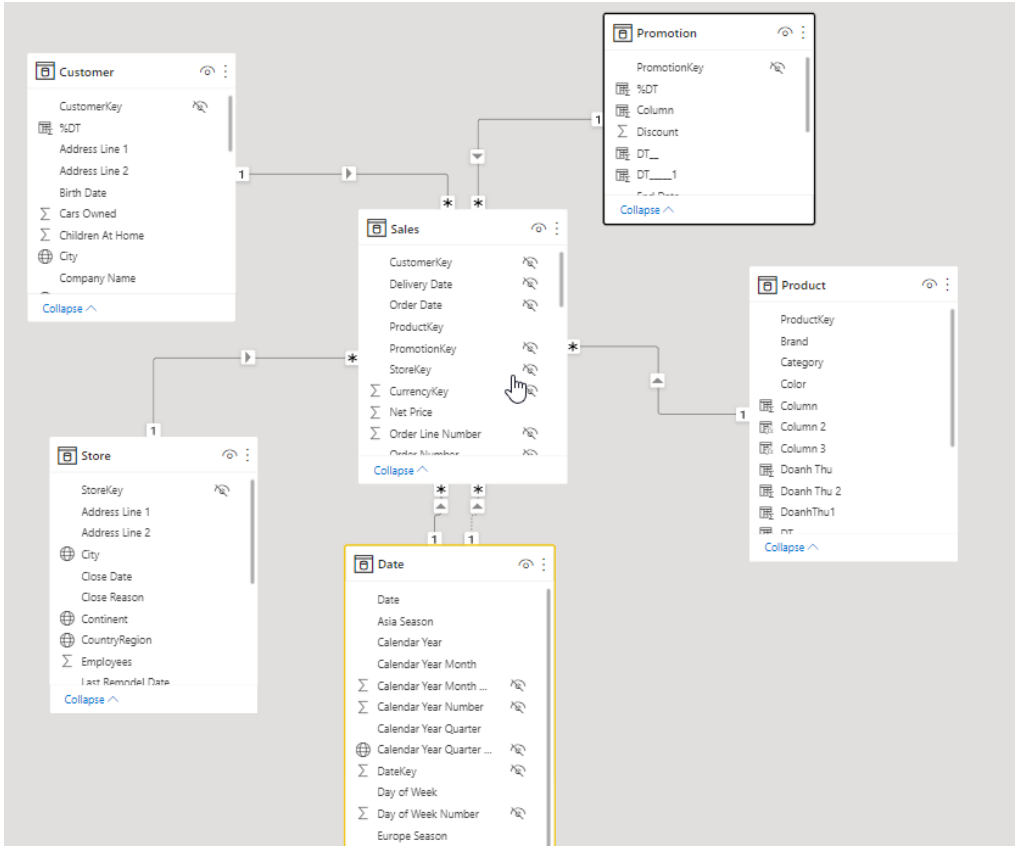
Power BI

DAX
Fundamental

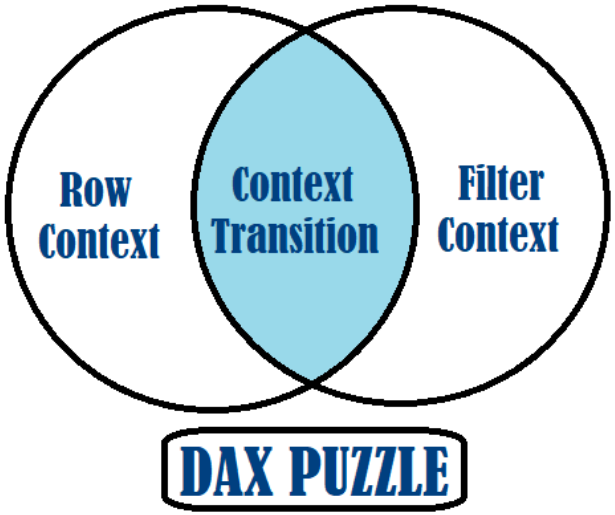
Nguyễn Huy Hoàng
Admin Data Analyst Skills (SQL, Power BI, ...)- Thực Hành Qua Tình Huống

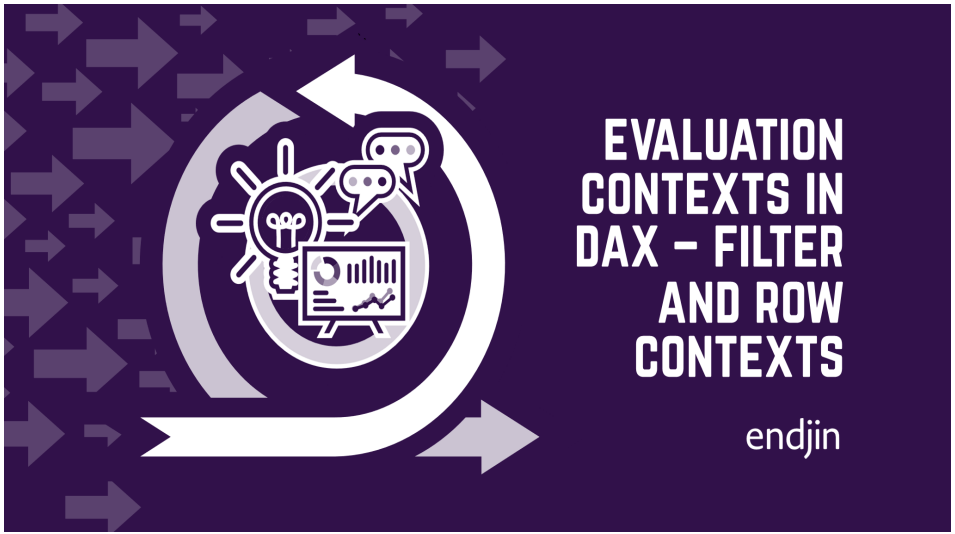
DATASET SỬ DỤNG ĐỂ MÔ PHỎNG CÁC VÍ DỤ

- File thực hành sẽ được mình upload lên nha
- Mô hình STAR SCHEMA được sử dụng mô để mô phỏng các ví dụ
- Bên cạnh đó các bạn có thể sử dụng tool Bravo để xem kết cấu dữ liệu trong bảng và tool DAX Studio để demo các ví dụ hàm bảng



CHAPTER 1: NGỮ CẢNH TRONG DAX





1. Row Context

- Bởi vì các **Table Data** trong Power BI được nối bằng các column database do đó chúng ta **chỉ có thể truy vấn cột** hay gọi 1 column trong power bi chứ **không thể thao tác từng dòng** trên power bi
- **Row Context** là việc thực hiện các phép toán theo hướng **row by row**

```
Combine = [Category] & " - "&[Product Name]
```

Combine
Home Appliances - Adventure Works Floor Lamp X1150 Bl
Home Appliances - Adventure Works Floor Lamp M2150 B
Home Appliances - Adventure Works Chandelier M8150 Bl
Home Appliances - Adventure Works Chandelier M6150 Bl
Home Appliances - Adventure Works Wall Lamp E2150 Blu
Home Appliances - Adventure Works Wall Lamp E3150 Blu
Home Appliances - Adventure Works Desk Lamp E1300 Blu
Home Appliances - Adventure Works Desk Lamp E1200 Blu
Home Appliances - WWI Floor Lamp X115 Blue
Home Appliances - WWI Floor Lamp M215 Blue
Home Appliances - WWI Chandelier M815 Blue
Home Appliances - WWI Chandelier M615 Blue
Home Appliances - WWI Wall Lamp E215 Blue
Home Appliances - WWI Wall Lamp E315 Blue
Home Appliances - WWI Desk Lamp E130 Blue

Ở đây sẽ được hiểu là: Qua mỗi dòng chúng ta đang nối các dòng của cột Category và Product name lại với nhau theo row by row

2. Evaluation Context

- Bởi vì các **Table Data** trong Power BI được nối bằng các column database do đó chúng ta chỉ có thể nên chúng ta **có thể truy vấn cột** hay gọi 1 column trong power bi chứ **không thể thao tác từng dòng** trên power bi
- **Evaluation Context** là việc tham chiếu 1 công thức lên tất cả các ô trong DAX.

```
Combine = [Category] & " - "&[Product Name]
```

Combine
Home Appliances - Adventure Works Floor Lamp X1150 Bl
Home Appliances - Adventure Works Floor Lamp M2150 B
Home Appliances - Adventure Works Chandelier M8150 Bl
Home Appliances - Adventure Works Chandelier M6150 Bl
Home Appliances - Adventure Works Wall Lamp E2150 Blu
Home Appliances - Adventure Works Wall Lamp E3150 Blu
Home Appliances - Adventure Works Desk Lamp E1300 Blu
Home Appliances - Adventure Works Desk Lamp E1200 Blu
Home Appliances - WWI Floor Lamp X115 Blue
Home Appliances - WWI Floor Lamp M215 Blue
Home Appliances - WWI Chandelier M815 Blue
Home Appliances - WWI Chandelier M615 Blue
Home Appliances - WWI Wall Lamp E215 Blue
Home Appliances - WWI Wall Lamp E315 Blue
Home Appliances - WWI Desk Lamp E130 Blue

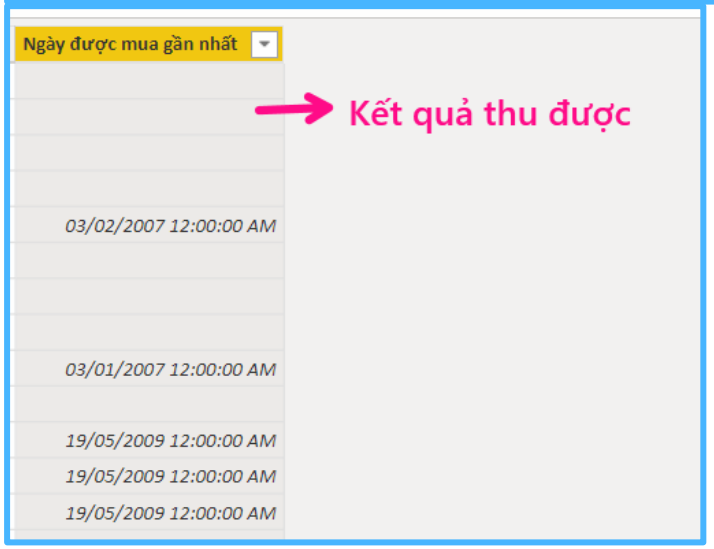
Ở đây sẽ được hiểu là:
Qua mỗi dòng chúng ta đang nối các dòng của cột Category và Product name lại với nhau theo row by row

2. Evaluation Context

- Một **ứng dụng khác** của Evaluation Context đó là gọi từng ô trên 1 column ra
- Việc **gọi từng ô** trên 1 column ra có thể giúp việc **kiểm tra dữ liệu** từ bảng dim đến bảng fact dễ dàng và cho thấy được ô nào đang bị lỗi dẫn đến các kết quả không mong muốn
- Ví dụ: bảng dim có SP A và SP A được bán qua các ngày các tháng. Nhưng khi chúng ta click filter vào SP A thì không trả về kết quả hoặc kết quả bị rỗng hoặc kết qua sai so với các tính toán ở bộ phận kế toán

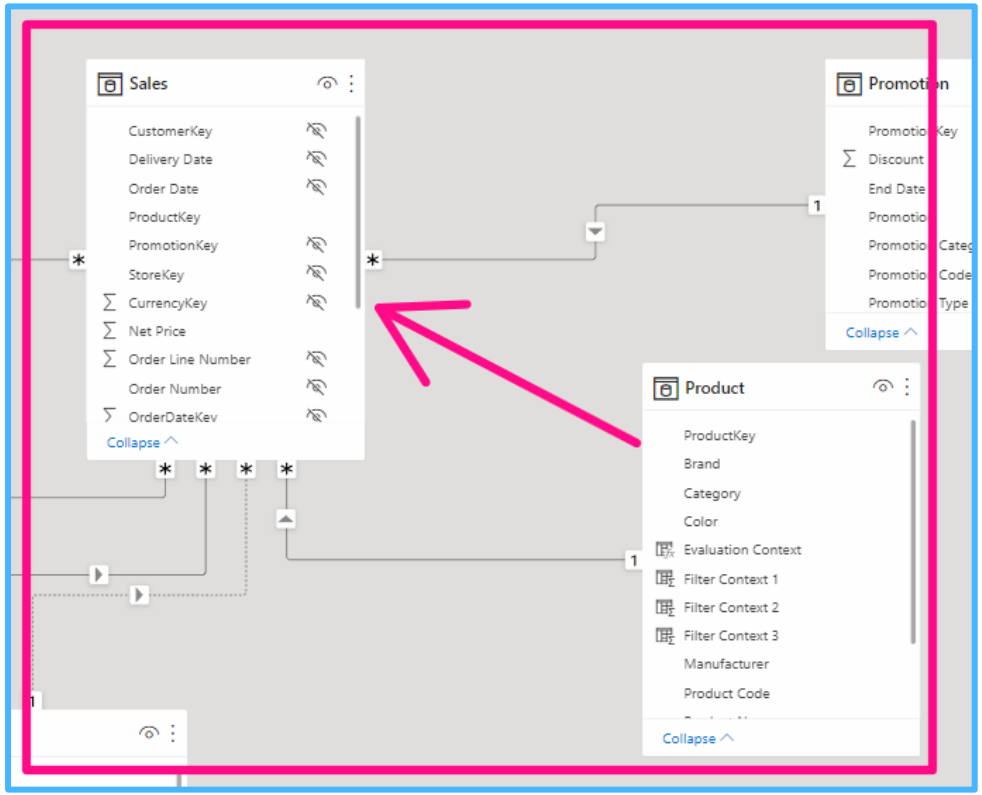
```
Ngày được mua gần nhất =
//Gọi từng ô trên cột Product Key ra
VAR CurrentKey = [ProductKey]
/*
    Qua mỗi ô thì CurrentKey thay đổi liên tục và nó đang đóng vai trò
    như một Parameter và khi đưa vào điều kiện lọc thì điều kiện đó thay đổi
    dựa trên sự thay đổi của các Parameter

*/
Return
    CALCULATE(
        MAX(Sales[Order Date]),
        ALL(),
        Sales[CustomerKey] = CurrentKey
    )
```



3. Filter Context

- **Filter Context** là việc chúng ta trả về các kết quả tính toán trên dữ liệu cho từng ngữ cảnh hiện hành
- Ví dụ ở đây chúng ta thấy bảng Product có Filter đến bảng sales theo hướng **one to many**
- **Product[ProductKey] -> Sales[ProductKey]**
- Do đó chúng ta có thể vào bảng Product tính toán các giá trị cho từng ProductKey. Ví dụ ở đây là tính doanh thu cho từng ProductKey từ bảng Sales.



3. Filter Context and Context Transition

- **Context Transition** là việc chúng ta chuyển từ **Row Context sang Filter Context**
- Như chúng ta đã thấy thì Row Context nó chỉ thực hiện các tính toán theo Row by Row mà thôi. Do đó nếu không có **kích hoạt** Filter Context thì sẽ dẫn đến việc ghi đè một giá trị trên tất các ô
- Lấy ví dụ tính toán doanh thu cho từng ProductKey của bảng Product
- Việc tính toán hoàn toàn hợp lý nhưng kết quả lại ra không đúng. Mặc dù **bảng Product có Filter đến bảng Sales**

```
1  Doanh Thu = SUMX(Sales,Sales[Net Price]*Sales[Quantity])
```

ProductKey	Product Code	Product Name	Doanh Thu
743	0308001	Contoso Rechargeable Battery E100 Black	30591343.9773999
744	0308002	Contoso Dual USB Power Adapter - power adapter E300 Black	30591343.9773999
745	0308003	Contoso Car power adapter M90 Black	30591343.9773999
746	0308004	Contoso Notebook Peripheral Kit M69 Black	30591343.9773999
747	0308005	Contoso Mouse Lock Bundle E200 Black	30591343.9773999
748	0308006	Contoso Education Supplies Bundle E200 Black	30591343.9773999
749	0308007	Contoso Laptop Starter Bundle M200 Black	30591343.9773999
750	0308008	Contoso Education Essentials Bundle M300 Black	30591343.9773999
751	0308009	Contoso Desktop Alternative Bundle E200 Black	30591343.9773999
752	0308010	Contoso Power Inverter - DC to AC power inverter E900 Black	30591343.9773999
753	0308011	Contoso Smart Battery M901 Black	30591343.9773999
754	0308012	Contoso Laptop Cooling Hub notebook fan with 4 ports USB hub E80 Black	30591343.9773999
755	0308013	Contoso Home/Office Laptop Power Adapter E300 Black	30591343.9773999
756	0308014	Contoso USB 2.0 Dock Station docking station M800 Black	30591343.9773999
757	0308015	Contoso Enhanced Capacity Battery M800 Black	30591343.9773999
758	0308016	Contoso Connectivity Starter Kit Smart Buy M680 Black	30591343.9773999
759	0308017	Contoso 90W AC/DC Power Adapter E300 Black	30591343.9773999
760	0308018	Contoso Reserve Pen Tablet Pen E200 Black	30591343.9773999
761	0308019	Contoso USB Data Link-direct connect adapter E600 Black	30591343.9773999
762	0308020	Contoso Primary Extended Capacity Battery Pack - notebook battery X100 Black	30591343.9773999
763	0308021	Contoso Digital camera accessory kit M200 Black	30591343.9773999
764	0308022	Contoso Leather Case - case for digital photo camera X20 Black	30591343.9773999
765	0308023	Contoso Lens cap E80 Black	30591343.9773999
766	0308024	Contoso Battery charger - bike E200 Black	30591343.9773999
767	0308025	Contoso USB Optical Mouse E200 Black	30591343.9773999
768	0308026	Contoso ADSL Modem Splitter/Filter X 1 E100 Black	

Vậy làm sao để kích hoạt Filter Context hay thực hiện Context Transition trong DAX ?

1. Bọc hàm Calculate bên ngoài biểu thức
2. Dùng Measure. Bởi vì Measure có hàm calculate bọc ở ngoài
3. Dùng cách cổ điển để giải quyết (cho filter theo từng ProductKey thay vì kích hoạt)

3. Filter Context and Context Transition

```
Filter Context 1 =
// Mục đích của hàm Calculate là kích hoạt Filter Context
CALCULATE( SUMX(Sales,[Net Price]*[Quantity]))
```

```
Filter Context 2 =
// Cổ điển tương tự như sumif Excel
SUMX(
    FILTER(
        Sales,
        Sales[ProductKey] = EARLIER('Product'[ProductKey])
    ),
    [Net Price] * [Quantity]
)
```

```
DT =
// New Measure
SUMX(Sales,[Net Price]*[Quantity])
```

```
Filter Context 3 = [DT]
```

Filter Context 1	Filter Context 2	Filter Context 3
602	602	602
2,655	2,655	2655
985	985	985
1,067	1,067	1067
554	554	554
1,158	1,158	1158
419	419	419
144	144	144
182	182	182
365	365	365
1,630	1,630	1630
2,094	2,094	2094
2,253	2,253	2253
223	223	223
1,116	1,116	1116
929	929	929
1,188	1,188	1188
15	15	15
353	353	353
607	607	607
447	447	447
251	251	251

CÁC KIẾN THỨC HỖ TRỢ TRONG DAX

DAX STUDIO

Power BI

1. Các kiểu tính toán trong DAX

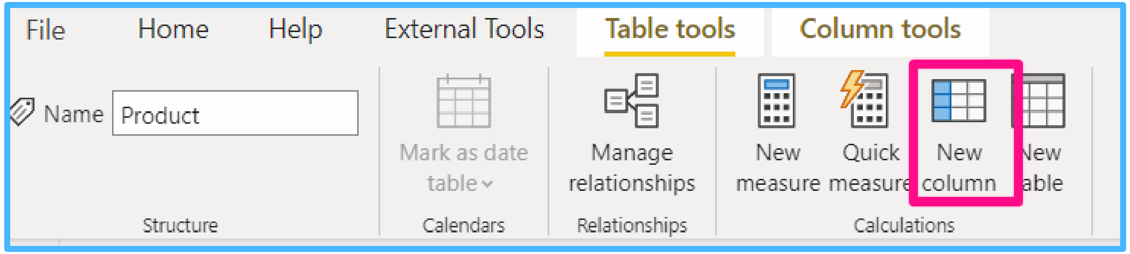
1.New Column

2.New Table

3.New Measure

1. Các kiểu tính toán trong DAX – New Column

New Column: Là việc thêm một cột trong một Table của DAX

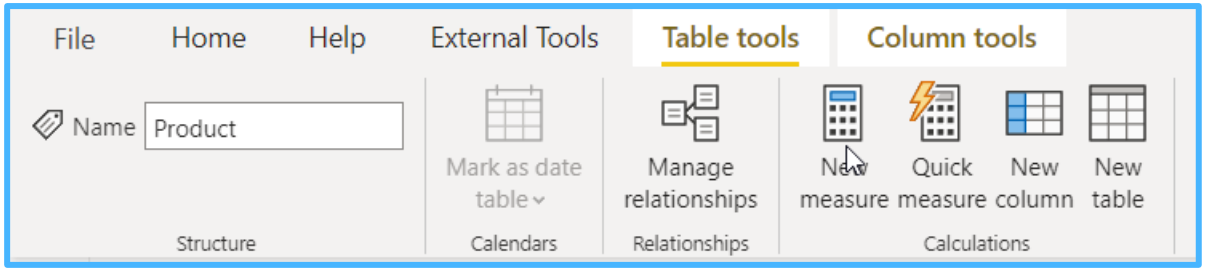



```
HHH =
// Thêm cột HHH vào bảng Prodcut
[Brand] & " "&[Subcategory]
```

HHH
Contoso Computers Accessories
Contoso Computers Accessories
Contoso Computers Accessories
Contoso Computers Accessories
Contoso Computers Accessories
Contoso Computers Accessories
Contoso Computers Accessories
Contoso Computers Accessories
Contoso Computers Accessories
Contoso Computers Accessories
Contoso Computers Accessories
Contoso Computers Accessories
Contoso Computers Accessories
Contoso Computers Accessories
Contoso Computers Accessories
Contoso Computers Accessories
Contoso Computers Accessories
Contoso Computers Accessories
Contoso Computers Accessories
Contoso Computers Accessories
Contoso Computers Accessories
Contoso Computers Accessories

1. Các kiểu tính toán trong DAX – New Measure

New Measure: Là việc tính toán để đưa ra một con số hoặc một chuỗi string
Lưu ý: New Measure là nơi để lưu trữ các biểu thức tính toán khi đưa Measure vào Visual thì nó sẽ sử dụng các biểu thức tính toán đó để tính và trả về kết quả.



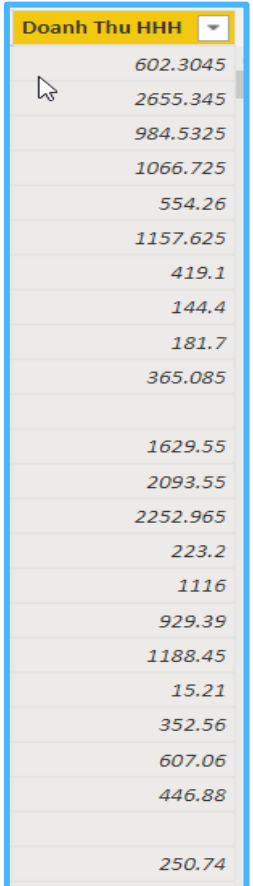

```
1  DT =
2  // New Measure
3  SUMX(Sales,[Net Price]*[Quantity])
```

1. Các kiểu tính toán trong DAX – New Measure

Vậy có thể đặt các Measure vào đâu ???

1. Vào các hàm bảng trong DAX (Summarize, Addcolumns, Filter,.....
2. New Columns

```
1  Doanh Thu HHH =
2  // Đặt Measure vào Column (thêm cột Doanh Thu HHH vào Table Product)
3  [DT]
```



1. Các kiểu tính toán trong DAX – New Measure

Vậy có thể đặt các Measure vào đâu ???

1. **Vào các hàm bảng trong DAX (Summarize, Addcolumns, Filter,…..**

```
Table =
// Tạo Table và thêm Measure vào
    ADDCOLUMNS(
        VALUES('Product'[Category]),
        "@DT",[DT]
    )
```

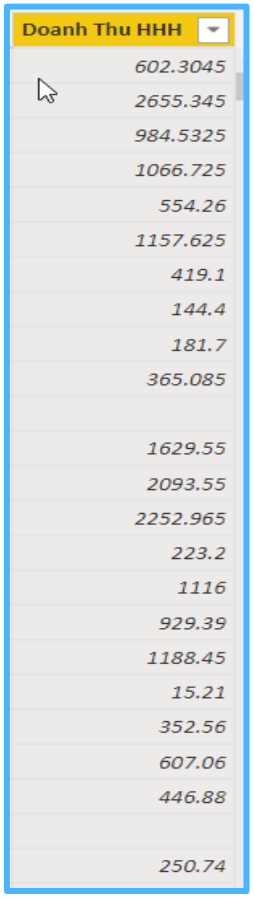
Category	@DT
Audio	384518.160300001
TV and Video	4392768.29230005
Computers	6741548.72819989
Cameras and camcorders	7192581.95279992
Cell phones	1604610.25989999
Music, Movies and Audio Books	314206.738000002
Games and Toys	360652.807400016
Home Appliances	9600457.03849981

1. Các kiểu tính toán trong DAX – New Measure

Vậy có thể đặt các Measure vào đâu ???

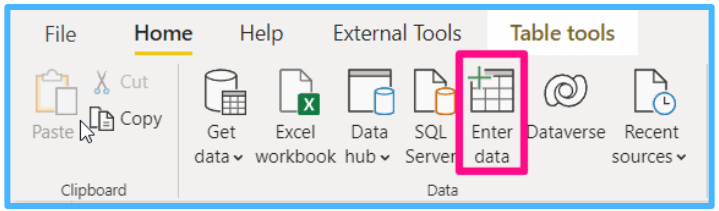
2. New Columns

```
1  Doanh Thu HHH =
2  // Đặt Measure vào Column (thêm cột Doanh Thu HHH vào Table Product)
3  [DT]
```

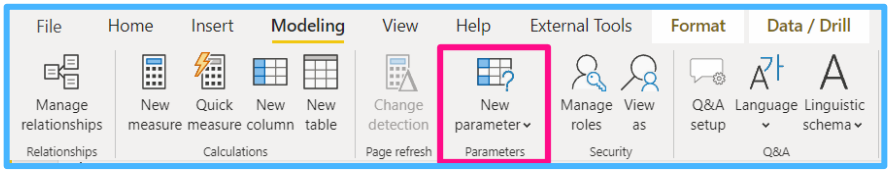


1. Các kiểu tính toán trong DAX – New Table

New Table: Tức là tạo một bảng mời từ một biểu thức DAX hoặc từ các nguồn khác như



1. Enter Data
2. Parameter (Table tạo Parameter)
3. New Table




```
Table =
// Tạo Table và thêm Measure vào
    ADDCOLUMNS(
        VALUES('Product'[Category]),
        "@DT",[DT]
    )
```

Category	@DT
Audio	384518.160300001
TV and Video	4392768.29230005
Computers	6741548.72819989
Cameras and camcorders	7192581.95279992
Cell phones	1604610.25989999
Music, Movies and Audio Books	314206.738000002
Games and Toys	360652.807400016
Home Appliances	9600457.03849981

2. Các kiểu khai báo biến trong DAX

1. **Trên Measure, Column, Table**
2. **Trên bảng ảo**
3. **Khai báo biến toàn cục**
4. **Khai báo biến cục bộ**

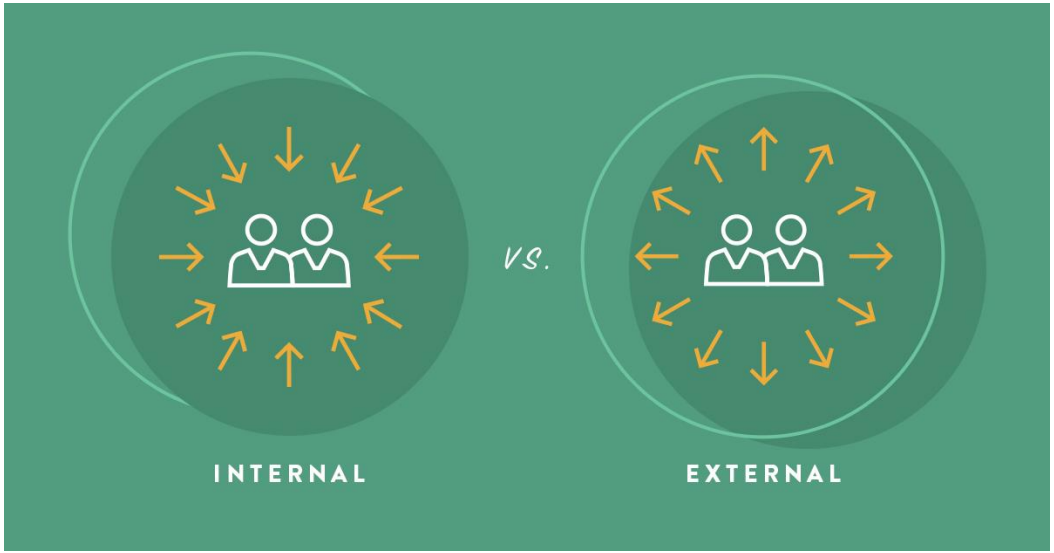
CHAPTER 2: Internal và External Filter (All, Allselected, Removefilter, Allexcept) kết hợp với Caculate, Caculatable

DAX GUIDE

Nguồn tham khảo: DAX Guide

1. Các loại Filter

- Có hai loại Filter mà chúng ta cần quan tâm: **Internal Filter** và **External Filter**
- **Internal Filter:** Là các Filter nằm chung cùng một Visual với Measure
- **External Filter:** Là các Filter nằm ngoài Visual với Measure



1. Các loại Filter – Internal Filter

- Internal Filter xuất phát từ những nơi nào ???
- Từ Visual nằm chung với Measure
- Từ các cột trong bảng Dimension có Filter đến bảng Fact
- Từ việc tạo các bảng ảo trong DAX (với các cột được lấy từ các bảng Dimension có Filter đến bảng Fact)

1. Các loại Filter – Internal Filter

- **Internal Filter xuất phát từ những nơi nào ???**
- Từ Visual nằm chung với Measure
- Ví dụ: Chúng ta tạo một Matrix gồm các trường Product[Name], Product[Brand], Product[Color] và Measure [DT]

Name	Brand	Color	DT
	A. Datum	Azure	91,238.42
Bell, Sebastian	A. Datum	Azure	148.00
Butler, Mya	A. Datum	Azure	187.06
Clark, Jonathan	A. Datum	Azure	187.06
Deng, Terrence	A. Datum	Azure	148.00
Foster, Mariah	A. Datum	Azure	187.06
Gill, Andy	A. Datum	Azure	148.00
Goldberg, Nicolas	A. Datum	Azure	444.00
Gonzales, Sarah	A. Datum	Azure	187.06
Griffin, Luke	A. Datum	Azure	187.06
Henderson, Alyssa	A. Datum	Azure	187.06
King, Wyatt	A. Datum	Azure	148.00
Total			**30,591,343.98**

Các trường Name, Brand, Color được gọi là các **Internal Filter. Measure [DT] có biểu thức là** `SUMX(Sales,[Net Price]*[Quantity])` bị Filter bởi các trường này

1. Các loại Filter – Internal Filter

- **Internal Filter xuất phát từ những nơi nào ???**
- Từ các cột trong bảng Dimension có Filter đến bảng Fact
- Ví dụ: Vào bảng Product tạo cột Doanh Thu HHH

```
1  Doanh Thu HHH =
2  // Đặt Measure vào Column (thêm cột Doanh Thu HHH vào Table Product)
3  [DT]
```

Doanh Thu HHH
602.3045
2655.345
984.5325
1066.725
554.26
1157.625
419.1
144.4
181.7
365.085
1629.55
2093.55
2252.965
223.2
1116
929.39
1188.45
15.21
352.56
607.06
446.88
250.74

-> Các trường trong bảng Product đều được gọi là các Internal Filter

1. Các loại Filter – Internal Filter

- **Internal Filter xuất phát từ những nơi nào ???**
- Từ việc tạo các bảng ảo trong DAX (với các cột được lấy từ các bảng Dimension có Filter đến bảng Fact)

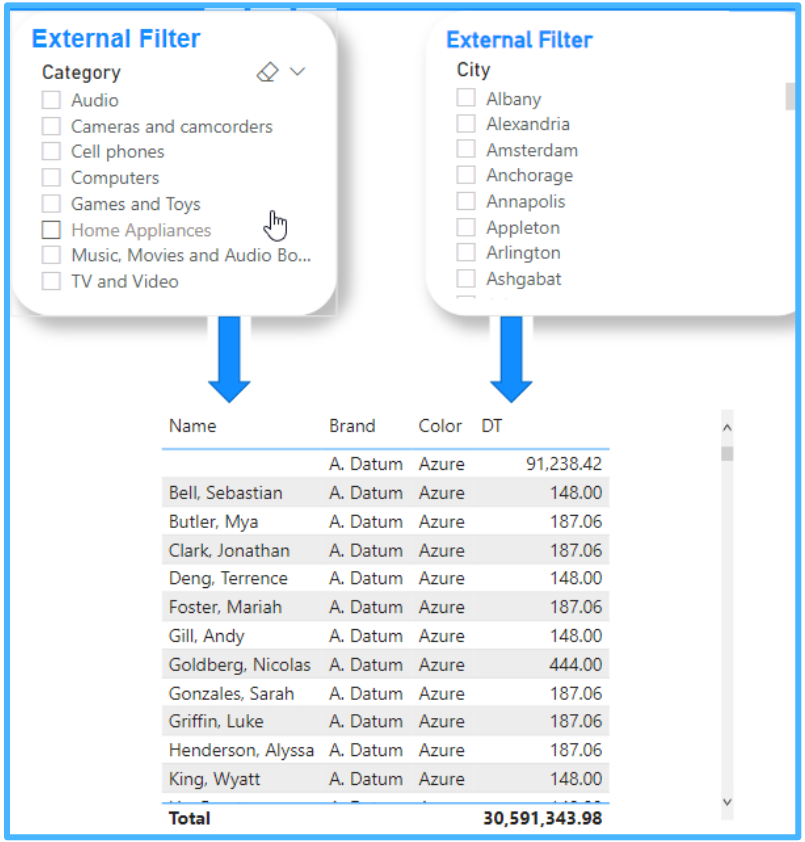
```
Table =
// Tạo Table và thêm Measure vào
    ADDCOLUMNS(
        VALUES('Product'[Category]),
        "@DT",[DT]
    )
```

Category	@DT
Audio	384518.160300001
TV and Video	4392768.29230005
Computers	6741548.72819989
Cameras and camcorders	7192581.95279992
Cell phones	1604610.25989999
Music, Movies and Audio Books	314206.738000002
Games and Toys	360652.807400016
Home Appliances	9600457.03849981

-> Trường Category được gọi là Internal Filter

1. Các loại Filter – External Filter

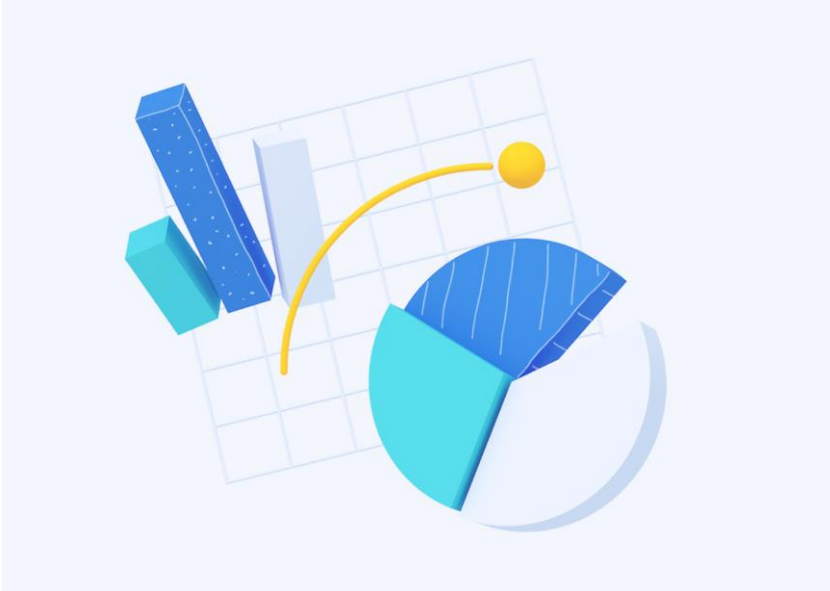
- **External Filter xuất phát từ những nơi nào ???**
- Từ các Visual ngoài Measure
- Ví dụ: Tạo một Matrix với các trường Name, Brand, Color và Measure DT
- Các trường Name, Brand, Color đều được gọi là Internal Filter bởi vì chúng **nằm trong** một Visual với Measure [DT]
- Các Slicer bên ngoài Matrix hoặc các biểu đồ… đều được gọi là các External Filter và chúng có các động đến Measure [DT] bởi vì chúng **nằm ngoài** Visual với Measure [DT]



Name	Brand	Color	DT
	A. Datum	Azure	91,238.42
Bell, Sebastian	A. Datum	Azure	148.00
Butler, Mya	A. Datum	Azure	187.06
Clark, Jonathan	A. Datum	Azure	187.06
Deng, Terrence	A. Datum	Azure	148.00
Foster, Mariah	A. Datum	Azure	187.06
Gill, Andy	A. Datum	Azure	148.00
Goldberg, Nicolas	A. Datum	Azure	444.00
Gonzales, Sarah	A. Datum	Azure	187.06
Griffin, Luke	A. Datum	Azure	187.06
Henderson, Alyssa	A. Datum	Azure	187.06
King, Wyatt	A. Datum	Azure	148.00
Total			**30,591,343.98**

2. Các hàm bỏ bộ lọc

1. Hàm All, Removefilter
2. Hàm Allselected
3. Hàm Allexcept
4. Cách định nghĩa lại bộ lọc
5. Cách sử dụng các hàm bỏ bộc lọc với hàm CALCULATE và CALCULATETABLE



2. Các hàm bỏ bộ lọc – ALL, REMOVEFILTER

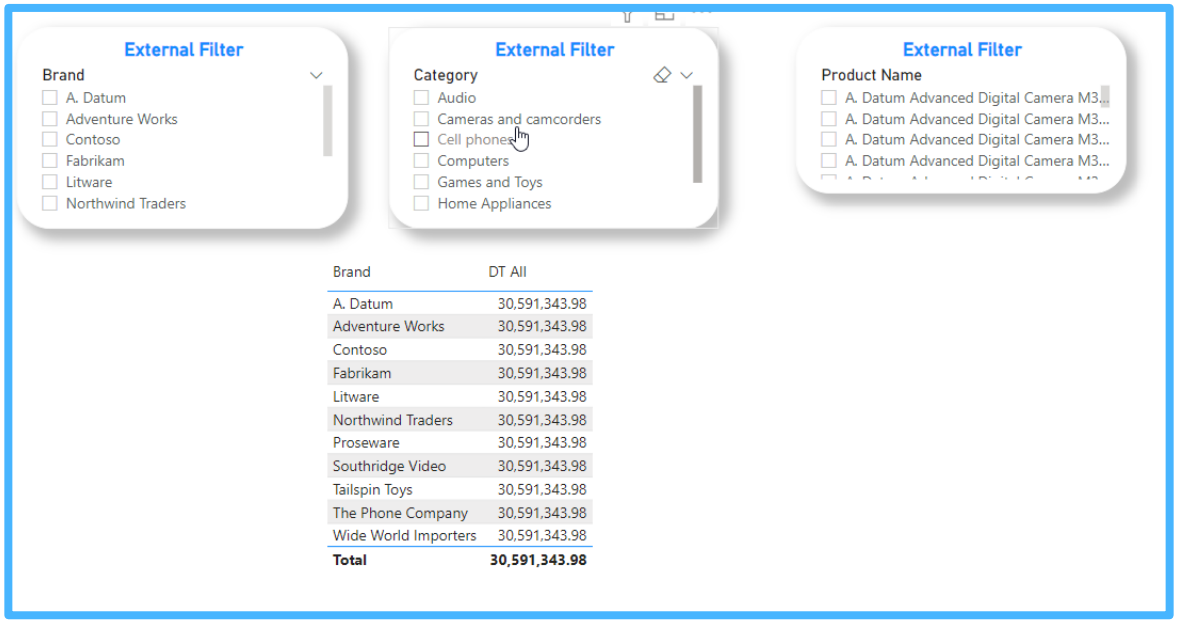
- Nhắc lại khái niệm Internal và External Filter ??
- Hàm All và hàm Removefilter mặc dù tên hàm khác nhau nhưng về bản chất hầu như không có sự khác biệt.
- Hàm All, Removefilter: đều bỏ các Internal Filter và External Filter của một column trong một table, hoặc của tất cả các column trong một table hoặc tất các table.
- Ví dụ: Chúng ta tạo một Matrix với trường Product[Brand] và measure [DT All]

Brand	DT All
A. Datum	30,591,343.98
Adventure Works	30,591,343.98
Contoso	30,591,343.98
Fabrikam	30,591,343.98
Litware	30,591,343.98
Northwind Traders	30,591,343.98
Proseware	30,591,343.98
Southridge Video	30,591,343.98
Tailspin Toys	30,591,343.98
The Phone Company	30,591,343.98
Wide World Importers	30,591,343.98
Total	**30,591,343.98**

```
DT All =
    CALCULATE(
        [DT],
        ALL('Product')
    )
```

2. Các hàm bỏ bộ lọc – ALL, REMOVEFILTER

- Chúng ta tiến hành tạo các External Filter từ các trường Product[Brand], Product[Category], Product[Name]



Brand	DT All
A. Datum	30,591,343.98
Adventure Works	30,591,343.98
Contoso	30,591,343.98
Fabrikam	30,591,343.98
Litware	30,591,343.98
Northwind Traders	30,591,343.98
Proseware	30,591,343.98
Southridge Video	30,591,343.98
Tailspin Toys	30,591,343.98
The Phone Company	30,591,343.98
Wide World Importers	30,591,343.98
Total	**30,591,343.98**

2. Các hàm bỏ bộ lọc – ALL, REMOVEFILTER

Khi nhấn các slicer từ các External Filter thì các giá trị trong matrix có thay đổi hay không ???

2. Các hàm bỏ bộ lọc – ALLSELECTED

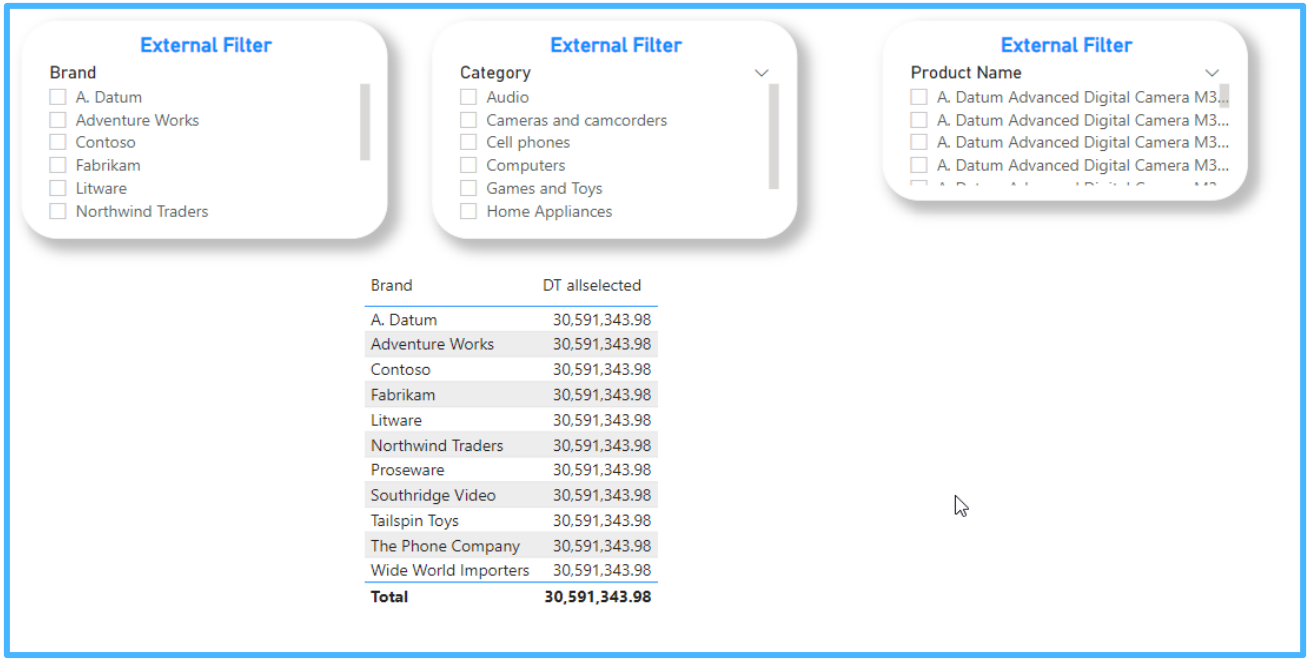
- Nhắc lại khái niệm Internal ???
- Hàm Allselected r: bỏ các Internal Filter của một column trong một table, hoặc của tất cả các column trong một table hoặc tất các table.
- Ví dụ: Chúng ta tạo một Matrix với trường Product[Brand] và measure [DT Allselected]

```
DT Allselected =
    CALCULATE(
        [DT],
        ALLSELECTED('Product')
    )
```

Brand	DT Allselected
A. Datum	30,591,343.98
Adventure Works	30,591,343.98
Contoso	30,591,343.98
Fabrikam	30,591,343.98
Litware	30,591,343.98
Northwind Traders	30,591,343.98
Proseware	30,591,343.98
Southridge Video	30,591,343.98
Tailspin Toys	30,591,343.98
The Phone Company	30,591,343.98
Wide World Importers	30,591,343.98
Total	**30,591,343.98**

2. Các hàm bỏ bộ lọc – ALLSELECTED

- Chúng ta tiến hành tạo các External Filter từ các trường Product[Brand], Product[Category], Product[Name]



Brand	DT allselected
A. Datum	30,591,343.98
Adventure Works	30,591,343.98
Contoso	30,591,343.98
Fabrikam	30,591,343.98
Litware	30,591,343.98
Northwind Traders	30,591,343.98
Proseware	30,591,343.98
Southridge Video	30,591,343.98
Tailspin Toys	30,591,343.98
The Phone Company	30,591,343.98
Wide World Importers	30,591,343.98
Total	**30,591,343.98**

2. Các hàm bỏ bộ lọc – ALLSELECTED

Khi nhấn các slicer từ các External Filter thì các giá trị trong matrix có thay đổi hay không ???

3. Các hàm bỏ bộ lọc – ALLEXCEPT

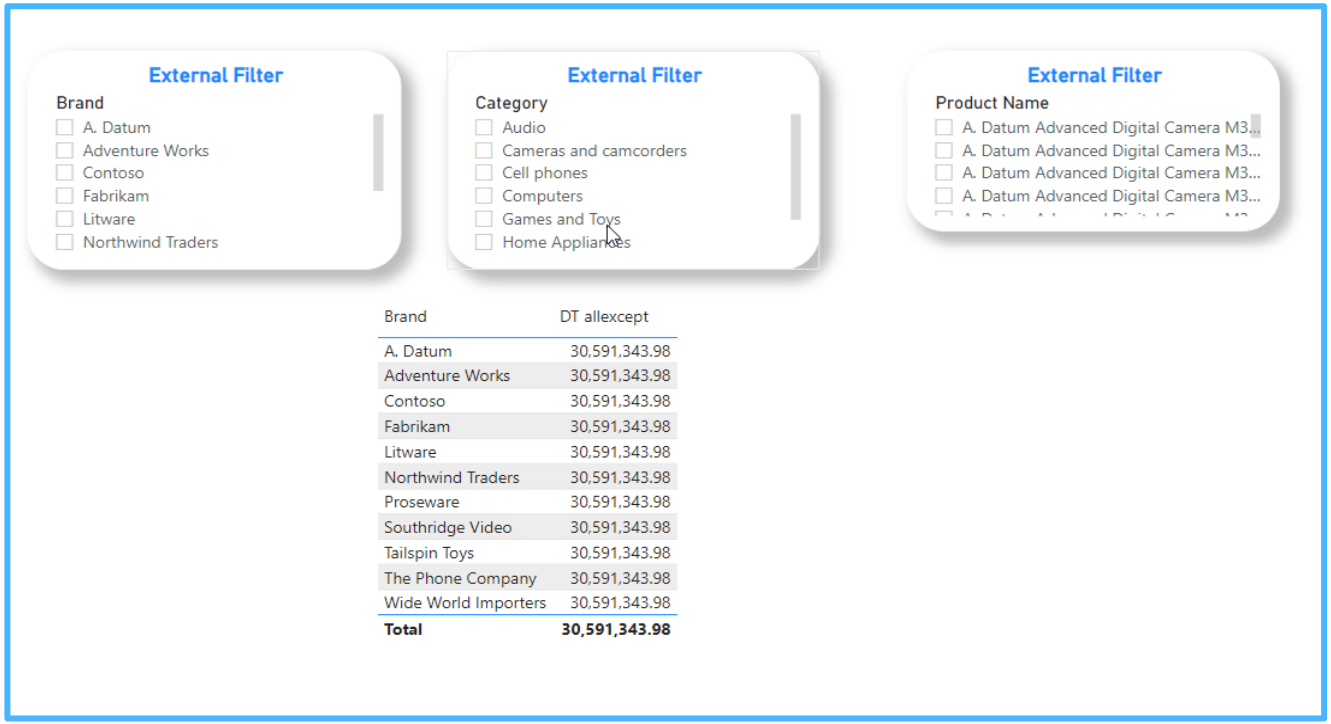
- Nhắc lại khái niệm Internal và external filter ???
- Hàm ALLEXCEPT : bỏ các Internal Filter và External của một một table và giữ lại một hoặc của tất cả các column trong một table hoặc tất các table và giữa một số bộ lọc
- Ví dụ: Chúng ta tạo một Matrix với trường Product[Brand] và measure [DT Allexcept]

```
DT allexcept =
    CALCULATE(
        [DT],
        ALLEXCEPT('Product','Product'[Category])
    )
```

Brand	DT allexcept
A. Datum	30,591,343.98
Adventure Works	30,591,343.98
Contoso	30,591,343.98
Fabrikam	30,591,343.98
Litware	30,591,343.98
Northwind Traders	30,591,343.98
Proseware	30,591,343.98
Southridge Video	30,591,343.98
Tailspin Toys	30,591,343.98
The Phone Company	30,591,343.98
Wide World Importers	30,591,343.98
Total	**30,591,343.98**

3. Các hàm bỏ bộ lọc – ALLEXCEPT

- Chúng ta tiến hành tạo các External Filter từ các trường Product[Brand], Product[Category], Product[Name]



Brand	DT allexcept
A. Datum	30,591,343.98
Adventure Works	30,591,343.98
Contoso	30,591,343.98
Fabrikam	30,591,343.98
Litware	30,591,343.98
Northwind Traders	30,591,343.98
Proseware	30,591,343.98
Southridge Video	30,591,343.98
Tailspin Toys	30,591,343.98
The Phone Company	30,591,343.98
Wide World Importers	30,591,343.98
Total	**30,591,343.98**

3. Các hàm bỏ bộ lọc – ALLEXCEPT

Khi nhấn các slicer từ các External Filter thì các giá trị trong matrix có thay đổi hay không ???.

Liệu có External Filter nào ảnh hưởng đến Measure và làm thay đổi giá trị không

LƯU Ý QUAN TRỌNG KHI SỬ DỤNG CÁC HÀM BỎ BỘ LỌC

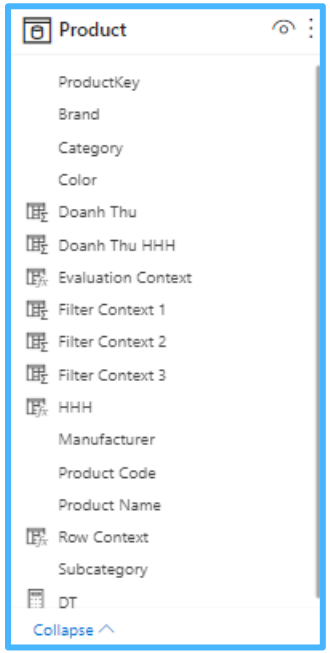
- Trong trường hợp chúng ta sử dụng các hàm hàm bỏ bộ lọc ngoại trừ hàm ALLEXCEPT nếu không chỉ định bỏ bộc lọc ở bất kì bảng nào hoặc cột nào thì nó mặc định là bỏ tất các Filter đang Filter đến Measure mà chúng ta đang tính toán
- **Ví dụ: ALL(), ALLSELECTED(), REMOVEFILTERS()**
- Từ bản chất của các hàm mà chúng ta tự suy ra bỏ các Internal hay External Filter đang Filter đến Measure

4. Cách tái định nghĩa bộ lọc

- Sau khi sử dụng các hàm bỏ bộ lọc chúng ta có thể **sử dụng thêm** các hàm **Values, Distict** và tái định nghĩa lại bộ lọc
- Tái định nghĩa lại bộ lọc được hiểu như thế vào: Ví dụ chúng ta đã bỏ các bộ lọc của bảng Product bằng hàm ALL hoặc hàm ALLSELECTED (ALL(Product), ALLSELECTD(Product))
- Lúc này chúng ta chỉnh lại và thêm vào đối số filter của hàm CALCULATE bằng hàm Values hoặc Distict(Product[Category]) thì Measure đó bị lọc bởi trường Category trong bảng Product.
- Nhưng lưu ý các trường trong bảng Product có liên kết với nhau và tạo thành các Dependent Filter tức là trường Product[Color] bị Filter thì trường Product[Category] cũng bị Filter theo

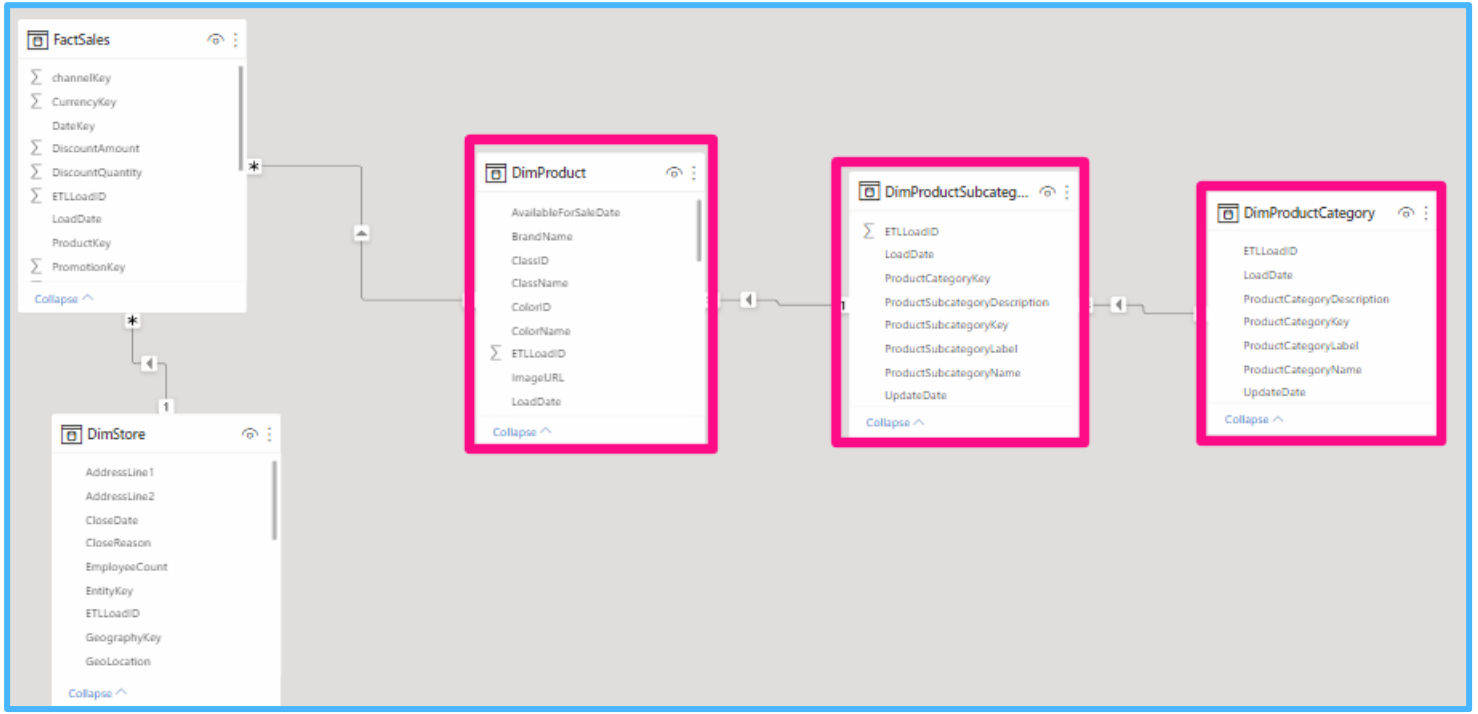
4. Cách tái định nghĩa bộ lọc – Dependent Filter

- Dependent Filter: là việc lọc filter ở trường này nó sẽ filter đến trường kia
- Ví dụ: Filter ở trường Product[Category] nó sẽ ảnh hưởng đến các trường Product[Color], Product[Brand], Product[SubCategory]…..
- Như trong bảng Product các trường đều có liên kết với nhau việc Filter ở trường này sẽ ảnh hưởng đến trường khác
- Hoặc chúng ta có thể thấy trong Data Model dạng Snowflake…



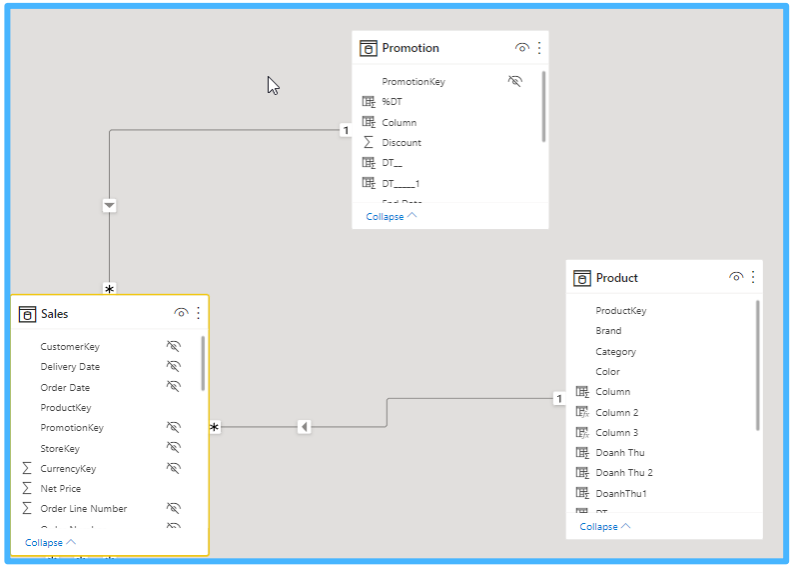
4. Cách tái định nghĩa bộ lọc – Dependent Filter

Mô hình Snowflake



4. Cách tái định nghĩa bộ lọc – Independent Filter

- Independent Filter: là việc lọc filter ở trường này nó sẽ không filter đến các trường của bảng kia
- Ví dụ: Filter ở trường Product[Category], Product[Color], Product[Brand], Product[SubCategory].. Nó sẽ không ảnh hưởng đến các trường của bảng Promotion
- Bởi vì các trường trong bảng Product không nằm chung bảng với các trường của bảng Promotion và hai bảng không có liên kết relationship với nhau do đó các Filter từ các trường của bảng Promotion hoàn toàn độc lập so với các trường của bảng Product



5. Cách sử dụng các hàm bỏ bộ lọc với hàm CALCULATE và CALCULATETABLE

CALCULATE DAX Function (Filter) CONTEXT TRANSITION ⓘ

Syntax

```
CALCULATE ( <Expression> [, <Filter> [, <Filter> [, … ] ] ] )
```

- Theo DAX Guide: hàm Calculate có chức năng điều chỉnh ngữ cảnh bằng Filter
- Filter ở đây được hiểu là đưa vào một bộ lọc có thể sử dụng hàm bảng như Filter, Intersect, Values, Distict (các hàm có liên quan đến định nghĩa lại bộ lọc)
- Và được sử dụng chung với các hàm bỏ bộ lọc như ALL, REMOVEFILTERS, ALLSELECTED, ALLEXCEPT..

```
Measure 2 =
    CALCULATE(
        [Doanh Thu],
        FILTER(
            'Date',
            'Date'[Calendar Year] = "CY 2008"
        ),
        ALLSELECTED()
    )
```

5. Cách sử dụng các hàm bỏ bộ lọc với hàm CALCULATE và CALCULATETABLE

CALCULATETABLE

DAX Function (Filter) CONTEXT TRANSITION ⓘ

Syntax

```
CALCULATETABLE ( <Table> [, <Filter> [, <Filter> [, … ] ] ] )
```

- Tương tự như hàm CALCULATE nhưng tham số đầu tiên là một Table chứ không phải là một Expression để tính toán ra giá trị.

5. Cách sử dụng các hàm bỏ bộ lọc với hàm CALCULATE và CALCULATETABLE

Cách hoạt động của tham số Filter trong các hàm CALCULATE và CALCULATETABLE

Filter	Optional Repeatable	A boolean (True/False) expression or a table expression that defines a filter.

- Đối với các điều kiện True thì nó sẽ trả về kết quả, ví dụ như Filter các Customer[CustomerType] là Company chẳng hạn hoặc những người có thu nhập cao….
- Định nghĩa bộ lọc ở đây có thể hiện là truyền một table chứa các list bộ lọc hoặc các hàm tạo bộ lọc ảo như USERELATIONSHIP (active Relationship), TREATAS(Tạo bộ lọc ảo)…., hoặc các hàm bỏ bộ lọc

CHAPTER 3: CÁC HÀM TÍNH TOÁN TRONG DAX

Các hàm X và các hàm thông thường trong DAX

DAX GUIDE

Nguồn tham khảo: DAX Guide

CHAPTER 3: CÁC HÀM TÍNH TOÁN TRONG DAX

Các hàm X và các hàm thông thường trong DAX

1. Các hàm X thường dùng
    - SUMX
    - MAXX
    - MINX
    - AVERAGEX

2. Các hàm thông thường
    - COUNTROWS
    - DISTINCTCOUNT
    - COUNTROWS phối hợp với DISTINCT

TẠI SAO NÊN DÙNG SUMX THAY VÌ DÙNG SUM ???

- Bởi vì hàm SUM chỉ sử dụng với một cột chúng ta có thể thấy qua Syntax của hàm SUM

Syntax

```
SUM ( <ColumnName> )
```

- Do đó chúng ta không thể cộng trừ nhân chừ khi sử dụng hàm SUM
- Hàm SUM chỉ sử dụng với các bảng vật lý
- Còn SUMX chúng ta có thể cộng trừ nhân chia được điển hình là chúng ta hay tính doanh thu và tính toán trên cả bảng vật lý, bảng ảo khai báo bằng biến
- SUMX(Sales, Sales[Net Price] * Sales[Quantity])

CÁCH VẬN HÀNH CỦA CÁC HÀM X

Ví dụ: Chúng ta có một Matrix với MEASURE Doanh Thu được viết như thế này

```
SUMX(Sales,[Net Price]*[Quantity])
```

Quy trình 1: Bảng Sales sẽ bị lọc bởi đồng thời hai bảng Date và Product
Trường Date[Calendar Year] tác động lên bảng Date làm thay đổi bảng Date sau đó Filter đến bảng Sales tương tự với bảng Product

Quy trình 2: Sau khi bảng Sales bị Filter bởi từng ngữ cảnh của Date[Calendar Year] và Product[Color] thì tiến hành tính toán nhân từng Net Price với Quantity lại với nhau theo Row Context

Quy trình 3: Tổng tất cả các giá trị và trả về ngữ cảnh hiện hành

Calendar Year	Color	Doanh Thu
CY 2007	Azure	27,895
CY 2008	Azure	33,072
CY 2009	Azure	36,423
CY 2007	Black	1,988,876
CY 2008	Black	1,955,354
CY 2009	Black	1,915,836
CY 2007	Blue	715,829
CY 2008	Blue	998,866
CY 2009	Blue	720,750
CY 2007	Brown	201,368
CY 2008	Brown	405,671
CY 2009	Brown	422,470
CY 2007	Gold	133,740
CY 2008	Gold	122,991
CY 2009	Gold	104,765
CY 2007	Green	429,480
CY 2008	Green	582,339
CY 2009	Green	391,365
CY 2007	Grey	1,419,966
CY 2008	Grey	1,016,073

1. Các hàm X thông dụng - SUMX

Syntax

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

- Table: Truyền một bảng vào có thể là một **bảng ảo** nằm trên biến hoặc là một bảng **vật lý**
- Expression: Được thực hiện dưới dạng Row Context, tính toán các cột dựa nằm trong Table đó

```
Doanh Thu = SUMX(Sales,[Net Price]*[Quantity])
```

```
Doanh Thu =
VAR A  =
    ADDCOLUMNS(
        SUMMARIZE(Sales,Customer[CountryRegion]),
        "@DT",
        [Doanh_Th]
    )
Return
    SUMX(A,[@DT])
```

1. Các hàm X thông dụng – AVERAGEX, MINX, MAXX

- **Các hàm trên cũng có cách vận hành tương tự như hàm SUMX**
- AVERAGEX: Tính giá trị trung bình trong một bảng vật lý hoặc một bảng ảo hoặc bảng vật lý
- MINX: Lấy ra giá trị nhỏ nhất trong một bảng vật lý hoặc một bảng ảo hoặc bảng vật lý
- MAXX: Lấy ra giá trị lớn nhất trong một bảng vật lý hoặc một bảng ảo hoặc bảng vật lý

2. Các hàm thông thường hay sử dụng - COUNTROWS

Syntax

```
COUNTROWS ( [<Table>] )
```

- Hàm COUNTROWS được dùng để đếm số lượng dòng trong một bảng thực hoặc một bảng ảo được tạo ra bằng cách khai báo biến
- Bảng có thể chứa một hoặc nhiều cột

```
So GD = COUNTROWS(Sales)
```

```
So GD_T =
VAR A =
    ADDCOLUMNS(
        SUMMARIZE(Sales,Customer[CountryRegion]),
        "@DT",
        [Doanh Thu]
    )
Return
    COUNTROWS(A)
```

2. Các hàm thông thường hay sử dụng - DISTINCTCOUNT

Syntax

```
DISTINCTCOUNT ( <ColumnName> )
```

- Hàm DISTINCTCOUNT được dùng để đếm số lượng dòng riêng biệt (tức là đã remove duplicate rồi) trong một bảng thực
- Bảng ở đây có thể chứa một hoặc nhiều cột

```
So Loai SP Da ban = DISTINCTCOUNT(Sales[ProductKey])
```

Vậy làm sao có thể đếm các giá trị riêng biệt trên một bảng ảo khi không thể sử dụng hàm DISTINCTCOUNT??

CHÚNG TA CÓ THỂ DÙNG KẾT HỢP HÀM DISTINCT VỚI HÀM COUNTROWS

```
Countrows + Distinct =
VAR A =
    DISTINCT(
        SELECTCOLUMNS(
            ADDCOLUMNS(
                SUMMARIZE(Sales,Customer[CountryRegion]),
                "@DT",
                [Doanh Thu]
            ),
            "@Country",
            [CountryRegion]
        )
    )
Return
    COUNTROWS(A)
```

CHAPTER 4: CÁC HÀM BẢNG TRONG DAX

DAX GUIDE

Nguồn tham khảo: DAX Guide

DAX.do

CHAPTER 4: CÁC HÀM BẢNG TRONG DAX

1. Các sử dụng để nhóm, lấy và lọc dữ liệu
   - SUMMARIZE
   - ADDCOLUMNS
   - FILTER
   - GROUPBY
   - CROSSJOIN

2. Các hàm tìm điểm chung điểm khác biệt và nối dữ liệu
   - INTERSECT
   - EXCEPT
   - UNION

3. Các hàm liên quan đến bộc lọc (Values, Distinct)

DAX GUIDE

Nguồn tham khảo: DAX Guide

DAX.do

HÀM BẢNG LÀ GÌ ???

Hàm bảng là những hàm được sử dụng cho việc ra các bảng có thể sử dung trong New Table, tạo bằng việc khai báo biến trong New Measure, New Column, New Table…

1. Các hàm để nhóm, lấy và lọc dữ liệu – SUMMARIZE

Syntax

```
SUMMARIZE ( <Table> [, <GroupBy_ColumnName> [, [<Name>] [,
[<Expression>] [, <GroupBy_ColumnName> [, [<Name>] [,
[<Expression>] [, … ] ] ] ] ] ] ] )
```

- Theo DAX Guide hàm SUMMARIZE là hàm tạo một table tóm tắt được nhóm theo các cột chỉ định
- Các cột chỉ định ở đây sẽ được ưu tiên từ trái qua phải tức là nếu chúng ta đặt cột chỉ định nào trước thì sẽ ưu tiên nhóm theo cột đó trước
- Bên cạnh đó chúng ta có thể thêm một cột mới với hàm SUMAMRIZE và đặt các biểu thức tính toán như SUMX, AVERAGEX… và biểu thức nó sẽ tự động chuyển từ ROW CONTEXT - > FILTER CONTEXT cho dù chúng ta không bọc hàm CALCULATE (Nhưng hiệu năng không tốt nên không được khuyến nghị khi sử dụng)

1. Các hàm để nhóm, lấy và lọc dữ liệu – SUMMARIZE

- Hàm SUMMARIZE cho phép thêm một Column vào và đưa các biểu thức tính toán để tính toán các giá trị cho Column đó
- Việc đưa biểu thức vào như vậy không cần chuyển đổi từ Row Context sang Filter Context mà nó tự động chuyển. Do đó chúng ta không cần dùng các phương pháp Context Transition
- Nhưng cách này ít được sử dụng vì nếu lương data lớn thì nó sẽ ảnh hưởng rất nhiều đến hiệu năng. Do đó dùng ADDCOLUMNS để thay thế là một giải pháp hữu hiệu

```
EVALUATE
    SUMMARIZE(
        Sales,
        'Product'[Subcategory],
        'Product'[Category],
        "@DT",
        SUMX(Sales,Sales[Net Price]*Sales[Quantity])
    )
```

Subcategory	Category	@DT
MP4&MP3	Audio	170194.002200001
Recording Pen	Audio	89873.3650000003
Bluetooth Headphones	Audio	124450.793099999
Televisions	TV and Video	1834257.0543
VCD & DVD	TV and Video	428571.267
Home Theater System	TV and Video	1525526.261
Car Video	TV and Video	604413.709999998
Laptops	Computers	1925105.27650001
Desktops	Computers	1017127.267

1. Các hàm để nhóm, lấy và lọc dữ liệu – ADDCOLUMNS

- Hàm ADDCOLUMNS cho phép thêm một hoặc nhiều cột mới, đặt tên cho nó và dung một biểu thức tính toán để tính toán các giá trị cho cột đó.
- Lưu ý khi đưa các biểu thức tính toán vào tham số Expression thì chúng ta phải thực hiện Context Transition. Bởi vì tham số này chỉ hoạt động dưới dạng Row Context chứ không có Filter Context
- Có 2 cách thường sử dụng: đó là bọc hàm calculate bên ngoài biểu thức hoặc là xây dựng Measure rồi đưa trực tiếp vào

Syntax

```
ADDCOLUMNS ( <Table>, <Name>, <Expression> [, <Name>,
<Expression> [, … ] ] )
```

PARAMETER	ATTRIBUTES	DESCRIPTION
Table ITERATOR		The table to which new columns are added.
Name	Repeatable	The name of the new column to be added.
Expression ROW CONTEXT	Repeatable	The expression for the new column to be added.

1. Các hàm để nhóm, lấy và lọc dữ liệu – ADDCOLUMNS

Trường hợp không bọc CALCULATE

```
EVALUATE
    ADDCOLUMNS(
        SUMMARIZE(
            Sales,
            'Product'[Subcategory],
            'Product'[Category]
        ),
        "@DT",
        SUMX(Sales,Sales[Net Price]*Sales[Quantity])
    )
```

Subcategory	Category	@DT
MP4&MP3	Audio	30591343.9773999
Recording Pen	Audio	30591343.9773999
Bluetooth Headphones	Audio	30591343.9773999
Televisions	TV and Video	30591343.9773999
VCD & DVD	TV and Video	30591343.9773999
Home Theater System	TV and Video	30591343.9773999
Car Video	TV and Video	30591343.9773999
Laptops	Computers	30591343.9773999
Desktops	Computers	30591343.9773999

Trường hợp bọc CALCULATE

```
EVALUATE
    ADDCOLUMNS(
        SUMMARIZE(
            Sales,
            'Product'[Subcategory],
            'Product'[Category]
        ),
        "@DT",
        CALCULATE(
            SUMX(Sales,Sales[Net Price]*Sales[Quantity])
        )
    )
```

Subcategory	Category	@DT
MP4&MP3	Audio	170194.002200001
Recording Pen	Audio	89873.3650000003
Bluetooth Headphones	Audio	124450.793099999
Televisions	TV and Video	1834257.0543
VCD & DVD	TV and Video	428571.267
Home Theater System	TV and Video	1525526.261
Car Video	TV and Video	604413.709999998
Laptops	Computers	1925105.27650001
Desktops	Computers	1017127.267

1. Các hàm để nhóm, lấy và lọc dữ liệu – FILTER

- Theo DAX Guide: Hàm Filter là hàm trả về một bảng mà các giá trị đã được lọc theo điều kiện
- Ví dụ: Lọc ra danh sách những khách hàng có doanh thu > 50000….
- Table: Truyền vào nó một Table từ bảng vật lý hoặc các từ các hàm bảng hoặc từ các biến tạo ra bảng ảo
- Filter Expression là các điều kiện lọc của chúng ta (ví dục lọc ra những khách hàng sống tại US..)
- Lưu ý Thay vì dung ADDCOLUMNS để thêm cột Doanh Thu thì chúng ta có thể đưa Measure vào thẳng luôn

Syntax

```
FILTER ( <Table>, <FilterExpression> )
```

PARAMETER	ATTRIBUTES	DESCRIPTION
Table ITERATOR		The table to be filtered.
FilterExpression ROW CONTEXT		A boolean (True/False) expression that is to be evaluated for each row of the table.

1. Các hàm để nhóm, lấy và lọc dữ liệu – FILTER

Trường hợp sử dụng ADDCOLUMNS

```
DEFINE
    VAR A =
        ADDCOLUMNS(
            VALUES('Product'[Category]),
            "@DT",
            [Doanh Thu]
        )
EVALUATE
    FILTER(
        A,
        [@DT] > 30000000
    )
```

Category	@DT
TV and Video	30185201.7869
Computers	30591343.9774
Cameras and camcorders	30591343.9774
Cell phones	30403409.5189
Home Appliances	30591343.9774

Trường hợp không sử dụng

```
EVALUATE
    FILTER(
        VALUES('Product'[Category]),
        [Doanh Thu] > 30000000
    )
```

Category
TV and Video
Computers
Cameras and camcorders
Cell phones
Home Appliances

1. Các hàm để nhóm, lấy và lọc dữ liệu – GROUPBY

- Theo DAX Guide: Hàm Groupby là hàm tạo ra một bảng mà được nhóm lại theo cột chỉ định
- Trông khái niệm có thể giống với hàm SUMMARIZE tuy nhiên trong thực tế thì lại khác
- Bởi vì hàm SUMMARIZE chỉ thao tác tính nhóm lại theo cột chỉ định một lần với các bảng vật lý và nếu sử dụng với bảng ảo thì nó không hiểu các ngữ cảnh hiện hành của cột được chỉ định nhóm lại
- Hàm Groupby vừa dùng được cho bảng ảo và bảng thực
- Thường hay sử dụng với hàm CURRENTGROUP()

Syntax

```
GROUPBY ( <Table> [, <GroupBy_ColumnName> [, [<Name>] [,
[<Expression>] [, <GroupBy_ColumnName> [, [<Name>] [,
[<Expression>] [, … ] ] ] ] ] ] ] )
```

PARAMETER	ATTRIBUTES	DESCRIPTION
Table		The input table.
GroupBy_ColumnName	Optional Repeatable	A column to group by.
Name	Optional Repeatable	A column name to be added.
Expression	Optional Repeatable	The expression of the new column.

1. Các hàm để nhóm, lấy và lọc dữ liệu – GROUPBY

Trường hợp sử dụng GROUPBY

```
DEFINE
    VAR A =
        ADDCOLUMNS(
            SUMMARIZE(
                Sales,
                'Product'[Category],
                'Date'[Calendar Year]
            ),
            "@DT",
            [Doanh Thu]
        )
EVALUATE
    GROUPBY(A,[Category],"@DT",SUMX(CURRENTGROUP(),[@DT]))
```

Category	@DT
Audio	29786953.4364002
TV and Video	30022164.4369002
Computers	30591343.9774002
Cameras and camcorders	30428306.6274002
Cell phones	30164392.5269002
Music, Movies and Audio Books	29221804.2963002
Games and Toys	28849752.9296002
Home Appliances	30428306.6274002

Trường hợp không sử dụng

```
DEFINE
    VAR A =
        ADDCOLUMNS(
            SUMMARIZE(
                Sales,
                'Product'[Category],
                'Date'[Calendar Year]
            ),
            "@DT",
            [Doanh Thu]
        )
EVALUATE
    SUMMARIZE(A,[Category],[@DT])
```

Category	@DT
Audio	11309946.1215001
TV and Video	11309946.1215001
Computers	11309946.1215001
Cameras and camcorders	11309946.1215001
Cell phones	11309946.1215001
Music, Movies and Audio Books	11309946.1215001
Games and Toys	11309946.1215001
Home Appliances	11309946.1215001
Audio	9764545.63850009

1. Các hàm để nhóm, lấy và lọc dữ liệu – CROSSJOIN

- Theo DAX Guide: Hàm CROSSJOIN là hàm trả về một bảng mà bảng đó được CROSS với một bảng chỉ định
- Nói một cảnh ngắn gọn nếu chúng ta có một bảng có m dòng, m’ cột và một bảng có n dòng, n’ cột sau khi sử dụng hàm crossjoin sẽ tạo ra một bảng gồm m x n dòng và n’ cột
- Tương tự nếu chúng ta có nhiều bảng thì kết quả trả về:
- Số dòng = Số dòng tất cả các bảng nhân lại với nhau
- Số cột = Số cột tất cả các bảng cộng lại với nhau
- Hàm Crossjoin cũng được sử dụng để tái kích hoạt ngữ cảnh

Syntax

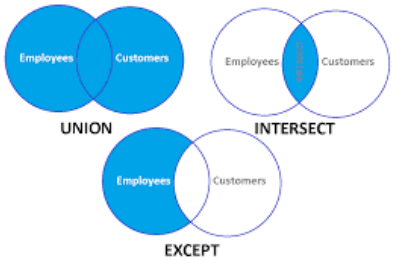
```
CROSSJOIN ( <Table> [, <Table> [, … ] ] )
```

PARAMETER	ATTRIBUTES	DESCRIPTION
Table	Repeatable	A table that will participate in the crossjoin.

```
EVALUATE
    CROSSJOIN(
        VALUES('Product'[Category]),
        VALUES('Date'[Calendar Year])
    )
```

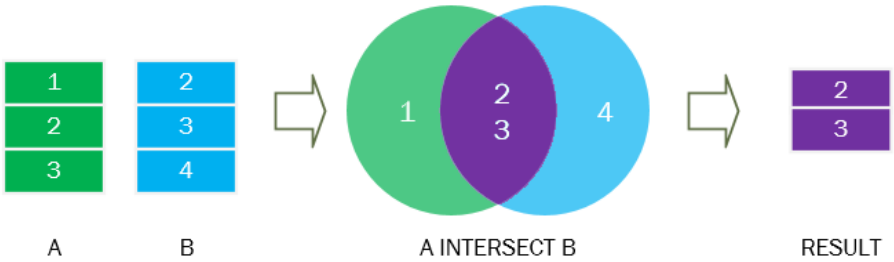
Category	Calendar Year
Cell phones	CY 2007
Music, Movies and Audio Books	CY 2007
Games and Toys	CY 2007
Home Appliances	CY 2007
Audio	CY 2008
TV and Video	CY 2008
Computers	CY 2008
Cameras and camcorders	CY 2008
Cell phones	CY 2008
Music, Movies and Audio Books	CY 2008

CÁCH HOẠT ĐỘNG CỦA CÁC HÀM INTERSECT, UNION, EXCEPT



- Các hàm này thường sẽ ưu tiên các tham số bên trái
- Hàm INTERSECT như chúng ta thấy thì không có sự ưu tiên bởi vì nó đang tìm điểm chung giữa hai vùng dữ liệu. Tuy nhiên khi áp dụng thực tế thì INTERSECT có thể được dung để tái kích hoạt FILTER CONTEXT
- Hàm EXCEPT thì sẽ ưu tiên tìm những điểm khác biệt của vùng dữ liệu bên trái so với bên phải. EXCEPT cũng được dung để tái kích hoạt FILTER CONTEXT
- Nếu tên cột giữa hai bên có sự khác biệt thì sẽ ưu tiên lấy tên của cột của tham số bên trái

1. Các hàm tìm điểm giao điểm chung và nối dữ liệu – INTERSECT



- INTERSECT: Là hàm tìm điểm chung giữa hai tập dữ liệu

Syntax

```
INTERSECT ( <LeftTable>, <RightTable> )
```

PARAMETER	ATTRIBUTES	DESCRIPTION
LeftTable		The Left-side table expression to be used for Intersect.
RightTable		The Right-side table expression to be used for Intersect.

```
DEFINE
// Tìm những sản phẩm đã bán trong năm 2007 và năm 2008
    VAR List_P2007 =
        SELECTCOLUMNS(
            CALCULATETABLE(
                VALUES(Sales[ProductKey]),
                'Date'[Calendar Year Number] = 2007

            ),
            "@P1",
            [ProductKey]
        )

    VAR List_P2008 =
        SELECTCOLUMNS(
            CALCULATETABLE(
                VALUES(Sales[ProductKey]),
                'Date'[Calendar Year Numbe
            ),
            "@P2",
            [ProductKey]
        )
EVALUATE
    INTERSECT(List_P2007,List_P2008)
```

@P1
1715
180
1846
2042
52
1334
1664
526
591

864 rows

2. Các hàm tìm điểm giao điểm chung và nối dữ liệu – INTERSECT

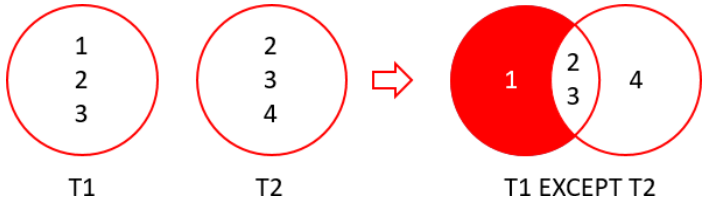
TÁI KÍCH HOẠT NGỮ CẢNH VỚI INTERSECT

```
DEFINE
// Tìm doanh thu của những sản phẩm đã bán trong năm 2007 và năm 2008
    VAR List_P2007 =
        CALCULATETABLE(
            VALUES(Sales[ProductKey]),
            'Date'[Calendar Year Number] = 2007
        )


    VAR List_P2008 =
        CALCULATETABLE(
            VALUES(Sales[ProductKey]),
            'Date'[Calendar Year Number] = 2008
        )
EVALUATE
{
    CALCULATE(
        [Doanh Thu],
        INTERSECT(
            VALUES('Product'[ProductKey]),
            INTERSECT(List_P2007,List_P2008)
        )
    )
}
```

Value
15517416.2568999

2. Các hàm tìm điểm giao điểm chung và nối dữ liệu – EXCEPT



EXCEPT: Là hàm tìm khác biệt tập dữ liệu 1 và tập dữ liệu 2

```
DEFINE
// Tìm những sản phẩm bán trong năm 2008 nhưng không được bán trong nă
    VAR List_P2007 =
        CALCULATETABLE(
            VALUES(Sales[ProductKey]),
            'Date'[Calendar Year Number] = 2007
        )


    VAR List_P2008 =
        CALCULATETABLE(
            VALUES(Sales[ProductKey]),
            'Date'[Calendar Year Number] = 2008
        )
EVALUATE
    EXCEPT(List_P2008,List_P2007)
```

Syntax

```
EXCEPT ( <LeftTable>, <RightTable> )
```

PARAMETER	ATTRIBUTES	DESCRIPTION
LeftTable		The Left-side table expression to be used for Except.
RightTable		The Right-side table expression to be used for Except.

ProductKey
1683
1245
1246
2096
2323
2062
2145
224
1021
1691

614 rows

2. Các hàm tìm điểm giao điểm chung và nối dữ liệu – EXCEPT

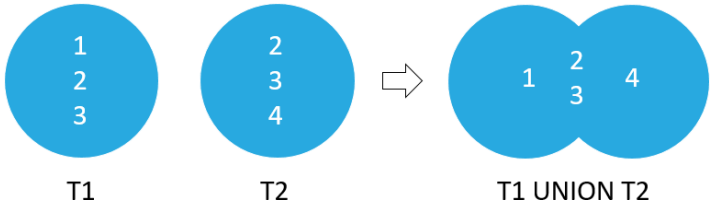
TÁI KÍCH HOẠT NGỮ CẢNH VỚI EXCEPT

```
DEFINE
// Tính doanh thu những sản phẩm bán trong năm 2008 nhưng không được bán trong năm 2007
    VAR List_P2007 =
        CALCULATETABLE(
            VALUES(Sales[ProductKey]),
            'Date'[Calendar Year Number] = 2007
        )


    VAR List_P2008 =
        CALCULATETABLE(
            VALUES(Sales[ProductKey]),
            'Date'[Calendar Year Number] = 2008
        )
EVALUATE
{
    CALCULATE(
        [Doanh Thu],
        INTERSECT(
            VALUES('Product'[ProductKey]),
            EXCEPT((List_P2007,List_P2008)
        )
    )
}
```

Value
5620619.75139998

2. Các hàm tìm điểm giao điểm chung và nối dữ liệu – UNION



- UNION: Là hàm nối hai tệp dữ liệu với nhau với tên COLUMN được lấy ưu tiên từ column bên trái

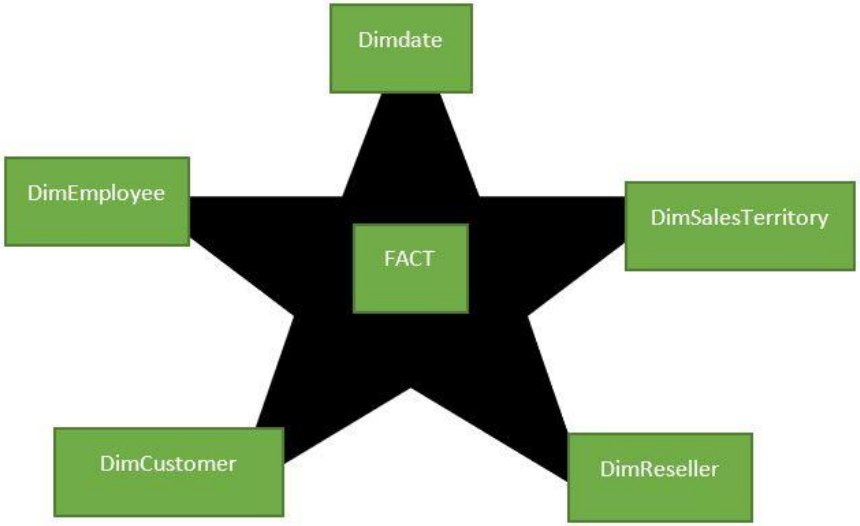
```
DEFINE
// Nối các sản phẩm được bán trong năm 2007 và 2008 lại với nhau
// Lưu ý: Dữ liệu trả về không có uniqe và bị duplicate
    VAR List_P2007 =
        CALCULATETABLE(
            VALUES(Sales[ProductKey]),
            'Date'[Calendar Year Number] = 2007
        )

    VAR List_P2008 =
        CALCULATETABLE(
            VALUES(Sales[ProductKey]),
            'Date'[Calendar Year Number] = 2008
        )
EVALUATE
    UNION(List_P2007,List_P2008)
```

ProductKey
1715
180
1846
2042
52
1334
1664
246
526
1117

2736 rows

CHAPTER 5: XÂY DỰNG MỐI QUAN HỆ ẢO VÀ GIAO THOA NGỮ CẢNH TRONG DAX



DAX GUIDE

Nguồn tham khảo: DAX Guide

DAX.do

CHAPTER 5: XÂY DỰNG MỐI QUAN HỆ ẢO VÀ GIAO THOA NGỮ CẢNH TRONG DAX

Chúng ta có khá nhiều phương pháp để xây dựng mối quan hệ ảo, nhưng ở đây mình chỉ giới thiệu một phương pháp được sử dụng phổ biến đó là sử dụng hàm TREATAS và KEEPFILTERS. Có một cách khác đó là userelationship nhưng bạn phải kéo relationship giữa hai bảng và unactive nó hoặc sử dụng cách cổ điển filter từng dòng (cách này mình đã loại bỏ không sử dụng)

DAX GUIDE

Nguồn tham khảo: DAX Guide

DAX.do

LƯU Ý RẰNG KHI KHAI BÁO BIẾN TRONG DAX THÌ KẾT QUẢ ĐÃ ĐƯỢC TRẢ VỀ. DO ĐÓ, CÁC EXPRESSION TRONG BIẾN CHỈ BỊ TÁC ĐỘNG BỞI CÁC NGỮ CẢNH HIỆN HÀNH VÀ CHỈ TÁC ĐỘNG LẠI LÊN CÁC EXPRESSION KHÁC NHƯNG LẠI KHÔNG CHỊU FILTER TỪ MỘT NGỮ CẢNH BÊN NGOÀI HOẶC NGỮ CẢNH XÁC ĐỊNH NÀO TRONG MỘT EXPRESSION NẰM Ở BIẾN KHÁC

1. Hàm TREATAS

- Hàm TREATAS là hàm dùng để tạo mqh ảo từ một hoặc nhiều column trong bảng ảo đến một column trong bảng thực, nó còn có thể xác định bảng thực chỉ Filter với các giá trị được đặt vào khao báo theo kiểu List ({2007,2008})
- Khi khai báo kiểu List thì nó sẽ trả về một bảng và có tên cột là Value (mặc định trong DAX)
- Hàm TREATAS có thể sử dụng với các column trong bảng ảo và kích hoạt mối quan hệ theo thứ tự của Column đến bảng thực
- Ví dụ lấy doanh thu lần lượt của năm 2007, 2008 theo trình độ giáo dục

```
TREATAS ( <Expression>, <ColumnName> [, <ColumnName> [, … ] ] )
```

```
DEFINE
    VAR A =
        ADDCOLUMNS (
            VALUES ( Customer[Education] ),
            "@DT_2007", CALCULATE ( [# Doanh Thu], TREATAS ( { 2007 }, 'Date'[Calendar Year Number] ) ),
            "@DT_2008", CALCULATE ( [# Doanh Thu], TREATAS ( { 2008 }, 'Date'[Calendar Year Number] ) )
        )

EVALUATE A
```

Education	@DT_2007	@DT_2008
	5818586.4299	8495062.5389
Bachelors	1546469.40899999	429554.129300002
Partial College	1443251.3321	317811.401100001
High School	1079802.71709999	272191.485600001
Partial High School	461560.951800001	112254.7439
Graduate Degree	960275.281599996	300708.6897

1. Hàm TREATAS

Ví dụ: Xem các sản phẩm được bán trong năm đầu tiên có doanh thu bao nhiêu qua các năm tiếp theo

Calendar Year	@DT	@DT_MinYear_Product
CY 2005		
CY 2006		
CY 2007	11309946.1214998	11309946.1214998
CY 2008	9927582.98849986	5330840.1861998
CY 2009	9353814.86739985	4497249.70060006
CY 2010		
CY 2011		

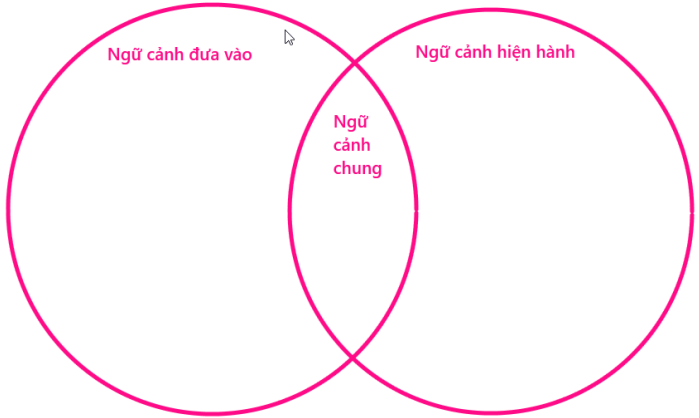
```
DEFINE
    VAR MinYear_Product =
        CALCULATETABLE (
            VALUES ( Sales[ProductKey] ),
            INDEX (
                1,
                CALCULATETABLE (
                    SUMMARIZE ( Sales, 'Date'[Calendar Year Number] ),
                    ALL ( 'Date' )
                ),
                ORDERBY ( [Calendar Year Number], ASC ),
                KEEP
            ),
            ALL ( 'Date' )
        )

EVALUATE
ADDCOLUMNS (
    VALUES ( 'Date'[Calendar Year] ),
    "@DT", [# Doanh Thu],
    "@DT_MinYear_Product", CALCULATE ( [# Doanh Thu], TREATAS ( MinYear_Product, Sales[ProductKey] ) )
)
```

2. Hàm KEEPFILTERS

```
KEEPFILTERS ( <Expression> )
```

- Là hàm sửa đổi các bộ lọc được áp dụng trong hàm CALCULATE và CALCULATETABLE
- Hàm KEEPFILTERS thực hiện việc giao thoa giữa ngữ cảnh đưa vào đối số trong hàm và ngữ cảnh hiện hành. Có thể hiểu như là INTERSECT hai ngữ cảnh với nhau
- Hàm KEEPFILTERS thường được sử dụng làm một đối số trong các hàm CALCULATE, CALCULATETABLE
- HàM KEEPFILTER thường hay sử dụng kết hợp với hàm TREATAS



When to use KEEPFILTERS over iterators - SQLBI

2. Hàm KEEPFILTERS

Ví dụ: Tạo một MEASURE và đưa {“Red”,”Blue} vào KEEPFILTERS sau đó nhấn 3 Filter trong Slicer gồm Azure, Red, Grey => Trả về DT của Color Red

```
DT KEEPFILTERS =
CALCULATE(
    [# Doanh Thu],
    KEEPFILTERS('Product'[Color] in {"Red","Blue"})
)
```

```
DT Red =
CALCULATE(
    [# Doanh Thu],
    'Product'[Color] = "Red"
)
```

Calendar Year	DT KEEPFILTERS	DT Red
CY 2007	366,027.15	366,027.15
CY 2008	395,277.22	395,277.22
CY 2009	348,797.73	348,797.73
Total	**1,110,102.10**	**1,110,102.10**

2. Hàm KEEPFILTERS

Ví dụ: Xem các sản phẩm được bán trong năm đầu tiên có doanh thu bao nhiêu qua các năm tiếp theo. Vào bảng Product kéo cột Produckey ra tạo môt

```
Using KEEPFILTERS =
VAR MinYear_Product =
    CALCULATETABLE(
        VALUES(Sales[ProductKey]),
        INDEX(
            1,
            CALCULATETABLE(
                SUMMARIZE(Sales,'Date'[Calendar Year Number]),
                ALL('Date')
            ),
            ORDERBY([Calendar Year Number]),
            KEEP
        ),
        ALL('Date')
    )
Return
    CALCULATE(
        [# Doanh Thu],
        KEEPFILTERS(TREATAS(MinYear_Product,Product[ProductKey]))
    )
```

```
Not Using KEEPFILTERS =
VAR MinYear_Product =
    CALCULATETABLE(
        VALUES(Sales[ProductKey]),
        INDEX(
            1,
            CALCULATETABLE(
                SUMMARIZE(Sales,'Date'[Calendar Year Number]),
                ALL()
            ),
            ORDERBY([Calendar Year Number]),
            KEEP
        ),
        ALL()
    )
Return
    CALCULATE(
        [# Doanh Thu],
        TREATAS(MinYear_Product,'Product'[ProductKey])
```

2. Hàm KEEPFILTERS

Calendar Year	Using KEEPFILTERS	Not Using KEEPFILTERS
CY 2007		11,309,946
CY 2008	312	5,330,840
CY 2009	246	4,497,250
Total	**558**	**21,138,036**

Việc sử dụng hàm KEEPFILTERS khi nhấn Slicer thay đổi kết quả bởi vì hàm này được sử dụng với mục đích giao thoa bộ lọc bên trong và bên ngoài

LƯU Ý VỀ FILTER

ĐỐI VỚI CỘT MÀ HÀM TREATAS TẠO MỐI QUAN HỆ ẢO ĐẾN. KHI TẠO FILTER CÁC GIÁ TRỊ TRONG CỘT ĐÓ KHÔNG LÀM THAY ĐỔI GIÁ TRỊ CỦA MEASURE TRỪ TRƯỜNG HỢP SỬ DỤNG HÀM KEEPFILTERS ĐI KÈM

HÀM TREATAS VÀ KEEPFILTERS ĐỀU BỊ ẢNH HƯỞNG BỞI CÁC DEPENDENT FILTER

Vậy làm sao có thể tránh cách trường hợp bị ảnh hưởng bởi Dependent Filter?

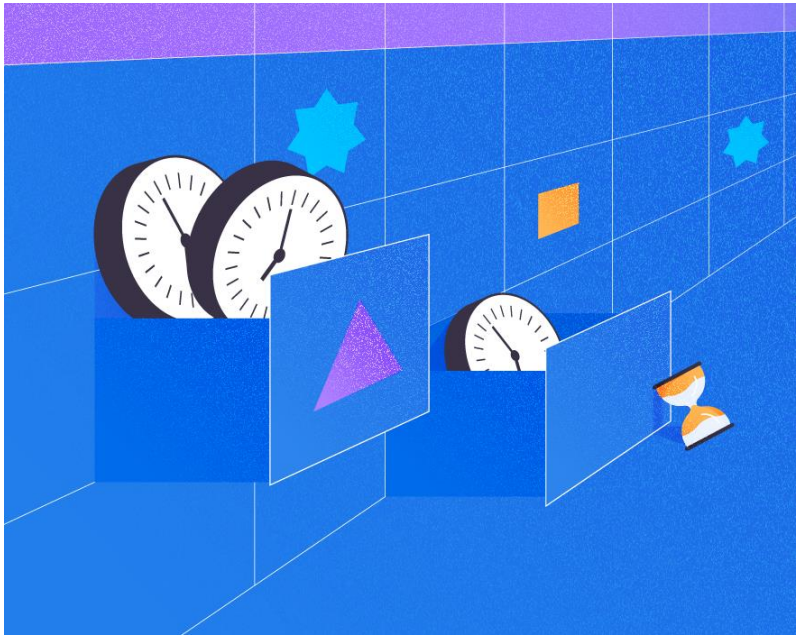
KẾT HỢP BIỂU THỨC TRẢ VỀ VỚI CÁC HÀM BỎ BỘ LỌC

Calendar Year	Not Using KEEPFILTERS
CY 2007	11,309,946
CY 2008	5,330,840
CY 2009	4,497,250
Total	**21,138,036**

Category

- ☐ Audio
- ☐ Cameras and camcorders
- ☐ Cell phones
- ☐ Computers
- ■ Games and Toys
- ☐ Home Appliances
- ☐ Music, Movies and Audio Books
- ☐ TV and Video

CHAPTER 6: CÁC HÀM TIME INTELLIGENCE TRONG DAX



DAX GUIDE

Nguồn tham khảo: DAX Guide

DAX.do

1. Các hàm tính lũy kế

- Hàm DATESYTD là hàm tính lũy kế theo ngày trong 1 năm. Đặc biệt đối với hàm này chúng ta có thể chỉ ngày kết thúc trong năm để tính toán
- Hàm DATESQTD là hàm tính lũy kê theo ngày trong 1 quý của 1 năm
- Hàm DATESMTD là hàm tính lũy kế theo ngày trong 1 tháng của 1 quý và của 1 năm (tính lũy kế theo tháng)
- Lưu ý các hàm này được hiểu như là một Filter (khi sử dụng phải kết hợp lại với Measure thông qua hàm CALCULATE
- Các hàm này phải được đặt ở đúng context. Ví dụ như YTD phải đặt đúng context là theo từng năm, nếu ngữ cảnh là Total thì là doanh thu lũy kế của năm gần nhất, QTD phải đặt đúng Context là quý còn nếu đặt ở ngữ cảnh năm thì là quý gần nhất của năm đó tương tự với MTD (áp dụng cho cả quý và năm).
- Nếu ngữ cảnh là ngày thì sẽ là lũy kế theo ngày của các ngày trong tháng, quý, năm
- Ví dụ chi tiết: Xem ở phần thực hành

```
DATESYTD ( <Dates> [, <YearEndDate>] )
```

```
DATESQTD ( <Dates> )
```

```
DATESMTD ( <Dates> )
```

1. Các hàm tính lũy kế

- Hàm TOTALYTD là hàm tính lũy kế theo ngày trong 1 năm. Đặc biệt đối với hàm này chúng ta có thể chỉ ngày kết thúc trong năm để tính toán
- Hàm TOTALQTD là hàm tính lũy kê theo ngày trong 1 quý của 1 năm
- Hàm TOTALMTD là hàm tính lũy kế theo ngày trong 1 tháng của 1 quý và của 1 năm (tính lũy kế theo tháng)
- Các hàm này có sự giống nhau với các hàm trước. Tuy nhiên những hàm này còn cho phép đặt Filter bên trong hàm.

```
TOTALYTD ( <Expression>, <Dates> [, <Filter>] [, <YearEndDate>]
)
```

```
TOTALQTD ( <Expression>, <Dates> [, <Filter>] )
```

```
TOTALMTD ( <Expression>, <Dates> [, <Filter>] )
```

2. Các hàm tính toán cộng trừ thời gian

- Hàm DATEADD: Giúp tính toán cộng trừ thời gian theo DATE/MONTH/YEAR để đưa ra kết quả. Tuy nhiên hàm này trả về kết quả là bảng chứ không phải là scalar value
- Hàm DATEDIFF: Giúp tính toán khoảng thời gian giữa 2 ngày cụ thể. Kết quả trả về có thể là số ngày, số tháng và số năm.

```
DATEADD ( <Dates>, <NumberOfIntervals>, <Interval> )
```

```
DATEDIFF ( <Date1>, <Date2>, <Interval> )
```

3. Các hàm tạo chuỗi thời gian

- Hàm DATEBETWEEN: Chỉ định một quãng thời gian được lọc trong cột DATE bảng DimDATE với quãng thời gian được xác định trong StartDate và EndDate
- Hàm DATESINPERIOD: Lấy ra một chuỗi trong cột DATE bảng DimDate với quãng thời gian được tùy ý chỉ định (Thường ứng dụng trong các bài toán rolling)

```
DATESBETWEEN ( <Dates>, <StartDate>, <EndDate> )
```

```
DATESINPERIOD ( <Dates>, <StartDate>, <NumberOfIntervals>,
<Interval> )
```

4. Các hàm lấy ra ngày đầu và ngày cuối theo ngữ cảnh

- Hàm FIRSTDATE: Lấy ra ngày đầu tiên theo context hiện hành. Ví dụ như context hiện tại là Year sẽ lấy ra ngày đầu tiên của năm đó hoặc ngày đầu tiên bán hàng, nếu là Product thì sẽ lấy ra ngày đầu tiên mà sản phẩm đó được bán ra
- Hàm LASTDATE: Lấy ra ngày cuối cùng theo context hiện hành. Ví dụ như context hiện tại là Year sẽ lấy ra ngày cuối cùng của năm đó hoặc ngày cuối cùng bán hàng, nếu là Product thì sẽ lấy ra ngày đầu tiên mà sản phẩm đó được bán ra

```
FIRSTDATE ( <Dates> )
```

```
LASTDATE ( <Dates> )
```

5. Các hàm so sánh cùng kì

- Hàm FIRSTDATE: Lấy ra ngày đầu tiên theo context hiện hành. Ví dụ như context hiện tại là Year sẽ lấy ra ngày đầu tiên của năm đó hoặc ngày đầu tiên bán hàng, nếu là Product thì sẽ lấy ra ngày đầu tiên mà sản phẩm đó được bán ra
- Hàm LASTDATE: Lấy ra ngày cuối cùng theo context hiện hành. Ví dụ như context hiện tại là Year sẽ lấy ra ngày cuối cùng của năm đó hoặc ngày cuối cùng bán hàng, nếu là Product thì sẽ lấy ra ngày đầu tiên mà sản phẩm đó được bán ra

```
FIRSTDATE ( <Dates> )
```

```
LASTDATE ( <Dates> )
```

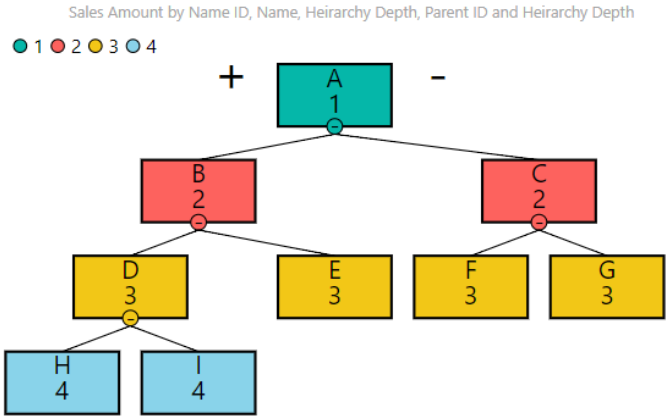
CHAPTER 7: CÁC KỸ THUẬT THƯỜNG SỬ DỤNG TRONG DAX

DAX GUIDE

Nguồn tham khảo: DAX Guide

DAX.do

CHAPTER 8: HÀM PATH VÀ CÁC HÀM LIÊN QUAN



DAX GUIDE

Nguồn tham khảo: DAX Guide

DAX.do

CHAPTER 9: CÁC HÀM MỚI TRONG DAX
OFFSET, INDEX, WINDOWN, PARTITIONBY, ORDERBY, RANK, ROWNUMBER, MATCHBY

Nguyễn Huy Hoàng
Admin Data Analyst Skills (SQL, Power BI, ...)- Thực Hành Qua Tình Huống

CÁC HÀM ORDERBY, PARTITIONBY, MATCHBY CHỈ ĐƯỢC SỬ DỤNG TRONG CÁC PARAMETER CỦA CÁC HÀM OFFSET, INDEX, WINDOW, RANK, ROWNUMBER

1. Hàm OFFSET

- Theo cách hiểu trong Excel, chúng ta có thể hiểu hàm offset di chuyển vị trí của ô lên trên, lên xuống dưới, qua phải, qua trái hay giữ nguyên vị trí
- Theo cách hiểu trong DAX dựa trên Context, chúng ta có thể hiểu hàm offset di chuyển vị trí của ngữ cảnh lên trước hoặc xuống hoặc giữ lại ngữ cảnh hiện tại
- Hàm OFFSET chỉ trả về 1 giá trị cũng giống như DISTINCT khi trả về 1 bảng gồm 1 cột và 1 dòng thì mặc định nó là **scalar value**
- Hàm OFFSET dùng được trong bảng ảo và bảng thực
- Với các giá trị Delta < 0: Di chuyển lùi
- Với các giá trị Delta > 0: Di chuyển tới
- PartitionBy: Bị lọc bởi giá của cột nào đó trên bảng thực hoặc bảng ảo
- Ví dụ : Tạo ra một table gồm một cột CurrentDT và một cột PreviousDT bị phân vùng theo Category

```
OFFSET ( <Delta> [, <Relation>] [, <OrderBy>] [, <Blanks>] [,
<PartitionBy>] )
```

```
DEFINE
VAR A =
    ADDCOLUMNS(
        SUMMARIZE(ALLSELECTED(Sales),'Product'[Category],'Date'[Calendar Year]),
        "@DT",
        [# Doanh Thu]
    )
VAR B =
    ADDCOLUMNS(
        A,
        "@PVDT",
        SELECTCOLUMNS(
            OFFSET(
                -1,
                A,
                ORDERBY([Calendar Year]),
                KEEP,PARTITIONBY([Category])
            ),
            [@DT]
        )
    )
EVALUATE
    WINDOW(0,REL,0,REL,B,ORDERBY([Calendar Year]),KEEP,PARTITIONBY([Category]))
```

Category	Calendar Year	@DT	@PVDT
Audio	CY 2007	102722.068099999	
Audio	CY 2008	105363.4204	102722.068099999
Audio	CY 2009	176432.6718	105363.4204
TV and Video	CY 2007	2269589.87580003	
TV and Video	CY 2008	919946.502999997	2269589.87580003
TV and Video	CY 2009	1203231.91349998	919946.502999997
Computers	CY 2007	2660318.86670004	
Computers	CY 2008	2066341.75160001	2660318.86670004

1. Hàm OFFSET - Code

```
DEFINE
VAR A =
  ADDCOLUMNS(
    SUMMARIZE(ALLSELECTED(Sales),'Product'[Category],'Date'[Calendar Year]),
    "@DT",
    [# Doanh Thu]
  )
VAR B =
  ADDCOLUMNS(
    A,
    "@PVDT",
    SELECTCOLUMNS(
      OFFSET(
        -1,
        A,
        ORDERBY([Calendar Year]),
        KEEP,PARTITIONBY([Category])
      ),
      [@DT]
    )
  )
EVALUATE
  WINDOW(0,REL,0,REL,B,ORDERBY([Calendar Year]),KEEP,PARTITIONBY([Category]))
```

2. Hàm WINDOW

- Hàm Window là hàm trả về về một Table trong đó các row được xác định từ việc đặt các From, FromType, To, ToType
- Type ABS: Khóa cố định From với việc đặt các index (dòng có thể có index bằng 0 hoặc là 1)
- Type REL: Cho From, To chạy lên trên hoặc xuống so với dòng hiện tại với các Index đã đặt từ trước
- PartitionBy phân vùng theo
- Blank: luôn luôn mặc định là KEEP (các giá trị rỗng được giữ lại)
- Với Fromtype, ToType là REL và giá trị From, To là như nhau thì bản chất của hàm Window trong việc xác định ngữ cảnh có thể được hiểu như hàm OFFSET
- Lấy ví dụ ở hàm OFFSET

```
WINDOW ( <From> [, <FromType>], <To> [, <ToType>] [, <Relation>]
[, <OrderBy>] [, <Blanks>] [, <PartitionBy>] )
```

```
DEFINE
    VAR A = SUMMARIZE(ALLSELECTED(Sales),'Product'[Category],'Date'[Calendar Year],'Date'[Calendar Year Number])
EVALUATE
    ADDCOLUMNS(
        A,
        "@DT",
        [# Doanh Thu],
        "@PVDT",
        CALCULATE(
            [# Doanh Thu],
            WINDOW(
                -1,REL,
                -1,REL,
                A,
                ORDERBY([Calendar Year Number]),KEEP,
                PARTITIONBY([Category])
            )
        )
    )
ORDER BY [Category], [Calendar Year Number] ASC
```

Category	Calendar Year	Calendar Year Number	@DT	@PVDT
Audio	CY 2007	2007	102722.068099999	
Audio	CY 2008	2008	105363.4204	102722.068099999
Audio	CY 2009	2009	176432.6718	105363.4204
Cameras and camcorders	CY 2007	2007	3274847.26189997	
Cameras and camcorders	CY 2008	2008	2184189.54259998	3274847.26189997
Cameras and camcorders	CY 2009	2009	1733545.1483	2184189.54259998
Cell phones	CY 2007	2007	477451.743700012	
Cell phones	CY 2008	2008	462713.468600009	477451.743700012

2. Hàm WINDOW - Code

```
DEFINE
  VAR A = SUMMARIZE(ALLSELECTED(Sales),'Product'[Category],'Date'[Calendar Year],'Date'[Calendar Year Number])
EVALUATE
  ADDCOLUMNS(
    A,
    "@DT",
    [# Doanh Thu],
    "@PVDT",
    CALCULATE(
      [# Doanh Thu],
      WINDOW(
        -1,REL,
        -1,REL,
        A,
        ORDERBY([Calendar Year Number]),KEEP,
        PARTITIONBY([Category])
      )
    )
  )
ORDER BY [Category], [Calendar Year Number] ASC
```

3. Hàm Index

- Trả về một dòng tuyệt đối (tức là dòng đó) với cột đã xác định và phân vùng. Trong trường hợp không phân vùng được có thể dẫn đến kết quả trả về có nhiều dòng
- Position < 0: với khởi đầu là index = -1, đi từ dưới lên
- Position > 0 với khởi đầu là index = 1, đi từ trên xuống
- Hàm Index có thể dùng với bảng ảo và bảng thựcs
- Ví dụ: Lấy ra doanh thu của năm đầu tiên cùng mỗi Category

```
INDEX ( <Position> [, <Relation>] [, <OrderBy>] [, <Blanks>] [,
<PartitionBy>] )
```

```
DEFINE
    VAR A = SUMMARIZE(ALLSELECTED(Sales),'Product'[Category],'Date'[Calendar Year],'Date'[Calendar Year Number])
EVALUATE
    ADDCOLUMNS(
        A,
        "@DT",
        [# Doanh Thu],
        "@EndDT",
        CALCULATE(
            [# Doanh Thu],
            INDEX(
                1,
                A,
                ORDERBY([Calendar Year Number]),KEEP,
                PARTITIONBY([Category])
            )
        )
    )
ORDER BY [Category], [Calendar Year Number] ASC
```

Category	Calendar Year	Calendar Year Number	@DT	@EndDT
Home Appliances	CY 2008	2008	3962572.24179997	2347281.8009
Home Appliances	CY 2009	2009	3290602.9958	2347281.8009
Music, Movies and Audio Books	CY 2007	2007	87874.4371999999	87874.4371999999
Music, Movies and Audio Books	CY 2008	2008	120717.8289	87874.4371999999
Music, Movies and Audio Books	CY 2009	2009	105614.471900001	87874.4371999999
TV and Video	CY 2007	2007	2269589.8758	2269589.8758
TV and Video	CY 2008	2008	919946.502999997	2269589.8758
TV and Video	CY 2009	2009	1203231.91349998	2269589.8758

3. Hàm INDEX - Code

```
DEFINE
  VAR A =
    SUMMARIZE (ALLSELECTED ( Sales ),'Product'[Category],'Date'[Calendar Year],'Date'[Calendar Year Number])
EVALUATE
  ADDCOLUMNS (
    A,
    "@DT",
    [# Doanh Thu],
    "@PVDT",
    CALCULATE (
      [# Doanh Thu],
      INDEX(
        1,
        A,
        ORDERBY([Calendar Year Number]),
        PARTITIONBY([Category])
      )
    )
  )
ORDER BY
  [Category],[Calendar Year Number] ASC
```

4. Hàm RANK

- Xét Rank dựa trên một cột chỉ định nào đó. Ví dụ như Doanh Thu, Chi phí….
- Ties: Xét RANK theo cách nào Dense/Skip (Tương tự như hàm RANKX)
- Relation: Bảng để xét RANK, ở đây có thể là bảng thực hoặc là bảng ảo
- Order By: DESC/ASC
- Blank: luôn luôn mặc định là KEEP (các giá trị rỗng được giữ lại)
- PartitionBy: Phân vùng theo
- Ví dụ: Xếp hạng theo Doanh Thu của các Category trong một năm

```
RANK ( [<Ties>] [, <Relation>] [, <OrderBy>] [, <Blanks>] [, <PartitionBy>] )
```

```
Table =
    VAR A = SUMMARIZE (ALLSELECTED ( Sales ),'Date'[Calendar Year Number],'Product'[Category])
    VAR B = ADDCOLUMNS(A,"@DT",INT([Doanh Thu]))
Return
    ADDCOLUMNS(
        B,
        "@Rank_Cate_In_Year",
        RANK(DENSE,B,ORDERBY([@DT],DESC),PARTITIONBY('Date'[Calendar Year Number]))
    )
```

Calendar Year Number	Category	@DT	@Rank_Cate_In_Year
2007	Cameras and camcorders	3274847	1
2007	Computers	2660318	2
2007	Home Appliances	2347281	3
2007	TV and Video	2269589	4
2007	Cell phones	477451	5
2007	Audio	102722	6
2007	Games and Toys	89860	7
2007	Music, Movies and Audio Books	87874	8

Calendar Year Number	Category	@DT	@Rank_Cate_In_Year
2008	Home Appliances	3962572	1
2008	Cameras and camcorders	2184189	2
2008	Computers	2066341	3
2008	TV and Video	919946	4
2008	Cell phones	462713	5
2008	Music, Movies and Audio Books	120717	6
2008	Games and Toys	105738	7
2008	Audio	105363	8

4. Hàm RANK - Code

```
Hàm Rank =
    VAR A = SUMMARIZE (ALLSELECTED ( Sales ),'Date'[Calendar Year Number],'Product'[Category])
    VAR B = ADDCOLUMNS(A,"@DT",INT([Doanh Thu]))
Return
    ADDCOLUMNS(
        B,
        "@Rank_Cate_In_Year",
        RANK(DENSE,B,ORDERBY([@DT],DESC),PARTITIONBY('Date'[Calendar Year Number]))
    )
```

4. Hàm ROWNUMBER

- Hàm ROW_NUMBER có tính chất khá giống Ties DENSE trong hàm RANK bởi vì nếu cùng một mức giá trị thì sẽ có STT khác nhau
- Relation: Bảng để xét RANK, ở đây có thể là bảng thực hoặc là bảng ảo
- Order By: DESC/ASC
- Blank: luôn luôn mặc định là KEEP (các giá trị rỗng được giữ lại)
- PartitionBy: Phân vùng theo
- Ví dụ: Lấy lại ví dụ ở phần

```
ROWNUMBER ( [<Relation>] [, <OrderBy>] [, <Blanks>] [,
<PartitionBy>] )
```

```
Hàm ROWNUMBER =
    VAR A = SUMMARIZE (ALLSELECTED ( Sales ),'Date'[Calendar Year Number],'Product'[Category])
    VAR B = ADDCOLUMNS(A,"@DT",INT([Doanh Thu]))
Return
    ADDCOLUMNS(
        B,
        "@Rank_Cate_In_Year",
        ROWNUMBER(B,ORDERBY([@DT],DESC),PARTITIONBY('Date'[Calendar Year Number]))
    )
```

Calendar Year Number	Category	@DT	@Rank_Cate_In_Year
2007	Cameras and camcorders	3274847	1
2007	Computers	2660318	2
2007	Home Appliances	2347281	3
2007	TV and Video	2269589	4
2007	Cell phones	477451	5
2007	Audio	102722	6
2007	Games and Toys	89860	7
2007	Music, Movies and Audio Books	87874	8

Calendar Year Number	Category	@DT	@Rank_Cate_In_Year
2008	Home Appliances	3962572	1
2008	Cameras and camcorders	2184189	2
2008	Computers	2066341	3
2008	TV and Video	919946	4
2008	Cell phones	462713	5
2008	Music, Movies and Audio Books	120717	6
2008	Games and Toys	105738	7
2008	Audio	105363	8

5. Hàm ROWNUMBER - Code

```
Hàm ROWNUMBER =
    VAR A = SUMMARIZE (ALLSELECTED ( Sales ),'Date'[Calendar Year Number],'Product'[Category])
    VAR B = ADDCOLUMNS(A,"@DT",INT([Doanh Thu]))
Return
    ADDCOLUMNS(
        B,
        "@Rank_Cate_In_Year",
        ROWNUMBER(B,ORDERBY([@DT],DESC),PARTITIONBY('Date'[Calendar Year Number]))
    )
```