P3 ADAPTIVE

Reference Card

Note : P3 Adaptive LLC ALLOWS and ENCOURAGES reprinting and/or electronic distribution of this reference material, at no charge, provided: 1) it is being used strictly for free educational purposes and 2) it is reprinted or distributed in its entirety, including all pages, and without alteration of any kind.

Power BI and PowerPivot:
How the DAX Engine Calculates Measures

1

ModelName	2018	2019
Mountain-200	$4,735,493	$4,425,566
Road-250	$2,771,195	$957,793
Road-650	$348,431	

IMPORTANT: Every single measure cell is calculated independently, as an island! (That's right, even the Grand Total cells!) So when a measure returns an unexpected result, we should pick one cell and step through it, starting with Step 1 here...

Detect Filter Coordinates of Current Measure Cell:

Calendar(Year)=2019, Products(Model)="Road-250")
Those are the inital **filter context.**

2 **Calculate Alters Filter Context**

If applicable <filters> from CALCULATE(), adding/removing/modifying coordinates and producing a new filter context.
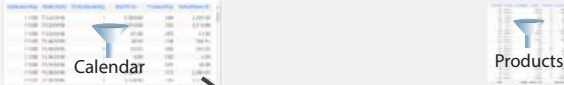
3 **Apply the Coordinates**

In the filter context to each of the respective tables (Calendar and Products in this example). This results in a set of "active" rows in each of those tables.

4 **Filters Follow the Relationship(s)**

If the filtered tables (Calendar and Products) are Lookup tables, follow relationships to their related Data tables and filter those tables too. Only Data rows related to active Lookup rows will remain active.

Calendar   Products

CustomerKey	OrderDate	OrderQuantity	UnitPrice	ProductKey	SalesAmount
11000	7/22/2016	1	3,399.99	344	3,399.99
11000	7/22/2018	1	2,319.99	353	2,319.99
11000	7/22/2018	1	21.98	485	21.98
11000	11/4/2018	3	34.99	214	104.97
11000	11/4/2018	5	53.99	488	269.95
11000	11/4/2018	1	4.99	530	4.99
11000	11/4/2018	1	28.99	541	28.99
11000	11/4/2018	1	2,384.07	573	

Data Table (Ex: Sales)

5 **Evaluate the Arithmetic**

Once all filters are applied and all relationships have been followed, evaluate the arithmetic - SUM(), COUNTROWS(), etc. in the formula against the remaining active rows.

6 **Return Result**

The result of the arthmetic is returned to the current measure cell in the pivot (or dashboard, etc.), then the process starts over at step 1 for the next measure cell.

P3
ADAPTIVE

Exercises for Step 1 (Filter Context) of DAX Measure Evaluation Steps

In each of the 9 pivots below, identify the filter context (the set of coordinates from the pivot) for the circled cell. (We find that coordiante identification often trips people up, hence this exercise).

In 1-4, the Region(Country) column is on Rows, & Products(Category) on Columns. (Total Sales) is on Values.

1

Country	Accessories	Bikes	Clothing	Total
Australia	$334,029	$11,364,439	$183,449	**$11,881,916**
Canada	$252,900	$2,450,599	$138,823	**$2,842,321**
France	$154,545	$3,272,211	$68,915	**$3,495,671**
Germany	$155,460	$3,645,491	$62,348	**$3,863,299**
United Kingdom	$194,163	$4,253,725	$83,270	**$4,531,159**
United States	$634,680	$11,660,745	$346,243	**$12,641,668**
Total	**$1,725,775**	**$36,647,211**	**$883,047**	**$39,256,034**

2

Country	Accessories	Bikes	Clothing	Total
Australia	$334,029	$11,364,439	$183,449	**$11,881,916**
Canada	$252,900	$2,450,599	$138,823	**$2,842,321**
France	$154,545	$3,272,211	$68,915	**$3,495,671**
Germany	$155,460	$3,645,491	$62,348	**$3,863,299**
United Kingdom	$194,163	$4,253,725	$83,270	**$4,531,159**
United States	$634,680	$11,660,745	$346,243	**$12,641,668**
Total	**$1,725,775**	**$36,647,211**	**$883,047**	**$39,256,034**

3

Country	Accessories	Bikes	Clothing	Total
Australia	$334,029	$11,364,439	$183,449	**$11,881,916**
Canada	$252,900	$2,450,599	$138,823	**$2,842,321**
France	$154,545	$3,272,211	$68,915	**$3,495,671**
Germany	$155,460	$3,645,491	$62,348	**$3,863,299**
United Kingdom	$194,163	$4,253,725	$83,270	**$4,531,159**
United States	$634,680	$11,660,745	$346,243	**$12,641,668**
Total	**$1,725,775**	**$36,647,211**	**$883,047**	**$39,256,034**

4

Country	Accessories	Bikes	Clothing	Total
Australia	$334,029	$11,364,439	$183,449	**$11,881,916**
Canada	$252,900	$2,450,599	$138,823	**$2,842,321**
France	$154,545	$3,272,211	$68,915	**$3,495,671**
Germany	$155,460	$3,645,491	$62,348	**$3,863,299**
United Kingdom	$194,163	$4,253,725	$83,270	**$4,531,159**
United States	$634,680	$11,660,745	$346,243	**$12,641,668**
Total	**$1,725,775**	**$36,647,211**	**$883,047**	**$39,256,034**

In #5, we've swapped Region(Country) from Rows to Columns, and Products(Category) from Columns to Rows. We've also turned off display of grand totals.

5

CategoryName	Australia	Canada	France	Germany	United Kingdom	United States
Accessories	$334,029	$252,900	$154,545	$155,460	$194,163	$634,680
Bikes	$11,364,439	$2,450,599	$3,272,211	$3,645,491	$4,253,725	$11,660,745
Clothing	$183,449	$138,823	$68,915	$62,348	$83,270	$346,243

P3
ADAPTIVE

Exercises for Step 1 (Filter Context) of DAX Measure Evaluation Steps

In 6-8, Region(Continent) and Region(Region) are on Rows. Customers(Gender) is on Report Filters. In 6 and 7, Customers(Gender) Is not filtered, but in 8, it is filtered to "F". In 6-8, (Total Sales) and (Orders) are on Values.

Table 6:

| Gender | F | | M | | Total | |
Continent	Total Sales	Orders	Total Sales	Orders	Total Sales	Orders
Europe	$5,853,119	3,957	$6,037,010	4,042	$11,890,128	7,999
France	$1,647,581	1,232	$1,848,090	1,252	$3,495,671	2,484
Germany	$2,082,635	1,245	$1,780,663	1,239	$3,863,299	2,484
United Kingdom	$2,122,903	1,480	$2,408,256	1,551	$4,531,159	3,031
North America	$7,768,235	6,412	$7,715,754	6,530	$15,483,989	12,942
Canada	$1,398,720	1,621	$1,443,602	1,754	$2,842,321	3,375
Central	$232	3	$3,357	6	$3,589	9
Northeast	$3,871	3	$3,126	7	$6,997	10
Northwest	$2,481,162	2,043	$2,463,580	2,015	$4,944,743	4,058
Southeast	$14,690	12	$621	5	$15,311	17
Southwest	$3,869,560	2,730	$3,801,468	2,743	$7,671,028	5,473
Pacific	$6,232,197	3,373	$5,649,719	3,345	$11,881,916	6,718
Australia	$6,232,197	3,373	$5,649,719	3,345	$11,881,916	6,718
Total	$19,853,551	13,742	$19,402,483	13,917	$39,256,034	27,659

Table 7:

| Gender | F | | M | | Total | |
Continent	Total Sales	Orders	Total Sales	Orders	Total Sales	Orders
Europe	$5,853,119	3,957	$6,037,010	4,042	$11,890,128	7,999
France	$1,647,581	1,232	$1,848,090	1,252	$3,495,671	2,484
Germany	$2,082,635	1,245	$1,780,663	1,239	$3,863,299	2,484
United Kingdom	$2,122,903	1,480	$2,408,256	1,551	$4,531,159	3,031
North America	$7,768,235	6,412	$7,715,754	6,530	$15,483,989	12,942
Canada	$1,398,720	1,621	$1,443,602	1,754	$2,842,321	3,375
Central	$232	3	$3,357	6	$3,589	9
Northeast	$3,871	3	$3,126	7	$6,997	10
Northwest	$2,481,162	2,043	$2,463,580	2,015	$4,944,743	4,058
Southeast	$14,690	12	$621	5	$15,311	17
Southwest	$3,869,560	2,730	$3,801,468	2,743	$7,671,028	5,473
Pacific	$6,232,197	3,373	$5,649,719	3,345	$11,881,916	6,718
Australia	$6,232,197	3,373	$5,649,719	3,345	$11,881,916	6,718
Total	$19,853,551	13,742	$19,402,483	13,917	$39,256,034	27,659

Filters (for 6 and 7):
- Gender is (All)
- Continent is (All)
- ModelName is (All)
- Orders is (All)
- Territory is (All)
- Total Sales is (All)
- Year is (All)

Table 8:

| Gender | F | | Total | |
Continent	Total Sales	Orders	Total Sales	Orders
Europe	$5,853,119	3,957	$5,853,119	3,957
France	$1,647,581	1,232	$1,647,581	1,232
Germany	$2,082,635	1,245	$2,082,635	1,245
United Kingdom	$2,122,903	1,480	$2,122,903	1,480
North America	$7,768,235	6,412	$7,768,235	6,412
Canada	$1,398,720	1,621	$1,398,720	1,621
Central	$232	3	$232	3
Northeast	$3,871	3	$3,871	3
Northwest	$2,481,162	2,043	$2,481,162	2,043
Southeast	$14,690	12	$14,690	12
Southwest	$3,869,560	2,730	$3,869,560	2,730
Pacific	$6,232,197	3,373	$6,232,197	3,373
Australia	$6,232,197	3,373	$6,232,197	3,373
Total	$19,853,551	13,742	$19,853,551	13,742

Filters (for 8):
- Gender is F
- Continent is (All)
- ModelName is (All)
- Orders is (All)
- Territory is (All)
- Total Sales is (All)
- Year is (All)

In 9, Region(Continent) is a Slicer. Consumers(Gender) is on Rows. (Orders) is on Values.

Slicer - Continent: Europe, NA, North America (selected), Pacific

Table 9:

Gender	Orders
F	6,412
M	6,530
Total	12,942

Answers

1) Region(Country) = "France", Products(Category) = "Bikes"

2) Region(Country) = "Germany"

3) Products(Category) = "Accessories"

4) No Filters

5) Same as #1!

6) Region(Continent) = "North America", Region(Region) = "Northwest"

7) Same as #6!

8) Region(Continent) = "North America", Customers(Gender) = "F"

9) Same as #8!

P3 ADAPTIVE

Commonly-Used DAX Functions and Techniques

CALCULATE ( ) Function

CALCULATE(<measure expression>, <filter1>, <filter2>, … <filterN>)

<measure expression>:	[MeasureName] SUM(Table[Column]) Any measure name or valid formula for a measure
"Simple"<filter>:	Sales[TransactionType]=1 Products[Color]="Blue" Calendar[Year]>=2009 Sales[TransType]=1 || TransType]=3
Advanced<filter>:	ALL(…) FILTER(…) DATESBETWEEN(…) Any other function that modifies filter context
Notes:	Raw <filter>'s override (replace) filter context from pivot Raw <filter>'s must be Table[Column] <operator> <fixed value> Multiple <filter>'s arguments get AND'd together

ALL ( ) Function

ALL(<table>) or ALL(Table[Col1], Table[Col2], …Table[ColN])

Basic usage:	As a <filter> argument to CALCULATE() Removes filters from specified table or column(s) Strips those tables/columns from the pivot's filter context
Advanced usage:	Technically, ALL() returns a table So it is also useable wherever a <table expr > is required …such as the first argument to FILTER()

Common Date Calculations

Year to Date:	CALCULATE(<measure>, DATESTYD(Calendar[Date])
Qtr or Month to: date:	Substitute DATESQTD or DATESMTD for Quarter or Month to date
Previous Month:	CALCULATE(<measure>, DATEADD(Calendar[Date], -1, Month)
Prev QT/Year/Day:	Substitute "Quarter" or "Year" or "Day" for "Month" as last argument
30-day Moving Avg:	CALCULATE(<measure>, DATESINPERIOD(Calendar[Date], MAX(Calendar[Date]), -30, Day)/30

Time Intelligence with Custom Calendar

When Your Biz Calendar is Too Complex for the Built-In Functions

=CALCULATE   (<measure expr>, FILTER(ALL(<Custom Cal Table>), <custom filter>), <optional VALUES() to restore filters on some Cal fields>)

=CALCULATE([Sales],   FILTER(ALL(Cal445), Cal445[Year]=MAX(Cal445[Year])-1))

=CALCULATE([Sales],   FILTER(ALL(Cal445), Cal445[Year]=MAX(Cal445[Year])-1),VALUES (Cal445[MonthOfYear]))

SWITCH( ) Function

Alternative to Nested IF's!

=SWITCH(<value to test>,   <if it matches this value>, <return this value>,
     <if it matches this value>, <return this value>,
     …more match/return pairspairs…,
     <if no matches found, return this optional "else" value>)

FILTER ( ) Function

FILTER(<table expression>, <single rich filter>)

<table expression>:	The Name of a Table, or any of the below… VALUES(Table[Column]) - unique values of Table[Column] for current pivot cell ALL(Table) or ALL(Table[Column]) Any expression that returns a table, such as DATESYTD() Even another FILTER() can be used here for instance
<rich filter>:	Table[Column1] >= Table[Column2] Table[Column] <= [Measure] [Measure1] <> [Measure2] <true/false expr1> && <true/false expr2> Any expression that evaluates to true/false
Notes:	Commonly used as a <filter> argument to CALCULATE() Useful when a richer filter test is required than "simple" filters can do Never use FILTER when a "simple" CALCULATE() <filter> will work Slow and eats memory when used on large tables Use against small (Lookup) tables for better performance Advanced usage: use anywhere a <table expr> is required

VALUES ( ) Function

VALUES(Table[Column])

1-column table, unique: (Most common usage)	Produces a temporary, single-column table during formula evaluation That table contains ONLY the UNIQUE values of Table[Column]. EX: CALCULATE(<measure>, FILTER(VALUES(Customers[Postal Code]),]),…)) That allows us to iterate as if we had a PostalCode table, even though we don't! And then the formula above calculates <measure> only for those Postal Codes that "survive" the <filter expr> test inside the FILTER function. And therefore only includes the customers IN those postal codes!
Restoring a filter: (2nd most common usage)	CALCULATE([M], ALL(Table), VALUES(Table[Col1])) …is roughly equiv to CALCULATE([M], ALLEXCEPT(Table,Table[Col1]))
Note:	VALUES(Table[Column]) returns filtered list even if Table[Column] isn't on pivot!

Forcing Grand/Sub Totals to Be the Sum of Their "Parts"

=SUMX(VALUES(Table[Column], <original measure>)

(Where the values of Table[Column] are the "small pieces" that need to be calculated individually and then added up.)

Calc Columns That Reference "Previous" Row(s)

=CALCULATE([Measure],FILTER(<table>, Table[Col]=EARLIER(Table[Col])-1))

=CALCULATE(AVERAGE(Tests[Score]),FILTER(Tests, Tests[ID]=EARLIER (Tests[ID])-1))

Suppressing Subtotals/Grand Totals

=IF(HASONEVALUE(Table[Column]), <measure expr for non non-totals>, BLANK())

RANKX( ) Function

RANKX(<table expr>, <arithmetic expression>, <optional alternate arithmetic expression>, <optional sort order flag>, <optional tie-handling flag>)

<table expression>:	RANKX(ALL(Table[Column]), <numericalexpr>) EX: RANKX(ALL(Products[Name]), [TotalSales])
Ascending Rank Order:	EX: RANKX(ALL(Products[Name]), [TotalSales],,,Dense)
"Dense" Tie Handling:	EX: RANKX(ALL(Products[Name]), [TotalSales],,1)

DIVIDE Function

Returns BLANK() Cells on " Div by Zero", No IF() or IFERROR() required!
=DIVIDE(<numerator>, <denominator>, <optional val to return when div by zero>)
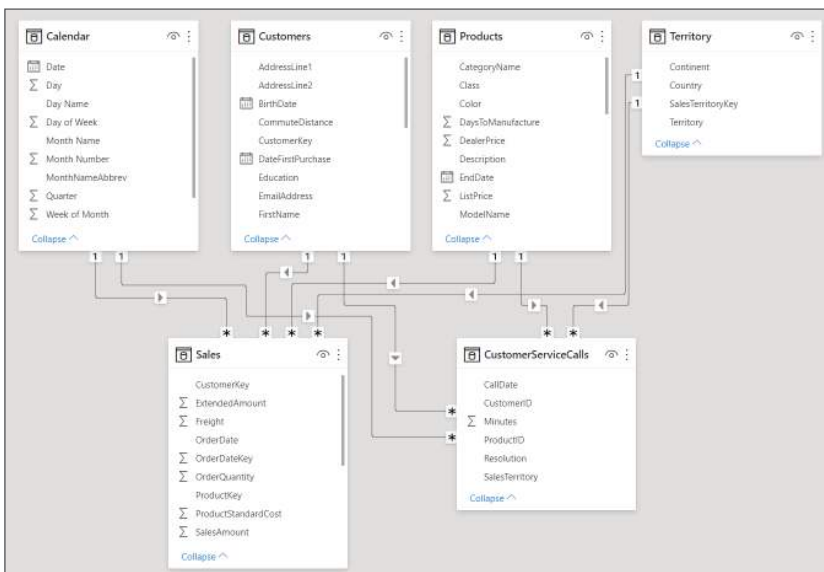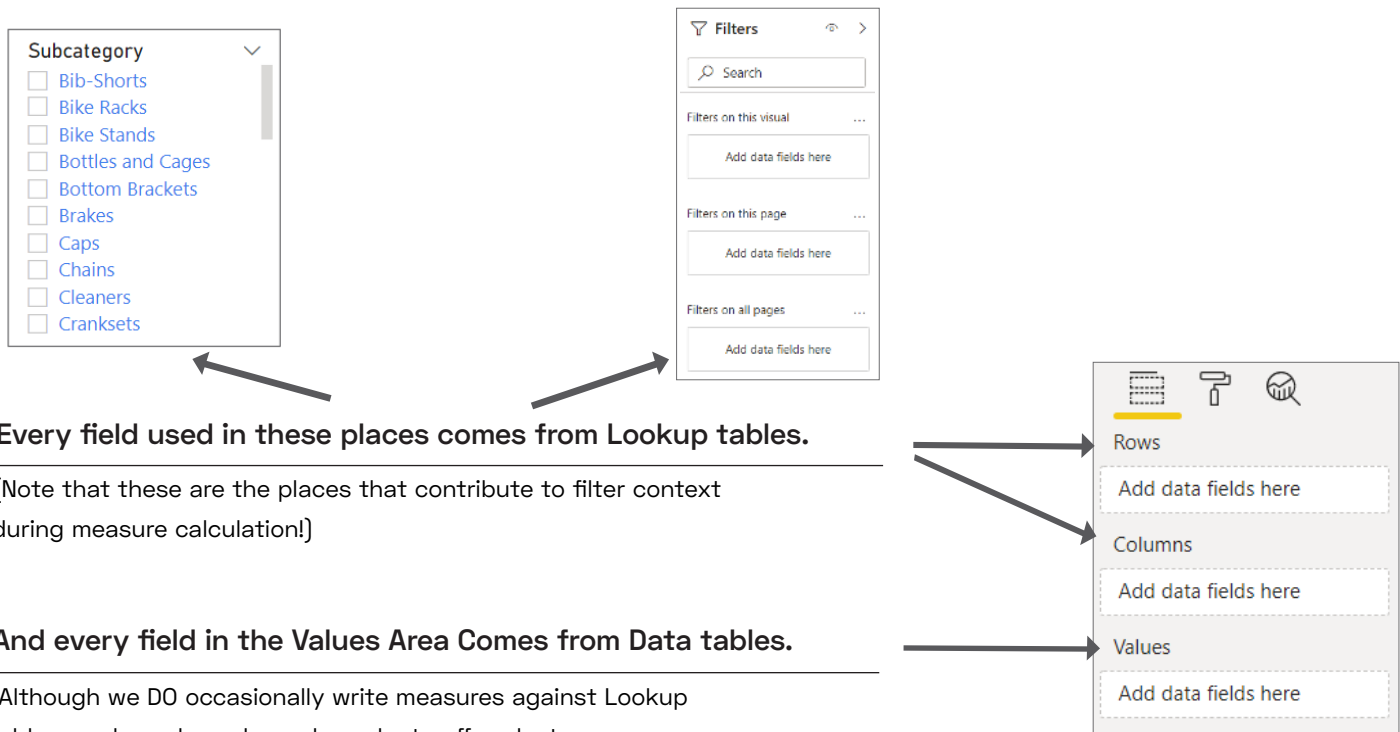
P3 ADAPTIVE

Lookup Tables

- Typically, WIDE (Many Columns)
- MOSTLY Text
- LOOKUP Information (E.g., Name, Address, Description, etc…)
- SLOW Changing (Updated Less Often)
- Does NOT typically contain Time Data
- The ONE side of a relationship

Data Tables

- Typically, TALL (Many Rows)
- MOSTLY Dates and Numbers
- Do MATH against it (E.g., SUM, AVERAGE, MIN, MAX, etc…)
- Fast changing (Updated Often)
- MAY contain Time Data (E.g,. Order Date, Record Time, etc…)
- The MANY side of a relationship

Under "Ideal" Conditions, Data and Lookup Tables are Used Like THIS:

Every field used in these places comes from Lookup tables.

(Note that these are the places that contribute to filter context during measure calculation!)

And every field in the Values Area Comes from Data tables.

(Although we DO occasionally write measures against Lookup tables, such as days elapsed, products offered, etc.

Note:
- Data tables are "spliced together" ONLY by sharing one or more Lookup tables
- Now follow the field list guidelines above and you can compare Budget v Actuals (for instance) in a single pivot!
- Data tables are never related directly to each other!

Also:
- Useful trick: Arrange Lookup tables "up high" on the diagram and Data tables "down low."
- This lets us envision filters flowing "downhill" across relationships (relationships are "1 1-way")

P3 ADAPTIVE

Make the formula font bigger!

(Hold CTRL key down and roll mouse wheel forward)

CTRL + 🖱
SUM() ⇒ SUM ( )

Insert New Lines in Formulas:

=CALCULATE([Total],Table[Column]=6)

SHIFT + ENTER

When writing measures/calc fields:

1) Always INCLUDE table names on column references.

`Table[Column]`   `[Column]`

YES   NO

2) Always EXCLUDE table names when referencing other measures.

`[Measure]`   `Table[Measure]`

YES   NO

By following this convention, you will ALWAYS immediately know the difference between a measure and a column reference, on sight, and that's a BIG win for readability and debugging.

(But when writing a calc column, it is acceptable to omit the table name from a column reference, since you rarely reference measures in calc columns.)

NEVER write the same formaula twice!

For example, you should define basic measures like these, even for "simple" calculations like SUM:

`[Total Sales]:= SUM(Table[Amount])`   `[Total Cost]:= SUM(Table[Cost])`

And then references those measures whenever you are tempted to rewrite the SUM in another measure:

YES   [Total Margin]:=
[Total Sales] – [Total Cost]

YES   [Year to Date Sales]:=
CALCULATE([Total Sales], DATESYTD (Dates[Date])

NO   [Total Margin]:=
SUM(…) – SUM(…)

NO   [Year to Date Sales]:=
CALCULATE (SUM(…), DATESYTD(Dates[Date])

Measures (Calculated Fields) Are:

1. Used in cases when a single row can't give you the answer (typically aggregates like sum,

2. Only "legal" to be used in the Values area of a pivot

3. Never pre pre-calculated

4. ALWAYS re re-calculated in response to pivot changes – slicer or filter change, drill down, etc.

5. Return different answers in different pivots

6. Not a source of file size increase

7. "Portable Fprmulas!!"

Calculated Columns Are:

1. Used to "stamp" numbers or properties on each row of a table

2. "Legal" on row/column/filter/slicer of pivots

3. Useful for grouping and filtering, for instance

4. Also usable as inputs to measures

5. PrePre-calculated and stored – making the file bigger

6. NEVER re re-calculated in response to pivot changes

7. Only re re-calculated on data source refresh or on change to "precedent" (upstream) columns

Rename after import!

Overly-long and/or cryptically cryptically-named tables and columns make your formulas harder to read AND write, and since Power Pivot 2010 and 2013 don't fix up formulas on rename, it pays to rename immediately after import.

ServiceCalls
Category ▼
Components   YES

vsIncomingServiceCalls
szCategoryName ▼
Components   NO

NEVER Use Columns in Pivot Values Area

(Write the Measure/Calc Field Instead)
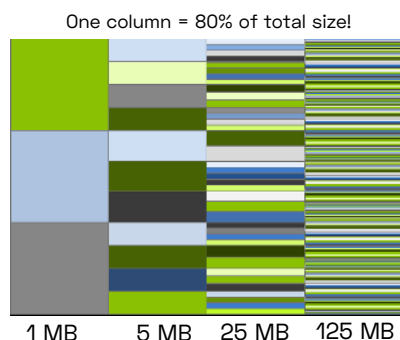
YES   NO

Σ VALUES
Amount Measure ▼
Quantity Measure ▼

Σ VALUES
Sum of Amount Column ▼
Sum of Quantity Column ▼

(See re-use & maintenance benefits in DAX Formulas for Power Pivot , Ch6)

P3 ADAPTIVE

Reducing File Size

One column = 80% of total size!

1 MB   5 MB   25 MB   125 MB

Power Pivot, Power BI Designer, and SSAS Tabular all store and compresses data in a "column stripe" format, as pictured here.

Each column is less compressed than the one before*it. (* The compression order of the columns is auto auto-decided by the engine at import time, and not something we can see or control.)

This column-oriented storage is VERY unlike traditional files, databases, and compression engines.

Sometimes, a single column is "responsible" for a large fraction of the file's size (like the 125 MB pictured here.)
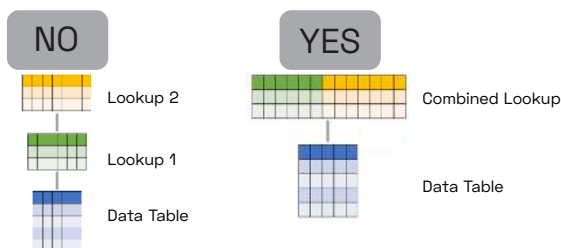
Calculated Column Notes

1. Calc columns bloat the file more than columns imported from a data source.

2. So consider implementing the calc column in the database (or use Power Query), then import it.

3. Unlike calc columns, measures do NOT add file size!

4. So in "simple arithmetic" cases like [Profit Margin], it's best to just subtract one measure from another ([Sales]–[Cost]), and avoid adding a calc column to perform the subtraction (which you'd then SUM to create your measure).

Words of Wisdom

1. If your file size is not a problem, don't worry about ANYTHING on this page. These tips are just for when you DO have a problem :)

2. The smaller the table is in terms of row count, the less these tips and tricks matter. A few extra columns in a a10k -row table are no big deal, but ONE extra column in a million million-row table sometimes IS.

3. So focus on Data tables. Lookup tables = less crucial.

4. Large files also eat more RAM. If your server is strained or 32 32-bit Excel breaks down, reduce file size.

Avoid "Multi-Hop" Lookups (if Possible)

Combine "chained" lookup tables into one table:

Separate Lookup Tables Offer BIG File Size Savings

ProductKey	OrderDate	CustomerKey	ExtendedAmount	ProductName	StandardCost	Color	ModelName
336	7/1/2014	14501	699.0982	Road-650 Black, 62	413.1463	Black	Road-650
336	7/5/2014	25249	699.0982	Road-650 Black, 62	413.1463	Black	Road-650
346	7/1/2014	25863	3399.99	Mountain-100 Silver, 44	1912.1544	Silver	Mountain-100
346	7/1/2014	28389	3399.99	Mountain-100 Silver, 44	1912.1544	Silver	Mountain-100
346	7/1/2014	11003	3399.99	Mountain-100 Silver, 44	1912.1544	Silver	Mountain-100
346	7/9/2014	25861	3399.99	Mountain-100 Silver, 44	1912.1544	Silver	Mountain-100
346	7/9/2014	11238	3399.99	Mountain-100 Silver, 44	1912.1544	Silver	Mountain-100
346	7/10/2014	11002	3399.99	Mountain-100 Silver, 44	1912.1544	Silver	Mountain-100
346	7/17/2014	11010	3399.99	Mountain-100 Silver, 44	1912.1544	Silver	Mountain-100
346	7/26/2014	11026	3399.99	Mountain-100 Silver, 44	1912.1544	Silver	Mountain-100
346	7/27/2014	11006	3399.99	Mountain-100 Silver, 44	1912.1544	Silver	Mountain-100
310	7/1/2014	21768	3578.27	Road-150 Red, 62	2171.2942	Red	Road-150
310	7/2/2014	16624	3578.27	Road-150 Red, 62	2171.2942	Red	Road-150
310	7/5/2014	27601	3578.27	Road-150 Red, 62	2171.2942	Red	Road-150
310	7/6/2014	13590	3578.27	Road-150 Red, 62	2171.2942	Red	Road-150
310	7/10/2014	16522	3578.27	Road-150 Red, 62	2171.2942	Red	Road-150
310	7/11/2014	13563	3578.27	Road-150 Red, 62	2171.2942	Red	Road-150
310	7/12/2014	27671	3578.27	Road-150 Red, 62	2171.2942	Red	Road-150
310	7/14/2014	16482	3578.27	Road-150 Red, 62	2171.2942	Red	Road-150
310	7/16/2014	27646	3578.27	Road-150 Red, 62	2171.2942	Red	Road-150

The table pictured above combines Data table columns (OrderDate, CustomerKey, ExtendedAmount , and ProductKey) with columns that should be "outsourced" to a Lookup table (ProductName, StandardCost , Color, and ModelName can all be "looked up" from the ProductKey).

P3 ADAPTIVE

Instead, split the Lookup-specific columns out into a separate Lookup table, and remove duplicate rows (in that Lookup table) so that we have just one row per unique ProductKey.

	1²₃ ProductKey	Aᴮ_C ProductName	1.2 StandardCost	Aᴮ_C Color	Aᴮ_C ModelName	Aᴮ_C Category	Aᴮ_C SubCategory
1	325	Road-650 Red, 62	486.7066	Red	Road-650	Bikes	Road Bikes
2	323	Road-650 Red, 60	486.7066	Red	Road-650	Bikes	Road Bikes
3	322	Road-650 Red, 60	413.1463	Red	Road-650	Bikes	Road Bikes
4	321	Road-650 Red, 58	486.7066	Red	Road-650	Bikes	Road Bikes
5	319	Road-450 Red, 52	884.7083	Red	Road-450	Bikes	Road Bikes
6	318	Road-450 Red, 48	884.7083	Red	Road-450	Bikes	Road Bikes
7	317	Road-450 Red, 44	884.7083	Red	Road-450	Bikes	Road Bikes
8	381	Road-550-W Yellow, 38	605.6492	Yellow	Road-550-W	Bikes	Road Bikes
9	328	Road-650 Red, 48	413.1463	Red	Road-650	Bikes	Road Bikes

YES

RELATIONSHIP

Duplicate removal makes a relationship possible with the Data table, AND makes the Lookup table small in terms of row count.

(Duplicate removal is performed in the database, or using Power Query – see Power Pivot Alchemy, chapter 5 for an example).

Our "big" table now has significantly fewer columns. On net, our file is potentially now MUCH smaller – because our largest table (Data table) has shed multiple columns. The small Lookup table is not significant, even if it contains 50+ columns.

	OrderDate	1²₃ CustomerKey	1.2 ExtendedAmount	1²₃ ProductKey
1	7/1/2014	14501	699.0982	336
2	7/1/2014	25863	3399.99	346
3	7/1/2014	28389	3399.99	346
4	7/1/2014	21768	3578.27	310
5	7/1/2014	11003	3399.99	346
6	7/2/2014	27645	3578.27	311
7	7/2/2014	11011	3399.99	344
8	7/2/2014	11005	3374.99	351
9	7/2/2014	16624	3578.27	310
10	7/3/2014	27621	3578.27	312
11	7/3/2014	27616	3578.27	312
12	7/3/2014	20042	699.0982	330
13	7/3/2014	16517	3578.27	314
14	7/3/2014	16351	3578.27	313
15	7/4/2014	27606	3578.27	314
16	7/4/2014	13513	3578.27	311
17	7/5/2014	27601	3578.27	310
18	7/5/2014	13591	3578.27	311
19	7/5/2014	25249	699.0982	336
20	7/5/2014	16483	3578.27	314
21	7/5/2014	16529	3578.27	311
22	7/6/2014	27612	3578.27	312
23	7/6/2014	27668	3578.27	311
24	7/6/2014	13264	3578.27	311
25	7/6/2014	13590	3578.27	310

YES

"Unpivot" ALSO Offers Big File Size Savings

	Aᴮ_C Region	1²₃ 1/1/2015	1²₃ 1/2/2015	1²₃ 1/3/2015	1²₃ 1/4/2015	1²₃ 1/5/2015	1²₃ 1/6/2015	1²₃ 1/7/2015	1²₃ 1/8/2015
1	North	2106	4712	1996	4147	5044	1869	3004	8032
2	South	2470	1375	6133	7040	1951	1141	7871	1850
3	East	6283	3591	7646	2417	8487	6973	3520	3540
4	West	8383	2925	8109	7996	6916	4401	8315	5995

NO

	Aᴮ_C Region	1²₃ Normal Sales	1²₃ Promotional Sales	1²₃ Refunds	1²₃ Bulk Sales	1²₃ Cost of Goods
1	North	2106	4712	1996	5044	4147
2	South	2470	1375	1850	1951	7040
3	East	6283	3591	1951	8487	2417
4	West	8383	2925	2106	6916	7996

NO

This "unpivot " transformation results in increased rows but fewer columns. Counterintuitively this can yield VERY significant file size reduction. (See Power Pivot Alchemy, Ch 5, for an example of performing this transformation with Power Query).

In the case of dates or months, this also removes the need for tedious formula repetition, AND enables time intelligence calcs.

P3
ADAPTIVE

	A^B_C Region	Date	1²₃ Value
1	North	1/1/2015	2106
2	South	1/1/2015	2470
3	East	1/1/2015	6283
4	West	1/1/2015	8383
5	North	1/2/2015	4712
6	South	1/2/2015	1375
7	East	1/2/2015	3591
8	West	1/2/2015	2925
9	North	1/3/2015	1996
10	South	1/3/2015	6133
11	East	1/3/2015	7646
12	West	1/3/2015	8109
13	North	1/4/2015	4147
14	South	1/4/2015	7040
15	East	1/4/2015	2417
16	West	1/4/2015	7996
17	North	1/5/2015	5044
18	South	1/5/2015	1951
19	East	1/5/2015	8487
20	West	1/5/2015	6916
21	North	1/6/2015	1869
22	South	1/6/2015	1141
23	East	1/6/2015	6973
24	West	1/6/2015	4401
25	North	1/7/2015	3004
26	South	1/7/2015	7871
27	East	1/7/2015	3520
28	West	1/7/2015	8315
29	North	1/8/2015	8032
30	South	1/8/2015	1850
31	East	1/8/2015	3540
32	West	1/8/2015	5995

YES

	A^B_C Region	A^B_C Amount Type	1²₃ Amount
1	North	Normal Sales	2106
2	North	Promotional Sales	4712
3	North	Refunds	1996
4	North	Bulk Sales	5044
5	North	Cost of Goods	4147
6	South	Normal Sales	2470
7	South	Promotional Sales	1375
8	South	Refunds	1850
9	South	Bulk Sales	1951
10	South	Cost of Goods	7040
11	East	Normal Sales	6283
12	East	Promotional Sales	3591
13	East	Refunds	1951
14	East	Bulk Sales	8487
15	East	Cost of Goods	2417
16	West	Normal Sales	8383
17	West	Promotional Sales	2925
18	West	Refunds	2106
19	West	Bulk Sales	6916
20	West	Cost of Goods	7996

YES

In this case you will need to use CALCULATE to write your "base" measures. EX:

CALCULATE(SUM(Table[Amount]),
Table[Amount Type]="Refunds")

What Makes a Valid Calendar/Dates Table?

	Date	A^B_C MonthShort	A^B_C Day Name	1²₃ Day of Week Number	1²₃ Quarter
1	7/1/2016	Jul	Friday	6	3
2	7/2/2016	Jul	Saturday	7	3
3	7/3/2016	Jul	Sunday	1	3
4	7/4/2016	Jul	Monday	2	3
5	7/5/2016	Jul	Tuesday	3	3
6	7/6/2016	Jul	Wednesday	4	3
7	7/7/2016	Jul	Thursday	5	3
8	7/8/2016	Jul	Friday	6	3

Must contain a column of actual Date data type, not just text or a number that looks like a date.

2. That Date column must NOT contain times – 12:00 AM is "zero time" and is EXACTLY what you want to see.

3. There CANNOT be "gaps" in the Date column. No skipped dates, even if your business isn't open on those days.
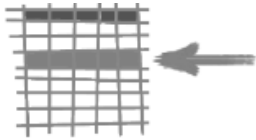
4. Must be "Marked as Date Table" via button on the Power Pivot window's ribbon (not applicable in Power BI Desktop).

5. May contain as many other columns as desired. Go nuts :)

6. Should not contain dates that "precede" your actual data – needless rows DO impact performance.

7. You MUST then use this as a proper Lookup table – don't use dates from your Data tables on Rows/Columns/Etc.!

P3
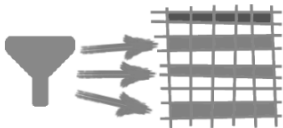*ADAPTIVE*

(Slightly) Advanced Concept: Row Context

• You HAVE a Row Context in a Calculated Column.

• But you do NOT have a Row Context in a Measure (Calculated Field)

• A calc column is calculated on a row row-by-row basis, so there's one row "in play" for each evaluation of the formula.

• So =[Column] resolves to a single value (the value from "this row"), w/out error.

• "The current row" is called Row Context

• You may only reference a "naked' column (naked = no aggregation fxn), and have it resolve to a single number, date, or text value when you have a Row Context.

Exception: Filter Context in Calc Columns

• Aggregation functions like SUM *always* reference the Filter Context

• Since there is no Filter Context in a calc column, =SUM([Column]) will return the sum of the ENTIRE column – you get the same answer all the way down.

• But you can tell the DAX engine to use a Row Context as if it were ALSO a Filter Context, by wrapping the aggregation function in a CALCULATE.

• EX: =CALCULATE(SUM[Column])) "respects" the context of each row, AND also relationships

• So in a Lookup table, you can use CALCULATE(SUM(Data[Col])) to get the sum of all "matching" rows from the related Data

• Furthermore, the DAX engine always "adds" a CALCULATE "wrapper" whenever you reference a Measure.

So =[MySumMeasure] ALSO respects Row Context and Relationships.

(Slightly) Advanced Concept: Filter Context

• You HAVE a Filter Context in a Measure / Calc Field.

• But you do NOT have a Filter Context in a Calc Column.

• Each cell in a Pivot's values area is calculated based on the filters (coordinates) specified for that cell.

• Those filters resolve to a set of multiple rows in the underlying data tables, rather than a single row.

• =[Column] is therefore illegal as a formula, or as part of a formula where a single value is needed.

• So this is why aggregation functions are required in measures – to "collapse" multiple values into one.

Exception: Row Context in Measures

• Certain functions step through tables one row at a time, even when used within a Measure.

• Those "iterator" functions are said to create Row Contexts during their operation.

• Ex: FILTER(tabletable, expr) and SUMX(tabletable, expr)

• In both examples, you CAN reference a column, within the expr argument, and use that column as a single value, within the expr argument.

• Note however that the column MUST "come from" the table specified in the table argument.

• Also note that this Row Context only exists within the evaluation of the iterator function itself (FILTER, SUMX, etc.) and does NOT exist elsewhere in the measure formula.

P3
ADAPTIVE