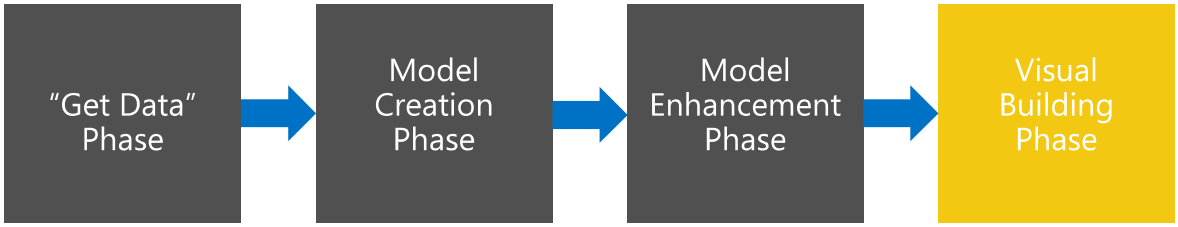
- Ratios are calculations that cannot be created using a Calculated Column or Default Summarization
- Use DAX **DIVIDE** for built in error handling

```
[Profit Margin %] = DIVIDE([Profit] , [Total Sales])
```

DAX Foundations

Power BI

When is a Measure Evaluated?

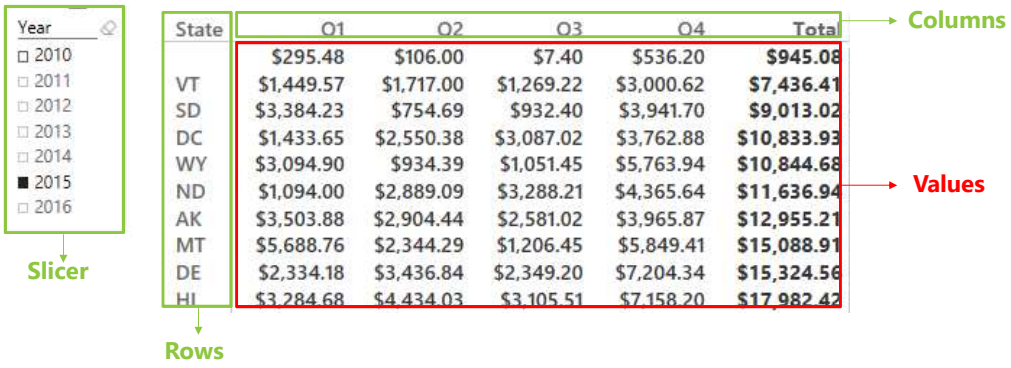


- Create Query in "M"
- Compress data
- auto detect relationships
- Add calc. columns, Measures
- Add missing relationships
- Evaluate Measures and build each visual

Calculated Column vs. Measure

Power BI

Calculated Column vs. Measure - When to Use What



Rule of Thumb for Calculated Column vs Measure

- **Calculated Column** – Use in Page, Report & Visual Filters as well as Slicers, Rows and Columns
- **Measures** - Use in Values section

Module 2 Lab

1. Create a MEASURE for Total Units Sold
   HINT: The formula will probably use SUM()

2. Create a CALCULATED COLUMN on the fact table that shows product category and campaign traffic channel combined
   Example: Urban, Organic Search

3. It is fairly easy to see that the CALCULATED COLUMN is working. Create some visuals that allow you to confirm that the Total Units Sold MEASURE is working right

KNOWLEDGE CHECK Module 2

Power BI

- When is Calculated Column Evaluated?
- What is Default Summarization?
- When is a Measure Evaluated?
- When to use Measures and Calculated Columns?

Module 3
CALCULATE

MODULE 3 OBJECTIVE

Power BI

- Understand the basics of the CALCULATE formula

DAX Foundations

Power BI

PATH to DAX Expertise

Evaluation Contexts

CALCULATE

Calculated Columns and Measures

CALCULATE

Power BI

Why is CALCULATE Useful?

You create a report of breakdown of Sales by Month

Typical Business Question:
Provide a break out of this Sales from Desktop

Month	Total Sales
January	$3,379,202
February	$4,434,793
March	$7,848,903
April	$8,175,811
May	$8,133,443
June	$7,847,091
July	$5,736,090
August	$5,739,110
September	$4,755,394
October	$3,746,354
November	$2,968,954
December	$2,781,997
Total	**$65,547,141**

Month	Total Sales	Desktop Sales
January	$3,379,202	$1,451,782
February	$4,434,793	$1,647,766
March	$7,848,903	$2,619,290
April	$8,175,811	$2,540,481
May	$8,133,443	$2,699,799
June	$7,847,091	$2,381,357
July	$5,736,090	$2,243,771
August	$5,739,110	$2,043,244
September	$4,755,394	$1,633,458
October	$3,746,354	$1,203,403
November	$2,968,954	$866,860
December	$2,781,997	$884,017
Total	**$65,547,141**	**$22,215,229**

CALCULATE

Power BI

Here is how you do it with CALCULATE

[Desktop Sales] = CALCULATE([Total Sales], CampaignDim[Device] = "Desktop")

- Use CALCULATE function to create a Measure which filters down to Desktop Sales

Month	Total Sales	Desktop Sales
January	$3,379,202	$1,451,782
February	$4,434,793	$1,647,766
March	$7,848,903	$2,619,290
April	$8,175,811	$2,540,481
May	$8,133,443	$2,699,799
June	$7,847,091	$2,381,357
July	$5,736,090	$2,243,771
August	$5,739,110	$2,043,244
September	$4,755,394	$1,633,458
October	$3,746,354	$1,203,403
November	$2,968,954	$866,860
December	$2,781,997	$884,017
Total	**$65,547,141**	**$22,215,229**

CALCULATE

Power BI

Anatomy of CALCULATE

CALCULATE(Expression, [Filter 1], [Filter 2].....)

Filter Arguments

- EXPRESSION used as the first parameter is essentially the same as a measure
- CALCULATE works differently from other DAX functions
- The second set of arguments, i.e. the "Filter arguments," are evaluated and applied first
- Then the Expression is evaluated under new "Filter Context"

CALCULATE – Add Filter

Power BI

CALCULATE – The Most Important Function in DAX

4 Key functions that CALCULATE can do:

- **Add Filter**
- Ignore Filter
- Update Filter
- Convert Row Context to Filter Context

CALCULATE – Add Filter

Power BI

[Desktop Sales] = CALCULATE([Total Sales], CampaignDim[Device] = "Desktop")

[Tablet Sales] = CALCULATE([Total Sales], CampaignDim[Device] = "Tablet")

[Mobile Sales] = CALCULATE([Total Sales], CampaignDim[Device] = "Mobile")

Month	Total Sales	Desktop Sales	Tablet Sales	Mobile Sales
January	$617,594	$248,081	$113,385	$256,128
February	$846,436	$300,692	$278,821	$266,922
March	$1,382,885	$492,987	$223,870	$334,252
April	$1,512,488	$461,759	$620,238	$404,458
May	$1,589,728	$558,984	$368,121	$511,447
June	$1,402,897	$433,576	$459,494	$313,134
July	$1,122,721	$430,424	$316,463	$375,833
August	$1,222,190	$501,972	$312,637	$404,067
September	$865,028	$308,490	$304,430	$244,142
October	$712,729	$232,041	$246,786	$203,002
November	$562,400	$192,873	$171,329	$169,693
December	$467,428	$148,821	$162,990	$144,679
Total	**$12,304,523**	**$4,310,700**	**$3,578,565**	**$3,627,759**

Year

- ☐ 2011
- ☐ 2012
- ☐ 2013
- ☐ 2014
- ■ 2015
- ☐ 2016

*When the Device Slicer is selected, only "Total Sales" changes.

CALCULATE – Ignore Filter

Power BI

CALCULATE – The Most Important Function in DAX

4 Key functions that CALCULATE can do

Add Filter	Ignore Filter	Update Filter	Convert Row Context to Filter Context

CALCULATE – Ignore an Existing Filter

Power BI

[Total Sales All Geo] = CALCULATE([Total Sales], ALL(GeographyDim))

State	Total Sales	Total Sales All Geo
UT	$482,268	$65,547,141
VA	$1,609,751	$65,547,141
VT	$42,233	$65,547,141
WA	$1,336,132	$65,547,141
WI	$2,297,199	$65,547,141
WV	$599,850	$65,547,141
WY	$351,374	$65,547,141
Total	**$65,547,141**	**$65,547,141**

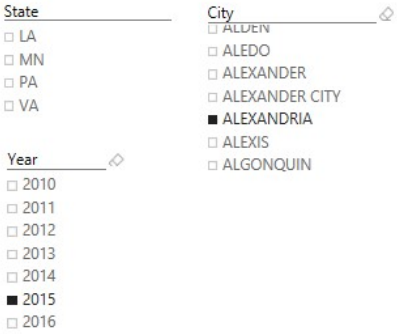
*Ignore filter on ANY column from the GeographyDim table, but allows filters from Year

CALCULATE – Ignore an Existing Filter

Power BI

[Total Sales All States] = CALCULATE([Total Sales], ALL(GeographyDim[State]))

State	Total Sales	Total Sales All Geo	Total Sales All States
AL	$206	$12,304,523	$15,387
IN	$710	$12,304,523	$15,387
KY	$702	$12,304,523	$15,387
LA	$3,343	$12,304,523	$15,387
MN	$2,545	$12,304,523	$15,387
MO		$12,304,523	$15,387
NE		$12,304,523	$15,387
OH		$12,304,523	$15,387
PA	$283	$12,304,523	$15,387
SD		$12,304,523	$15,387
TN	$144	$12,304,523	$15,387
VA	$7,455	$12,304,523	$15,387
Total	**$15,387**	**$12,304,523**	**$15,387**



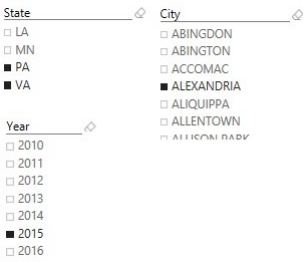
*Ignore filter on the STATE column from the GeographyDim table, but allows filters from Year

CALCULATE – Ignore Existing Filter

Power BI

[Total Sales All Selected States] = CALCULATE([Total Sales], ALLSELECTED(GeographyDim[State]))

State	Total Sales	Total Sales All Geo	Total Sales All States	Total Sales All Selected States
PA	$283	$12,304,523	$15,387	$7,737
VA	$7,455	$12,304,523	$15,387	$7,737
Total	**$7,737**	**$12,304,523**	**$15,387**	**$7,737**



*Ignore filter on the STATE column from the GeographyDim table, but allows filters from Year

CALCULATE – Update Filter

Power BI

CALCULATE – The Most Important Function in DAX

4 Key functions that CALCULATE can do

Add Filter	Ignore Filter	Update Filter	Convert Row Context to Filter Context

CALCULATE – Update Existing Filter

Power BI

[2014 Sales] = CALCULATE([Total Sales], DateDim[Year] = 2014)

Month	Total Sales	2014 Sales
January	$617,594	$624,956
February	$846,436	$817,549
March	$1,382,885	$1,245,627
April	$1,512,488	$1,400,954
May	$1,589,728	$1,510,563
June	$1,402,897	$1,481,390
July	$1,122,721	$1,281,466
August	$1,222,190	$1,273,948
September	$865,028	$1,201,762
October	$712,729	$916,774
November	$562,400	$714,021
December	$467,428	$575,281
Total	**$12,304,523**	**$13,044,290**

Year
- ☐ 2010
- ☐ 2011
- ☐ 2012
- ☐ 2013
- ☐ 2014
- ■ 2015
- ☐ 2016

*Ignores filter on the Year Slicer

CALCULATE – Convert Row Context to Filter Context

Power BI

CALCULATE – The Most Important Function in DAX

4 Key functions that CALCULATE can do

Add Filter	Ignore Filter	Update Filter	Convert Row Context to Filter Context

Let's investigate what we mean by **Filter Context**

Module 4
DAX Evaluation Contexts

MODULE 4 OBJECTIVES

Power BI

- Understand that there are different kinds of evaluation contexts and be able to explain what different contexts are in play
- Be able to use iterator functions and CALCULATE to create sophisticated measures

71

DAX Foundations

Power BI

PATH to DAX Expertise

Evaluation Contexts

CALCULATE

Calculated Columns and Measures

Evaluation Context

Power BI

There are two contexts under which calculations are evaluated

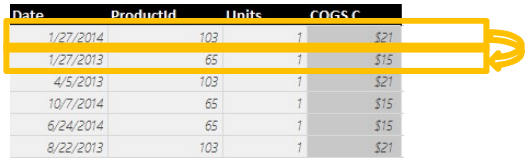
Row Context

Filter Context

Row context in Calculated Column

Power BI

Sales[COGS] = RELATED(ProductDim[Unit Cost]) * Sales[Units]



Date	ProductId	Units	COGS C
1/27/2014	103	1	$21
1/27/2013	65	1	$15
4/5/2013	103	1	$21
10/7/2014	65	1	$15
6/24/2014	65	1	$15
8/22/2013	103	1	$21

- Formula is evaluated row by row
- The context under which formula is evaluated for each row is called “Row Context”

Pro Tip: To accumulate up from Fact to Dimension, use **RELATEDTABLE()**

Evaluation Context

Power BI

Both Calculated Columns and Measures are always evaluated under two contexts

Row Context

Filter Context

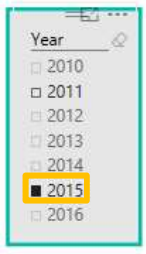
Filter Context in Measures

Power BI

Filter Context in a Measure – Example 1

[Total Sales] = SUM(Sales[Sales Amount])

Filter Context for current coordinate Year = 2015, State = **HI**, Quarter = **Q1**



State	Q1	Q2	Q3	Q4	Total
	$295.48	$106.00	$7.40	$536.20	$945.08
VT	$1,449.57	$1,717.00	$1,269.22	$3,000.62	$7,436.41
SD	$3,384.23	$754.69	$932.40	$3,941.70	$9,013.02
DC	$1,433.65	$2,550.38	$3,087.02	$3,762.88	$10,833.93
WY	$3,094.90	$934.39	$1,051.45	$5,763.94	$10,844.68
ND	$1,094.00	$2,889.09	$3,288.21	$4,365.64	$11,636.94
AK	$3,503.88	$2,904.44	$2,581.02	$3,965.87	$12,955.21
MT	$5,688.76	$2,344.29	$1,206.45	$5,849.41	$15,088.91
DE	$2,334.18	$3,436.84	$2,349.20	$7,204.34	$15,324.56
HI	$3,284.68	$4,434.03	$3,105.51	$7,158.20	$17,982.42

- Formula is evaluated for each "Coordinate" in each visual
- The context for each coordinate is called "Filter Context"

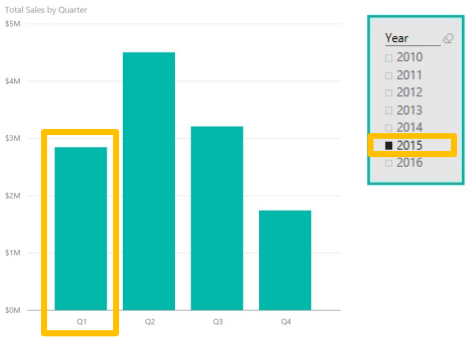
Filter Context in a Measure

Power BI

Filter Context in a Measure – Example 2

[Total Sales] = SUM(Sales[Sales Amount])

Filter Context : Year = 2015, Quarter = **Q1**



Filter Context : Year = 2015, Quarter = **Q2**

Filter Context in a Measure

Power BI

Filter Context in a Measure

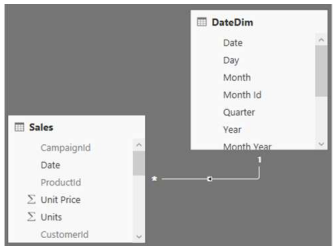
[Total Sales] = SUM(Sales[Sales Amount])

Better definition of above measure:

"Total Sales" – SUM of Sales[Sales Amount] column **under a filter context**

Filter Context and Multiple Tables

Power BI

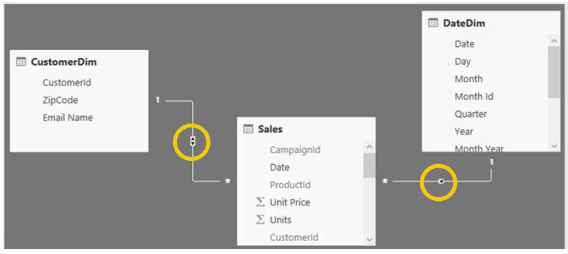


- Filter context automatically propagates from Dim Table to Fact Table
- Filtering the DateDim Table to Year = 2015 returns only Sales for 2015

Filter Context and Multiple Tables

Power BI

Filter Context and Multiple Tables

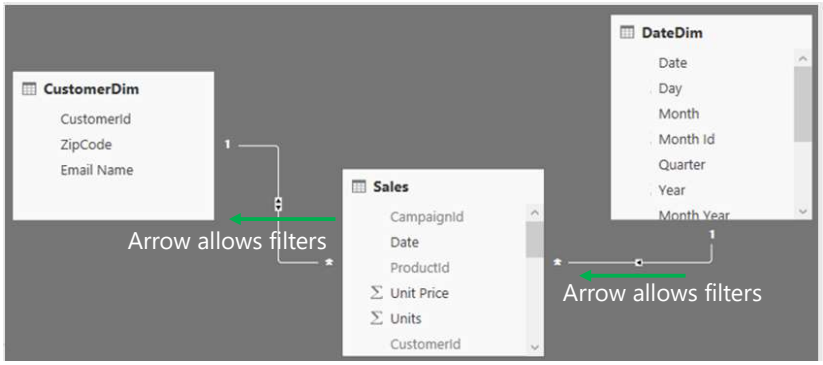


- Filters (Filter context) automatically propagate based on direction of arrows in relationships
- Examples
  - Filter goes from DateDim to CustomerDim
  - Filter does not go from CustomerDim to DateDim

Filter Context and Multiple Tables

Power BI

Filter Context and Multiple Tables – Right Arrow Direction



Cross filtering works properly

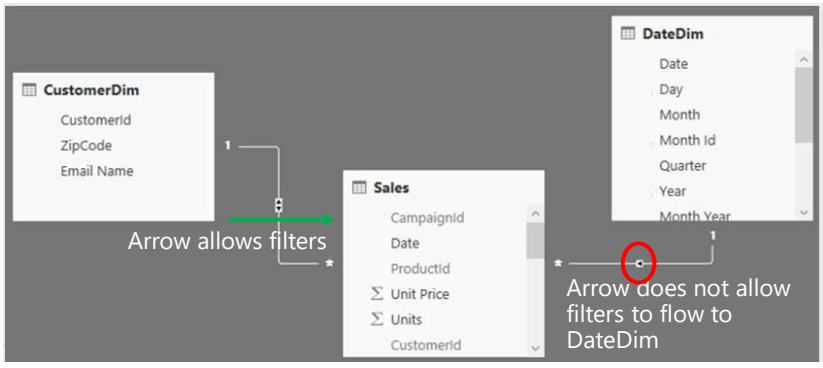
Month	Total Sales M	Count of CustomerId
Jan	$1,673,394.03	7132
Feb	$431,531.13	2820
Mar	$690,671.10	4017
Apr	$852,018.76	4629
May	$972,018.47	5185
Jun	$907,703.04	4854
Jul	$608,678.35	3680
Aug	$1,355,530.22	6242
Sep	$720,851.83	4186
Oct	$1,117,087.73	5728
Nov	$2,372,763.71	8242
Dec	$2,003,261.11	7683
Total	**$13,705,509.48**	**10000**

- Filter goes from DateDim to CustomerDim
- This is why the above Pivot table works

Filter Context and Multiple Tables

Power BI

Filter Context and Multiple Tables – Wrong Arrow Direction



Cross filtering does not work

CustomerId	Total Sales M	Count of Month
	$1,985.76	12
00001	$438.34	12
00002	$840.08	12
00003	$1,246.69	12
00004	$706.23	12
00005	$1,653.97	12
00006	$2,170.10	12
00007	$2,308.44	12
00008	$1,517.34	12
00009	$1,184.11	12
00010	$2,221.02	12
00011	$1,646.48	12
Total	**$13,705,509.48**	**12**

- Filter goes from DateDim to CustomerDim
- This is why the Count of Month in the table above is incorrect

Evaluation Context and Multiple Tables

Power BI

Evaluation Context Multiple Table – Summary and Take Aways

Row Context

- Does not propagate automatically
- Need to use
  RELATED
  RELATEDTABLE

Filter Context

- Propagates automatically
- Depends on direction of arrow in relationship diagram

DAX Function Types

Power BI

Scalar Functions

- Scalar functions return a Single value as an output
- Ex. SUM(Sale[Sales Amount])

Table Functions

- Table functions return a Table as an output
- Ex. ALL(GeographyDim)

There are other ways to classify functions – By kind of operation they perform etc.

Applications of Table functions

Power BI

Table functions can be used 2 ways in Power BI Desktop

- As an input to another DAX function
  - CALCULATE
  - Iterator functions
- Calculated Tables

Basic TABLE functions

Power BI

Return All Rows	Return Distinct Rows	Return Filtered Rows
ALL & variants	ALL, DISTINCT, VALUES	FILTER

There are more advanced Table functions, which we will not cover

Basic Table functions – Return All Rows

Power BI

- The **ALL** function - Can take either Table or Columns in a Table as input

ALL with Entire Table	ALL with One Column	ALL with Multiple Columns
ALL(GeographyDim)	**ALL(GeographyDim[Region]))**	**ALL(GeographyDim[Region], GeographyDim[State])**
Returns all rows all columns in Table	Returns all unique values of Column	Returns all unique combinations of Column values

Basic Table functions – ALL versions

Power BI

- There are several forms of the ALL function
  - **ALL**
  - **ALLEXCEPT** - Return all columns in a Table except 1 or more columns
  - **ALLSELECTED** – Return all values in a column selected by users in Slicers
  - **ALLNONBLANKROW** – Return all non-Blank rows

Basic Table Functions – Return Distinct Rows

Power BI

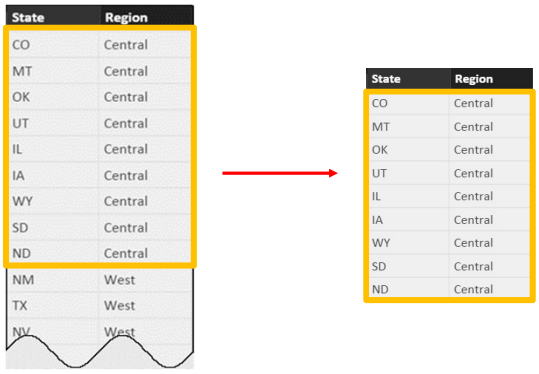
- **VALUES –** Return all distinct values in a column or Table **(including blank rows)**
- **DISTINCT** - Return all distinct values in a column or Table **(not including blank rows)**

Basic Table Functions – Return Filtered Set of Rows

Power BI

FILTER(ALL(GeographyDim[Region], GeographyDim[State]), GeographyDim[Region] = "Central")

- Take all unique combinations of GeographyDim[Region], GeographyDim[State]
- Filter down to the rows where GeographyDim[Region] = "Central"



State	Region
CO	Central
MT	Central
OK	Central
UT	Central
IL	Central
IA	Central
WY	Central
SD	Central
ND	Central
NM	West
TX	West
NV	West

State	Region
CO	Central
MT	Central
OK	Central
UT	Central
IL	Central
IA	Central
WY	Central
SD	Central
ND	Central

DAX Iterator Functions

Power BI

DAX Iterator Functions Take Advantage of Evaluation Context

Iterator Functions

- Creates a *row context* by iterating over a table that you specify
- Ex. SUMX

Table Functions Application – Iterators

Power BI

```
[COGS] = SUMX(Sales, Sales[Units] * RELATED(ProductDim[Unit Cost]))
```

Argument 1 **Argument 2**

Table Functions Application – Iterators

Power BI

[COGS] = SUMX(Sales, Sales[Units] * RELATED(ProductDim[Unit Cost]))

Argument 1

Date	ProductId	Units
1/27/2014	103	1
1/27/2013	65	1
4/5/2013	103	1
10/7/2014	65	1
6/24/2014	65	1
8/22/2013	103	1
11/8/2013	65	1
3/27/2015	103	1
5/26/2013	103	1
12/23/2014	103	1
1/28/2015	103	1
5/21/2015	65	1
7/5/2015	103	1
7/19/2015	65	1

Iterate through each row in Argument 1

Sales

Table Functions Application – Iterators

Power BI

[COGS] = SUMX(Sales, Sales[Units] * RELATED(ProductDim[Unit Cost]))

Argument 2

ProductID	Date	CustomerID	CampaignID	Units
449	7/29/14	128304	1	1
449	7/29/14	89917	1	1
449	7/29/14	128811	1	1
449	7/29/14	59550	1	1
449	7/29/14	207690	1	1
449	7/29/14	121043	1	1

Sales

ProductID	Product	Category	Segment	ManufacturerID	Manufacturer	Unit Cost
446	Maximus UM-50	Urban	Moderation	7	VanArsdel	62.15
447	Maximus UM-52	Urban	Moderation	7	VanArsdel	109.59
449	Maximus UM-54	Urban	Moderation	7	VanArsdel	74.73
450	Maximus UM-55	Urban	Moderation	7	VanArsdel	125.32
451	Maximus UM-56	Urban	Moderation	7	VanArsdel	66.49
452	Maximus UM-57	Urban	Moderation	7	VanArsdel	79.71
456	Maximus UM-61	Urban	Moderation	7	VanArsdel	86.23

ProductDim

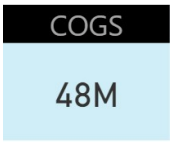
© 2017 Microsoft. All rights reserved.

Row Context in a Measure – Iterator Functions

Power BI

[COGS] = SUMX(Sales, Sales[Units] * RELATED(ProductDim[Unit Cost]))

SUM it up



SUM up list obtained

Iterators

Power BI

Why Can an Iterator be a Better Approach then a Calculated Column?

- You avoid creating a Calculated Column
- Let us see the impact of a Calculated Column Called COGS on Data model with 100K rows - What if we have 10 M rows?
- Iterators help you avoid several “Intermediate Calculated Columns”

DAX Foundations

Power BI

CALCULATE – Converting Row Context to Filter Context (Example 1)

```
Sales velocity Segment = IF(
                          SUMX(RELATEDTABLE(Sales), Sales[Sales Amount])>=200000,
                         “High  Velocity”,
                         “Low Velocity”)

Sales Velocity (Using CALCULATE) = IF (
                                    CALCULATE(SUM(Sales[Sales Amount])) >= 200000,
                                    "High Velocity",
                                    "Low Velocity")
```

- Another way to do Dynamic Segmentation
- This method does not use Iterators
- Instead it uses CALCULATE to convert Row Context to Filter Context

Other Iterator Functions

Power BI

AVERAGEX , PRODUCTX, MINX, MAXX– All work the same way as SUMX

RANKX – Works similar to SUMX, but slightly more complex (more options)

Table Functions – Summary and Application

Power BI

Table functions can be used in 2 ways in Power BI Desktop:

- As an input to another DAX function
  - CALCULATE
  - Iterator functions
- Calculated Tables

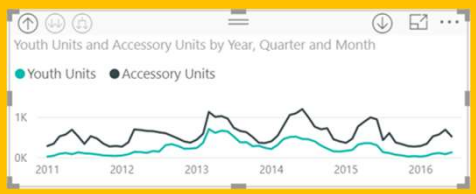
CALCULATE is one of the primary places where Table functions are used

Module 4 Lab

Create a report for the VP in charge of the Youth and Accessory Segments

1. Include a table visualization showing total units sold in the Youth Segment, Accessory Segment, and all other segments; by Campaign Device
2. Include a line chart showing total units sold in Youth and Accessory Segments by month
3. BONUS: Use the Unit Cost and Unit Price from the ProductDim table to calculate Sales Amount, Cost of Goods Sold, Profit and build some visuals around them

Device	Total Units	Youth Units	Accessory Units	Rest of Company Units
Deskop	10806	222	653	9931
Desktop	218680	4933	12412	201335
Mobile	198014	4427	11420	182167
Paper	40524	908	2376	37240
Tablet	207344	5151	12308	189885
Total	**675368**	**15641**	**39169**	**620558**



KNOWLEDGE CHECK Module 4

Power BI

- What are the different kinds of evaluation contexts?
- When are filter or a row contexts present?
- Which functions are commonly used to *modify* existing evaluation contexts?

Module 5
Advanced DAX
Time Intelligence Functions

MODULE OBJECTIVES

Power BI

- Be able to parse advanced DAX formulas (e.g., cumulative functions)
- Gain familiarity with standard DAX patterns
- Introduction to resources for further learning

103

Advanced DAX

Power BI

Before we get to Time Intelligence - Let us apply all of the DAX techniques

```
[SalesYTD] =

CALCULATE (
   [Total Sales],
   FILTER (
      ALL ( DateDim),
      DateDim[Year] = MAX ( DateDim[Year] )
         && DateDim[Date] <= MAX(DateDim[Date])
   )
)
```

Date	Year	Total Sales	SalesYTD
January 1, 2011	2011	$551	$551
January 2, 2011	2011	$7,366	$7,917
January 3, 2011	2011	$1,873	$9,790
January 4, 2011	2011	$10,113	$19,902
January 5, 2011	2011	$9,660	$29,562
January 6, 2011	2011	$14,450	$44,012
January 7, 2011	2011	$7,883	$51,895
January 8, 2011	2011	$11,793	$63,688
January 9, 2011	2011	$10,341	$74,029
January 10, 2011	2011	$1,374	$75,404
January 11, 2011	2011	$10,950	$86,353
January 12, 2011	2011	$20,217	$106,570
January 13, 2011	2011	$16,812	$123,382
January 14, 2011	2011	$15,215	$138,597
January 15, 2011	2011	$15,841	$154,438
January 16, 2011	2011	$14,391	$168,828
January 17, 2011	2011	$2,423	$171,252
January 18, 2011	2011	$15,712	$186,964
January 19, 2011	2011	$23,557	$210,521
January 20, 2011	2011	$20,912	$231,434

Let us take a super complicated DAX statement and break it down and understand what it means

Advanced DAX

Power BI

Before we get to Time Intelligence - Let us apply all of the DAX techniques

```
[SalesYTD] =

CALCULATE (
   [Total Sales],
   FILTER (
      ALL ( DateDim),
      DateDim[Year] = MAX ( DateDim[Year] )
         && DateDim[Date] <= MAX( DateDim[Date] )
   )
)
```

- In a CALCULATE statement Filter arguments are evaluated first
- The Filter in this case comes from a FILTER function
- FILTER function is an iterator

Advanced DAX

Power BI

Let us apply all of the data modeling techniques

```
[SalesYTD] =

CALCULATE (
   [Total Sales],
   FILTER (
      ALL ( DateDim),
      DateDim[Year] = MAX ( DateDim[Year] )
         && DateDim[Date] <= MAX( DateDim[Date] )
   )
)
```

- In a FILTER statement the input Table is evaluated first
- ALL statement means take all DateDim

Advanced DAX

Power BI

Let us apply all of the data modeling techniques

```
[SalesYTD] =

CALCULATE (
   [Total Sales],
   FILTER (
      ALL ( DateDim),
      DateDim[Year] = MAX ( DateDim[Year] )
         && DateDim[Date] <= MAX(DateDim[Date] )
   )
)
```

- Iterate through each row in DateDim
- Check for condition based on row context and filter context

Pro Tip: if you need to concatenate two conditions with an **AND** use **&&** for and **OR** use **||**

Advanced DAX

Power BI

Let us apply all of data modeling techniques

```
[SalesYTD] =

CALCULATE (
   [Total Sales],
   FILTER (
      ALL ( DateDim),
      DateDim[Year] = MAX ( DateDim[Year] )
         && DateDim[Date] <= MAX(DateDim[Date] )
   )
)
```

- Now you have a FILTERED list of dates
- Use this to update the filter context (since it is in a CALCULATE statement)

Advanced DAX

Power BI

Let us apply all of the data modeling techniques

```
[SalesYTD] =

CALCULATE (
   [Total Sales],
   FILTER (
      ALL ( DateDim),
      DateDim[Year] = MAX ( DateDim[Year] )
         && DateDim[Date] <= MAX(DateDim[Date] )
   )
)
```

- Use updated FILTER context to evaluate 'Total Sales'

Advanced DAX – Time Intelligence

Power BI

Introducing Time Intelligence – There is an App for that!!

```
[SalesYTD Easier] =

CALCULATE (
   [Total Sales],
   DATESYTD(DateDim[Date])
)
```

- This allows you to write the formula without being a DAX guru!
- Microsoft is continuously improving Time Intelligence functions to make it simple to use

Time Intelligence functions are your friends – They will save you time!

Advanced DAX – Month over Month

Power BI

```
Total Sales Last Month =
CALCULATE([Total Sales],
   PREVIOUSMONTH(DateDim[Date]))

MoM =
DIVIDE([Total Sales] - [Total Sales Last Month],
  [Total Sales Last Month])
```

- DAX has several shortcut Time Intelligence functions

Advanced DAX – Monthly Active Users

Power BI

```
[Monthly Active Users] =
CALCULATE(
   SUMX(VALUES(Sales [CustomerId]),1),
   ALL('DateDim'),
   DATESINPERIOD('DateDim'[Date],
   LASTDATE('DateDim'[Date]), -1, MONTH)
)
```

- Sumx vs DistinctCount of CustomerID
- SUMX can be more performant

Advanced DAX – Time Intelligence

Power BI

Other Time Intelligence Functions

DATESINPERIOD

DATESYTD

DATESQTD

NEXTMONTH

NEXTYEAR

PREVIOUSYEAR

PREVIOUSMONTH

SAMEPERIODLASTYEAR

PARALLELPERIOD

Pro Tip: Learn about Time Intelligence functions - https://msdn.microsoft.com/en-us/library/ee634763.aspx

KNOWLEDGE CHECK Module 5

Power BI

- Can I parse advanced DAX formulas?
- What are some standard DAX patterns?
- Which time intelligence functions are built-in to DAX?

Appendix

KNOWLEDGE CHECK ANSWERS Module 1

Power BI

- What is a *data model* in the context of Power BI?
  - *A data model is a collection of tables and relationships*
- What are some advantages of a star schema over a flat or denormalized model?
  - *Dimension tables save space by reducing the amount of data that needs to be repeated over and over in every row*
  - *Relationships between tables can be leveraged for more complex measures*
- How might you improve the performance of a Power BI model?
  - *Try using a star schema instead of a flat or denormalized model*
  - *Remove unnecessary columns*
  - *Set appropriate data types*
- How does Power BI store DateTime information? What are some consequences of this?
  - *DateTime information is stored as a floating-point decimal number. This means that datetimes are very precise but not very efficient to store.*

KNOWLEDGE CHECK ANSWERS Module 2

Power BI

- When is Calculated Column Evaluated?
  - *At the time of data load/data refresh.*
- What is Default Summarization?
  - *A default summarization is an implicit measure created in the background when you put a numeric field on a visualization. The function used (sum/max/min/avg/…) is based on the numeric field's default summarization setting.*
- When is a Measure Evaluated?
  - *At render time.*
- When to use Measures and Calculated Columns?
  - *It depends ☺. Calculated columns are useful when each row of data should be independently considered (although measures can do this too!) and the result won't change until the next data refresh. Measures should be used everywhere else.*

KNOWLEDGE CHECK ANSWERS Module 4

Power BI

- What are the different kinds of evaluation contexts?
  - *Filter context and row context*
- When are filter or a row contexts present?
  - *Row contexts are present in iterator functions and calculated column evaluations. Filter contexts are present in pivot tables and other visualizations.*
- Which functions are commonly used to *modify* existing evaluation contexts?
  - *CALCULATE, ALL, etc.*

KNOWLEDGE CHECK ANSWERS Module 5

Power BI

- Can I parse advanced DAX formulas?
  - *Yes I can!*
- What are some standard DAX patterns?
  - *CALCULATE(…)*
- Which time intelligence functions are built-in to DAX?
  - *Lots of them…YTD, FY, previous month, etc*

119

CALCULATE

Power BI

CALCULATE – Steps in Evaluating the CALCULATE Function

CALCULATE(Expression, [Filter1], [Filter2]…..)

- Step1 : Copy the current filter context
- Step 2: Add new filters if any
- Step 3: Update/ignore existing filters if any
- Step 4: Convert row context to filter context
- Step 5: AND all filter conditions to create new filter context
- Step 6: Evaluate the Expression
- Step 7: Return back to original filter context

Iterator Function

Power BI

Iterator Function Example 3

- Ranking Using Iterators in Calc. Column
- CALCULATE to convert Row Context to Filter Context

```
Rank Of Sales =

    VAR CurrentProductSales = CALCULATE ( SUM (Sales[Sales Amount] ) )

    RETURN

    SUMX(FILTER(ProductDim, CALCULATE ( SUM (Sales[Sales Amount] ) )  > CurrentProductSales), 1)+1
```