a brand of Loader

www.ssasworkshop.com

ADVANCED DAX WORKSHOP

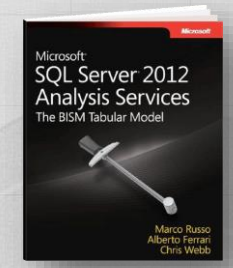
brought to you by sqlbi

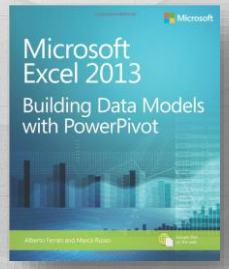
Alberto Ferrari
alberto.ferrari@sqlbi.com
twitter @FerrariAlberto

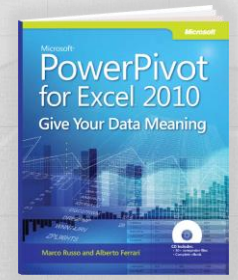
Many-to-many in DAX

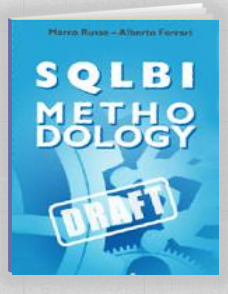
Who We Are

- BI Experts and Consultants
- Founders of www.sqlbi.com
  - Problem Solving
  - Complex Project Assistance
  - DataWarehouse Assesments and Development
  - Courses, Trainings and Workshops
- Book Writers
- Microsoft Gold Business Intelligence Partners
- SSAS Maestros – MVP – MCP

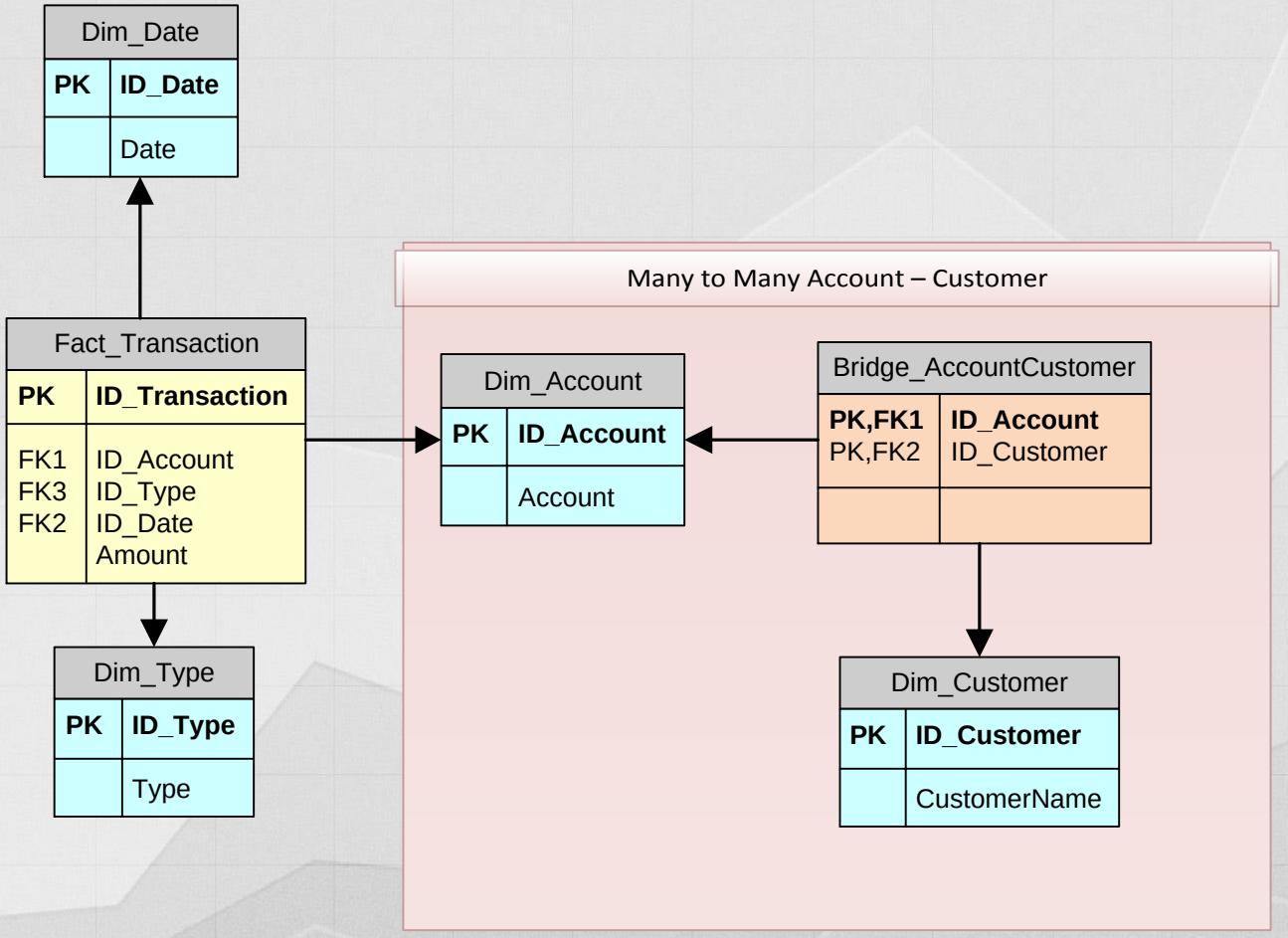








Many-to-many Relationships



No support for M2M in Tabular

- Multidimensional handles M2M
  - Directly in the engine
  - Performance may suffer
- Tabular does not handle M2M
  - Only plain vanilla 1:N relationships
  - Only on one column
  - Seems not to be enough…
- But… hey, we have DAX!

sqlbi

Demo – Classical M2M

- We will start looking at the final result
- Then, we dive into the DAX code

sqlbi

The M2M DAX Pattern

- Leverages
  - CALCULATE
  - Row Contexts
  - Filter Contexts
  - Automatic transformation of Row Context into Filter Context using CALCULATE
- Next slide: the formula
  - Keep it in mind
  - It will appear quite often from now on

sqlbi

The DAX Formula

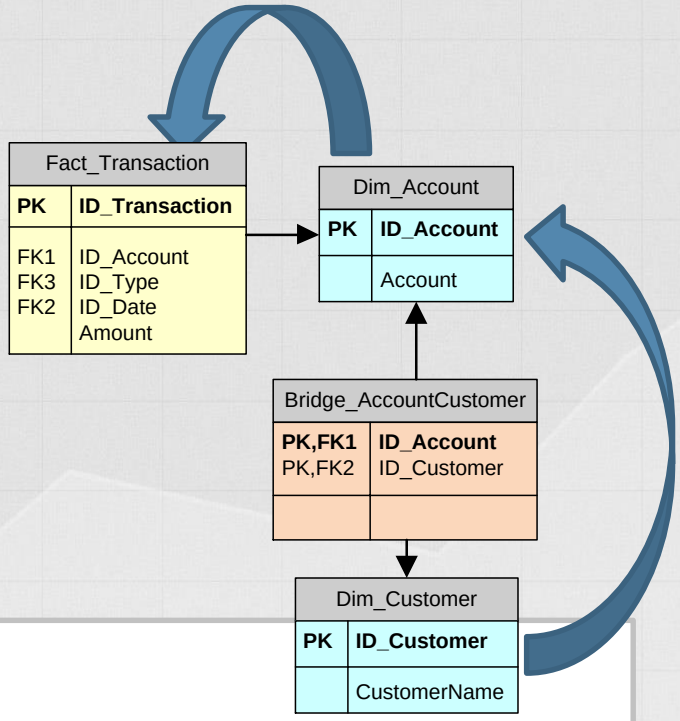
The pattern of many-to-many is always the same
(Later we will see an optimized version)

```
AmountM2M :=

CALCULATE(
    SUM( Fact_Transaction[Amount] ),
    FILTER(
        Dim_Account,
        CALCULATE(
            COUNTROWS( Bridge_AccountCustomer )
        ) > 0
    )
)
```

sqlbi

What the formula should perform



Filter on Dim_Account

Filter on Dim_Customer

```
AmountM2M :=

CALCULATE (
    SUM( Fact_Transaction[Amount] ),
    FILTER(
        Dim_Account,
        CALCULATE(
            COUNTROWS( Bridge_AccountCustomer )
        ) > 0
    )
)
```

sqlbi

The Karma of CALCULATE

Row Labels	AmountM2M_Wrong	AmountM2M_Correct
Luke	**5000**	**800**
Luke	800	800
Mark	**5000**	**2800**
Mark	800	800
Mark-Paul	1000	1000
Mark-Robert	1000	1000
Paul	**5000**	**1700**
Mark-Paul	1000	1000
Paul	700	700
Robert	**5000**	**1700**
Mark-Robert	1000	1000
Robert	700	700
Grand Total	**5000**	**5000**

```
AmountM2M_Correct := CALCULATE (
    SUM (Fact_Transaction[Amount]),
    FILTER (
        Dim_Account,
        CALCULATE (
            COUNTROWS (Bridge_AccountCustomer)
        ) > 0
    )
)
```

```
AmountM2M_Wrong := CALCULATE (
    SUM (Fact_Transaction[Amount]),
    FILTER (
        Dim_Account,
        COUNTROWS (Bridge_AccountCustomer) > 0
    )
)
```

Wrong formula in action

Customer
Mark
Paul
Robert
Luke

Customer	Account
Mark	Mark
Paul	Paul
Robert	Robert
Luke	Luke
Mark	Mark-Ro
Robert	Mar
Mark	aul
Paul	ark-Paul

Account
Mark
Paul
Robert
Luke
Mark-Robert
Mark-Paul

```
AmountM2M :=

CALCULATE (
    SUM (Fact_Transaction[Amount])
    FILTER (
        Dim_Account,
        COUNTROWS (Bridge_AccountCustomer) > 0
    )
)
```

All the rows in the Account table survived the FILTER

Many-to-many in action

Customer
Mark
Paul
Robert
Luke

Customer	Account
Mark	Mark
Paul	Paul
Robert	Robert
Luke	Luke
Mark	Mark-Robert
Robert	Mark-R
Mark	
Paul	rk-Paul

Account
Mark
Paul
Robert
Luke
Mark-Robert
Mark-Paul

```
AmountM2M :=

CALCULATE (
    SUM (Fact_Transaction[Am
    FILTER (
        Dim_Account,
        CALCULATE (
            COUNTROWS (Bridge_AccountCustomer)
        ) > 0
    )
)
```

Two rows in the Account table survived the FILTER

The Karma of CALCULATE

Row Labels	AmountM2M_Wrong	AmountM2M_Correct
Luke	**5000**	**800**
Luke	800	800
Mark	**5000**	**2800**
Mark	800	800
Mark-Paul	1000	1000
Mark-Robert	1000	1000
Paul	**5000**	**1700**
Mark-Paul	1000	1000
Paul	700	700
Robert	**5000**	**1700**
Mark-Robert	1000	1000
Robert	700	700
Grand Total	**5000**	**5000**

```
AmountM2M_Correct := CALCULATE (
    SUM (Fact_Transaction[Amount]),
    FILTER (
        Dim_Account,
        CALCULATE (
            COUNTROWS (Bridge_AccountCustomer)
        ) > 0
    )
)
```

```
AmountM2M_Wrong := CALCULATE (
    SUM (Fact_Transaction[Amount]),
    FILTER (
        Dim_Account,
        COUNTROWS (Bridge_AccountCustomer) > 0
    )
)
```

DAX Formula with SUMMARIZE

The formula performs much better using SUMMARIZE instead of COUNTROWS.

SUMMARIZE returns the ID_Account that are visible in the bridge with a single step, transforming them in DimAccount column references

```
AmountM2M :=

CALCULATE(
    SUM( Fact_Transaction[Amount] ),
    SUMMARIZE(
        Bridge_AccountCustomer,
        DimAccount[ID_Account]
    )
)
```

sqlbi

SUMMARIZE in action

Customer
Mark
Paul
Robert
Luke

Customer	Account
Mark	Mark
Paul	Paul
Robert	Robert
Luke	Luke
Mark	Mark-Robert
Robert	Mark-Robert
Mark	Mark-Paul
Paul	Mark-Paul

Account
Mark
Paul
Robert
Luke
Mark-Robert
Mark-Paul

```
AmountM2M :=

CALCULATE(
    SUM( Fact_Transaction[Amount] ),
    SUMMARIZE(
        Bridge_AccountCustomer,
        DimAccount[ID_Account]
    )
)
```

The context is moved in a single Vertipaq operation

Many-to-many with SUMMARIZE

- The formula is
  - Much faster, no iteration needed
  - Easier to code
  - Slightly harder to understand
- Using COUNTROWS can still be useful
  - If you want to find missing values
  - To use more complex relationships
- Both formulas are worth learning

sqlbi

Direct Table Filtering

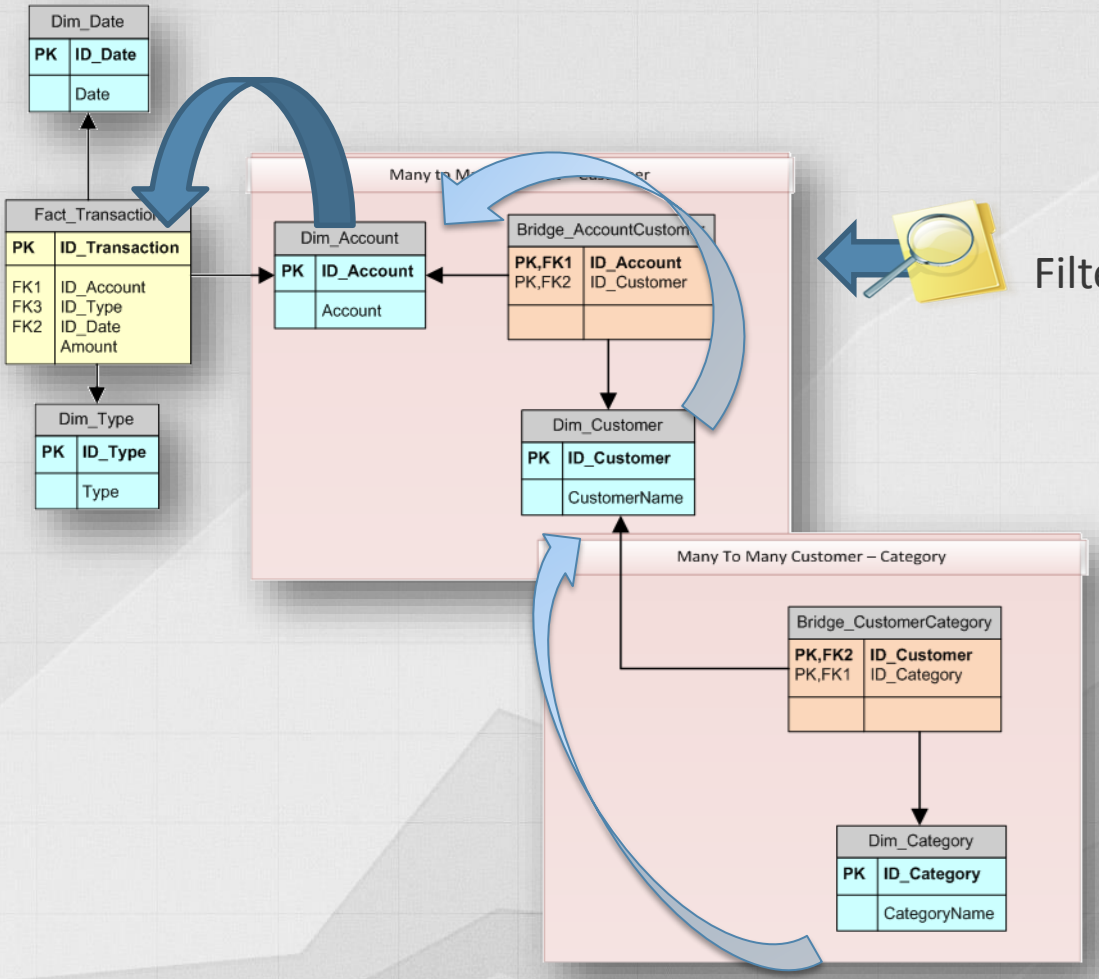
Behave and performs the same as SUMMARIZE
More compact
Somewhat harder to understand, unless you learned table filtering

```
AmountM2M :=

CALCULATE(
    SUM( Fact_Transaction[Amount] ),
    Bridge_AccountCustomer
)
```

sqlbi

Cascading Many To Many



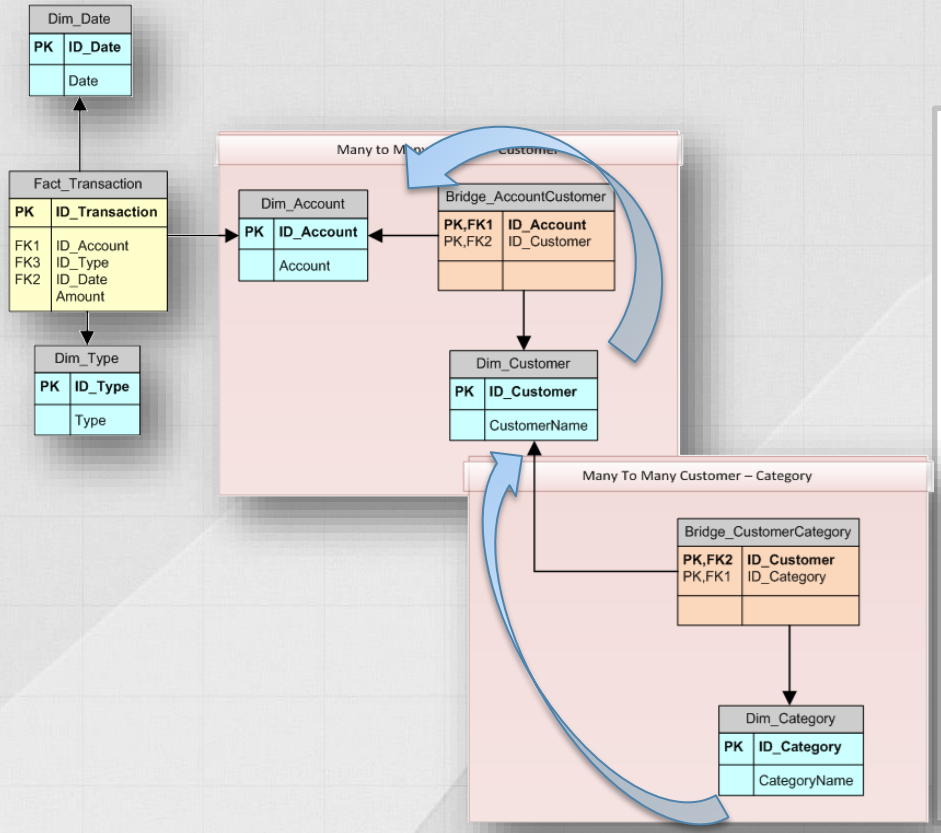
Filter on Dim_Account

The pattern is the same, but this time we need to jump two steps to complete our task

Filter on Dim_Category

sqlbi

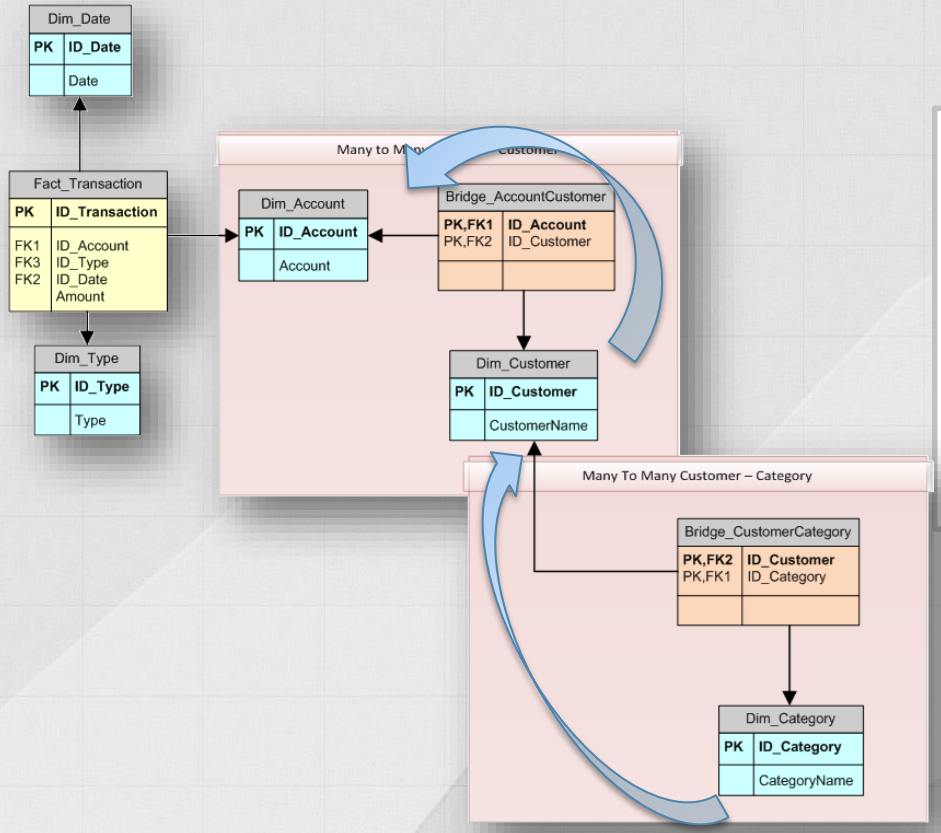
Cascading Many To Many




```
CALCULATE(
    SUM( Fact_Transaction[Amount] ),
    CALCULATETABLE (
        SUMMARIZE (
            BridgeAccountCustomer,
            DimAccount[ID_Account]
        ),
        SUMMARIZE (
            Bridge_CustomerCategory,
            DimCustomer[ID_Customer]
        )
    )
)
```

sqlbi

Cascading Many To Many




```
CALCULATE(
    SUM( Fact_Transaction[Amount] ),
    CALCULATETABLE (
        BridgeAccountCustomer,
        Bridge_CustomerCategory
    )
)
```

The formula is compact and fast, but not easy to read

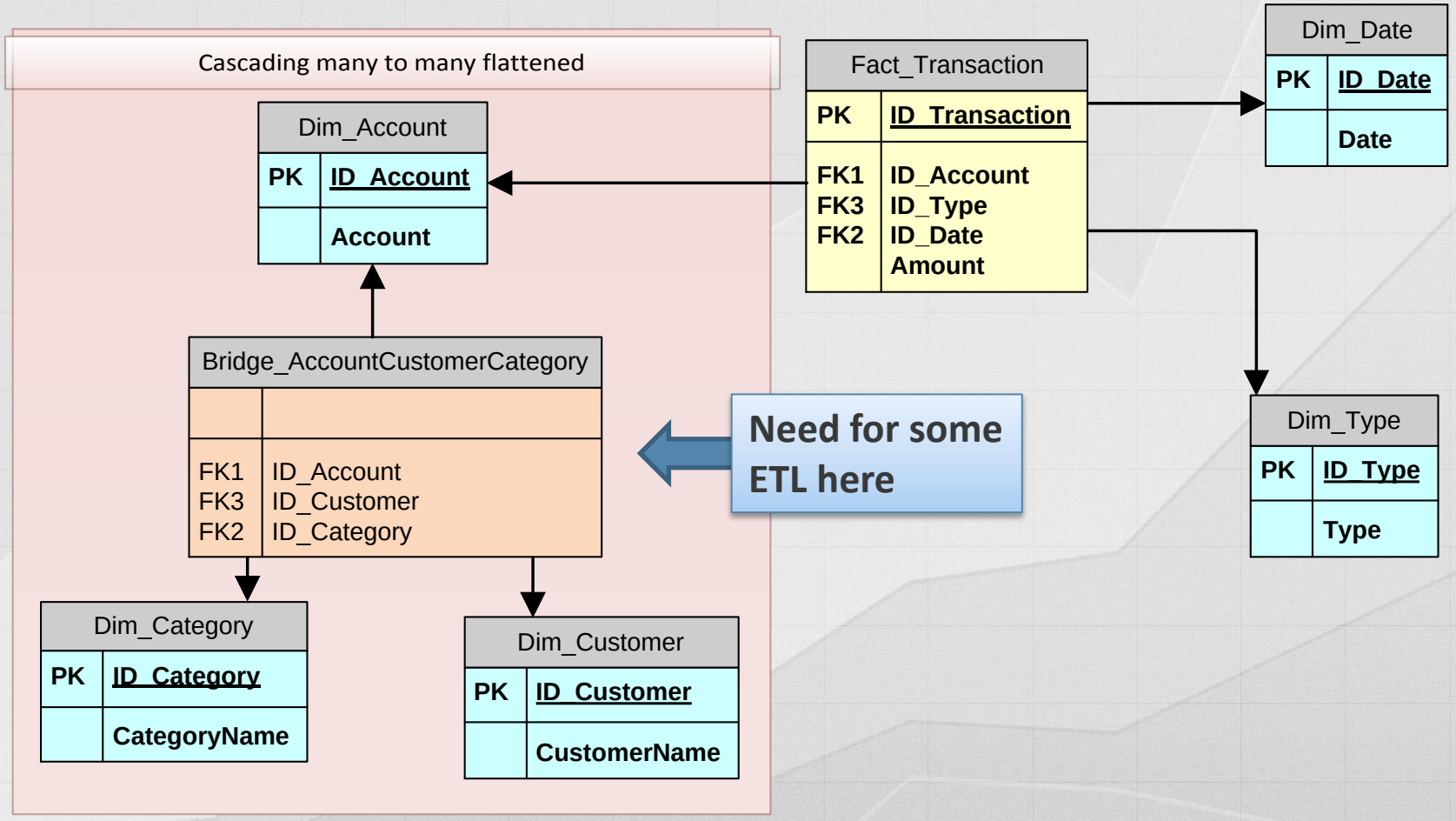
sqlbi

Cascading Many To Many

- Generic Formula
- Works with any number of steps
- Be careful
  - Start with the farthest table
  - Move one step a time towards the fact table
  - One SUMMARIZE for each step
- Complexity: M x N (geometric…)

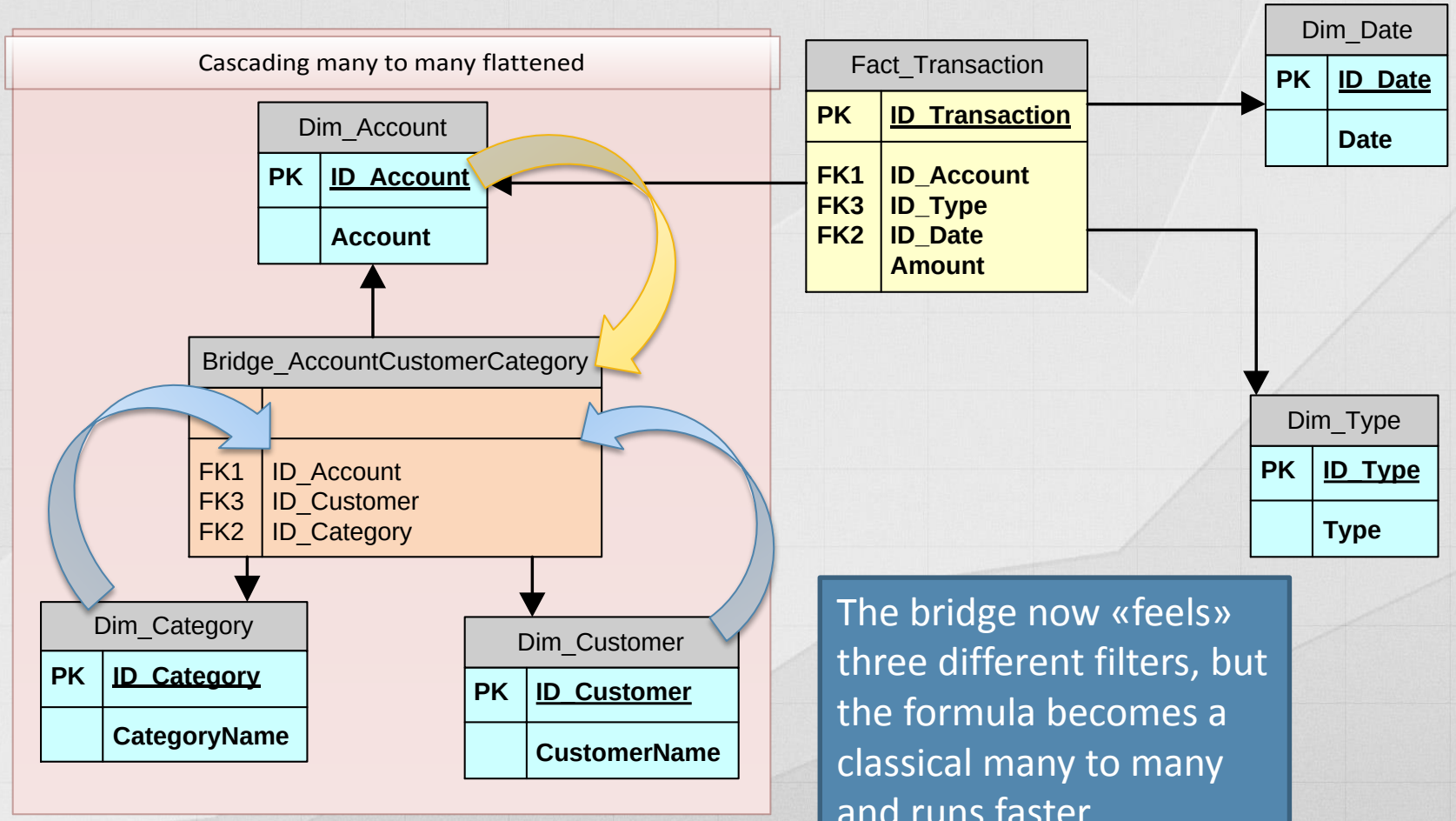
sqlbi

Cascading Alternative



sqlbi

Cascading Alternative



The bridge now «feels» three different filters, but the formula becomes a classical many to many and runs faster

sqlbi

Flattened Cascading Many To Many

- Flattened Data Model
- Faster than the cascading one
- Simpler formula
- Needs some ETL
  - A view is enough most of the time
- Worth a try in Multidimensional too…

sqlbi

Compute the new and the returning customers is not a complex topic using the DAX language

New and Returning Customers

sqlbi

New Customers

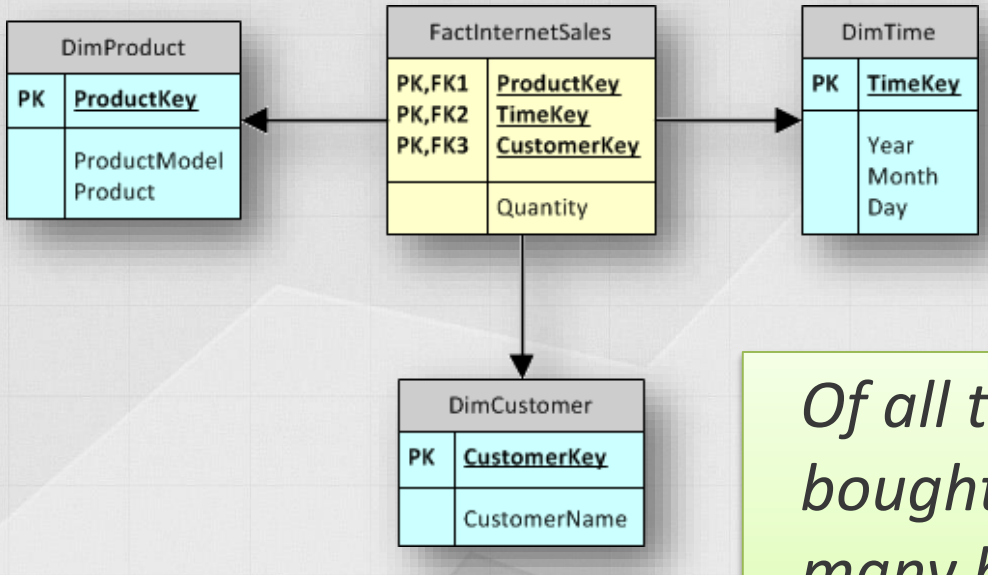
- Many useful calculations
  - Customers
  - Buying customers
  - New Customers
  - Returning customers
- Not very simple, but not «advanced»
- Once you learn the pattern
  - Sales of new customers
  - Sales of returning customers

sqlbi

Basket analysis uses the fact table as a bridge between two dimensions

Basket Analysis

Basket Analysis: The Scenario



Of all the customers who have bought a Mountain Bike, how many have never bought a mountain tire tube?

sqlbi

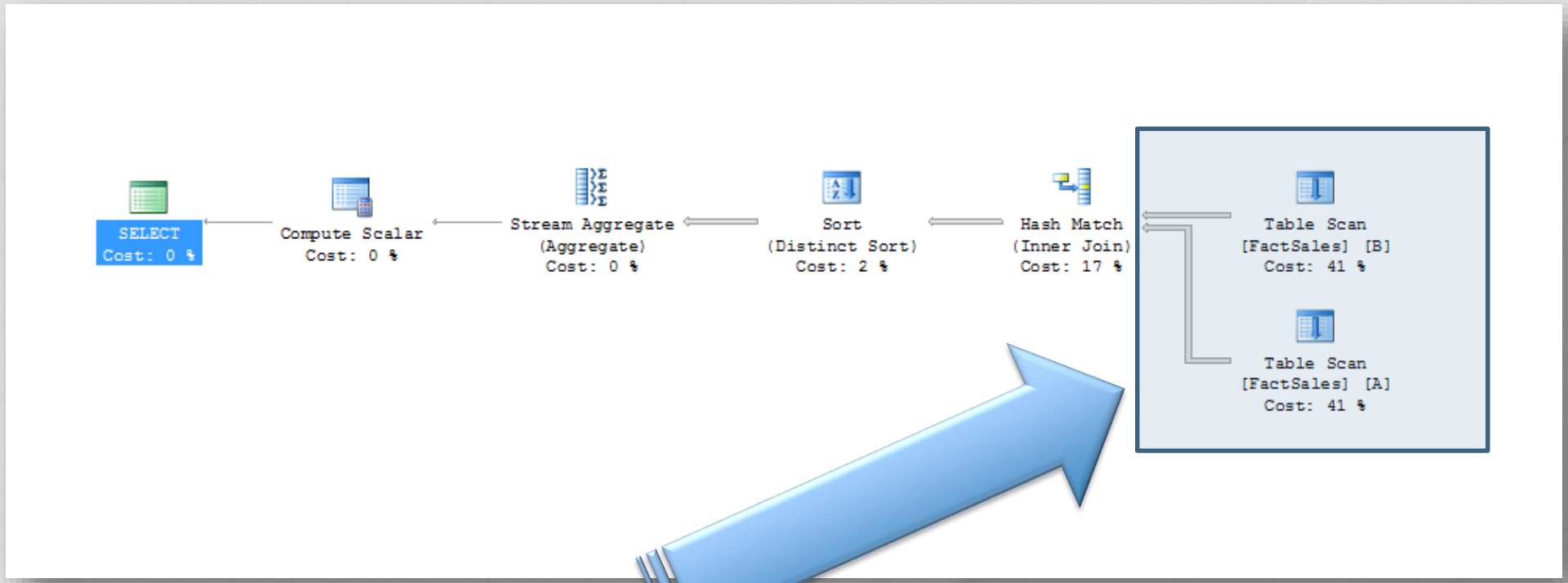
Basket Analysis in SQL

Two iterations over the fact table needed

```
SELECT
    COUNT (DISTINCT A.CustomerKey)
FROM
    Sales A
    INNER JOIN Sales B
        ON A.CustomerKey = B.CustomerKey
WHERE
    A.ProductModel = ‘MOUNTAIN TIRE TUBE' AND A.Year <= 2004
AND
    B.ProductModel = ‘MOUNTAIN-100' AND B.Year <= 2004
```

sqlbi

Look the query plan…

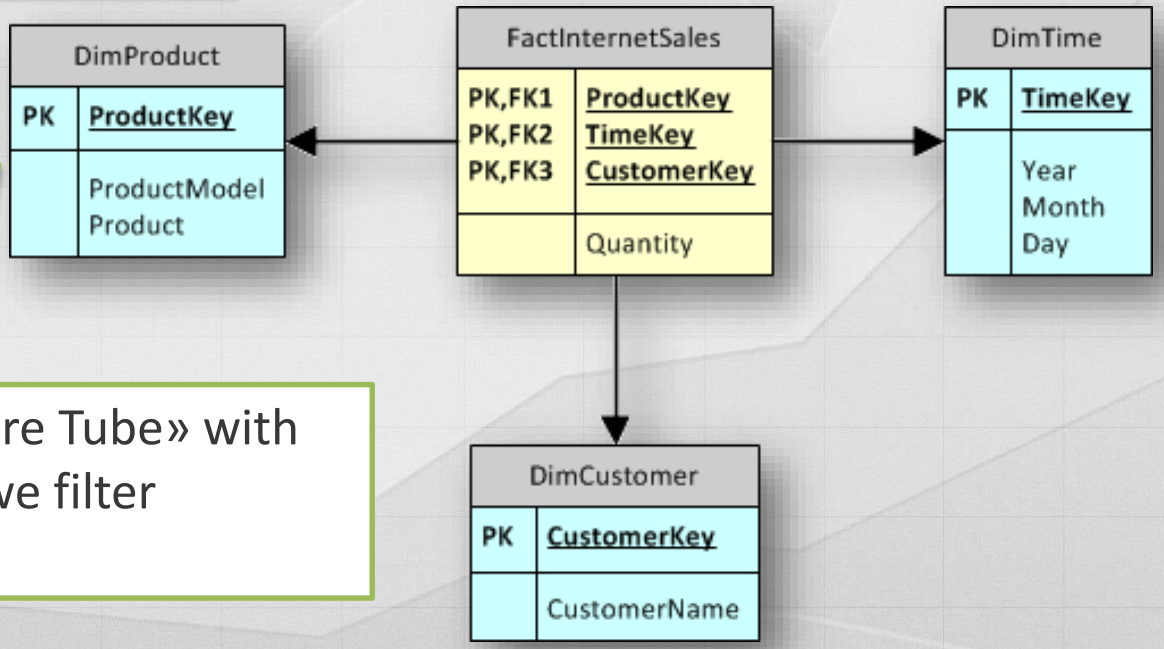


This is the fact table…
Do you really like to self-join it?

sqlbi

Basket Analysis: The Data Model

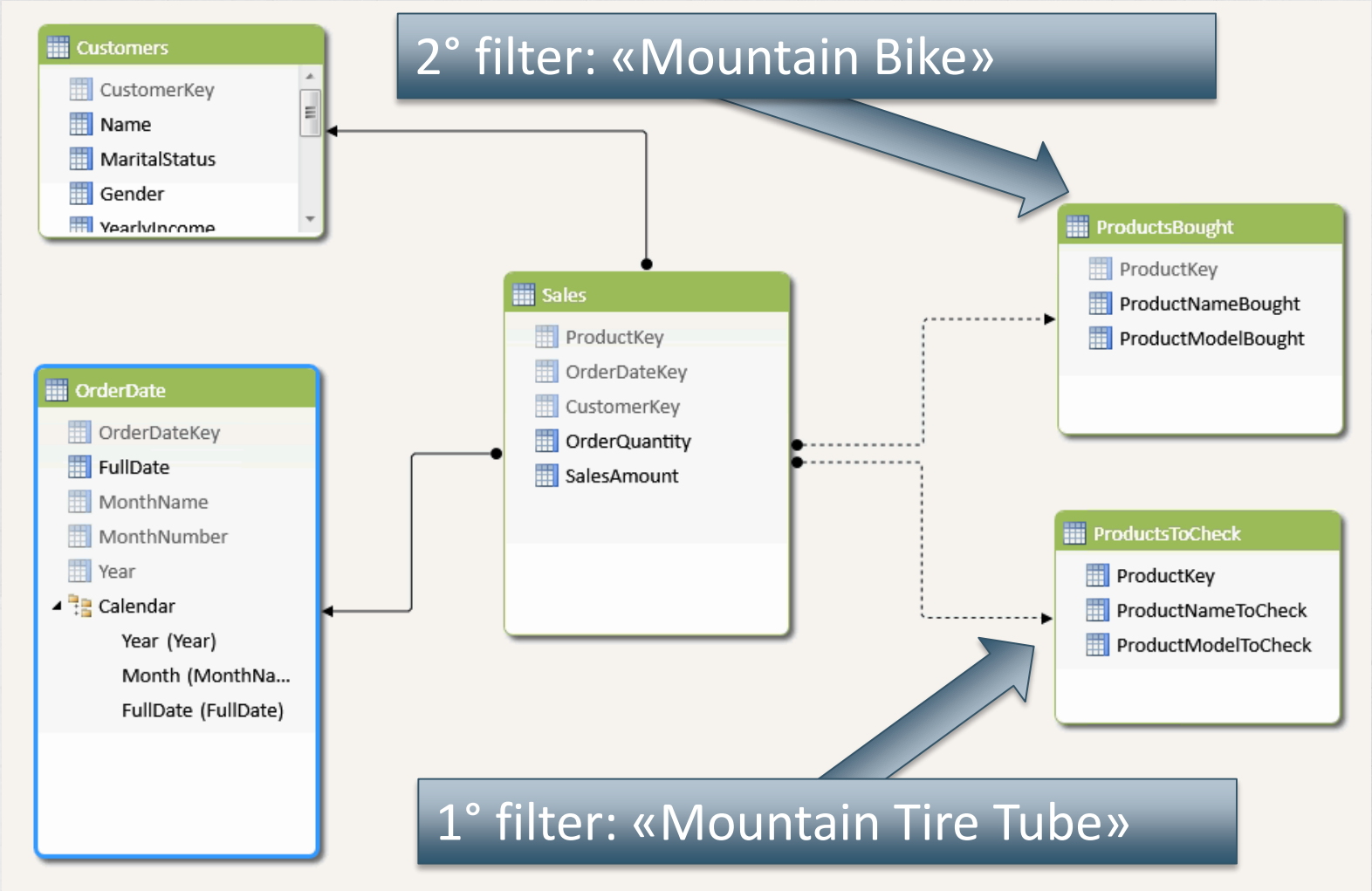
*Of all the customers who have bought a **Mountain Bike**, how many have never bought a **mountain tire tube**?*



We can filter «Mountain Tire Tube» with this table but… where do we filter «Mountain Bike»?

sqlbi

Basket Analysis: The Data Model



sqlbi

The Final Result

2° filter: «Mountain Tire Tube»

ProductModelToCheck

Men's Sports Shorts	Minipump	ML Bottom Bracket	ML Crankset	ML Fork	ML Headset
ML Mountain Frame	ML Mountain Frame-W	ML Mountain Front Wheel	ML Mountain Handlebars	ML Mountain Pedal	ML Mountain Rear Wheel
ML Mountain Seat/Saddl...	ML Mountain Tire	ML Road Frame	ML Road Frame-W	ML Road Front Wheel	ML Road Handlebars
ML Road Pedal	ML Road Rear Wheel	ML Road Seat/Saddle 2	ML Road Tire	ML Touring Seat/Saddle	Mountain Bike Socks

ProductModelBo...

- ML Road Rear ...
- ML Road Seat/...
- ML Road Tire
- ML Touring Se...
- Mountain Bike...
- Mountain Bott...
- Mountain Pump
- Mountain Tire ...
- Mountain-100
- Mountain-200
- Mountain-300
- Mountain-400...
- Mountain-500
- Patch kit
- Racing Socks
- Rear Brakes

Row Labels	Column Labels: Mountain Tire Tube HavingProduct	NotHavingProduct	Total HavingProduct	Total NotHavingProduct
2007	**33**	**174**	**33**	**174**
Mountain-400-W Silver, 38	9	48	9	48
Mountain-400-W Silver, 40	8	42	8	42
Mountain-400-W Silver, 42	7	41	7	41
Mountain-400-W Silver, 46	9	43	9	43
2008	**83**	**460**	**83**	**460**
Mountain-400-W Silver, 38	26	122	26	122
Mountain-400-W Silver, 40				
Mountain-400-W Silver, 42				
Mountain-400-W Silver, 46				
2009				
Mountain-400-W Silver, 38				
Mountain-400-W Silver, 40	20	108	20	108
Mountain-400-W Silver, 42	21	108	21	108
Mountain-400-W Silver, 46	16	122	16	122
2010	**83**	**460**	**83**	**460**
Mountain-400-W Silver, 38	26	122	26	122
Mountain-400-W Silver, 40	20	108	20	108
	21	108	21	108
	16	122	16	122
	83	**460**	**83**	**460**

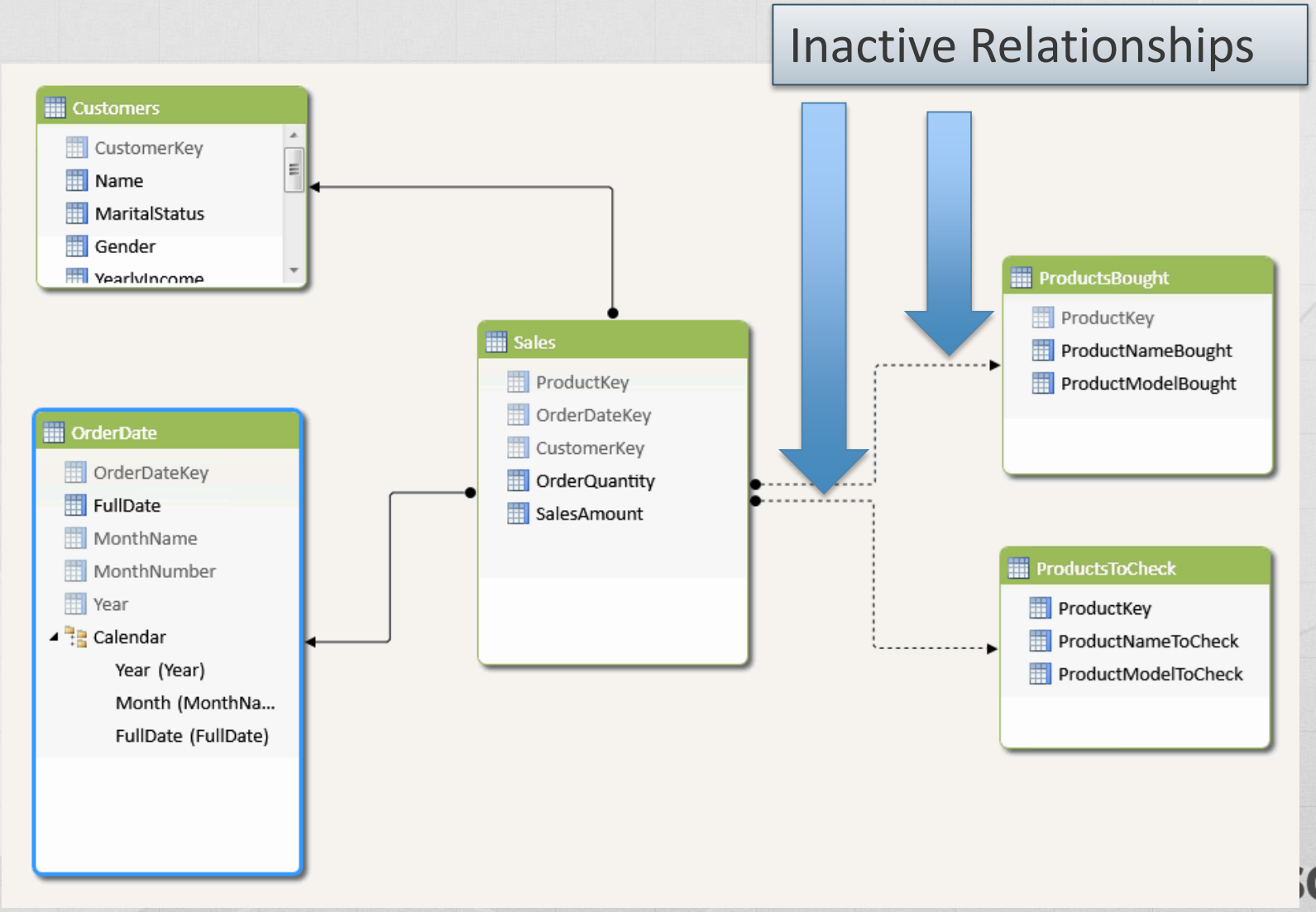
HavingProduct = Bought Both

NotHavingProduct = Bought Bike, No Tire

1° filter: «Mountain Bike»

sqlbi

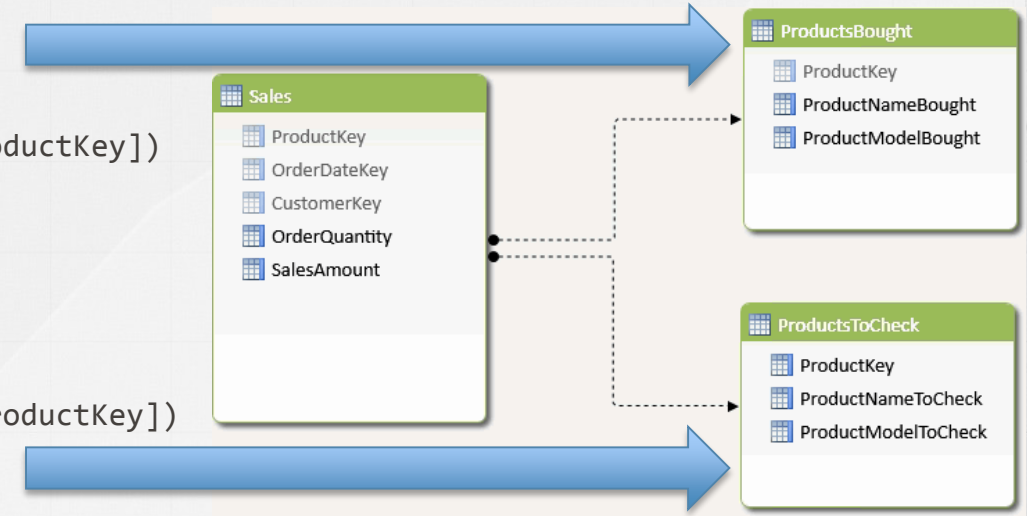
Inactive Relationships



sqlbi

The formula for HavingProducts

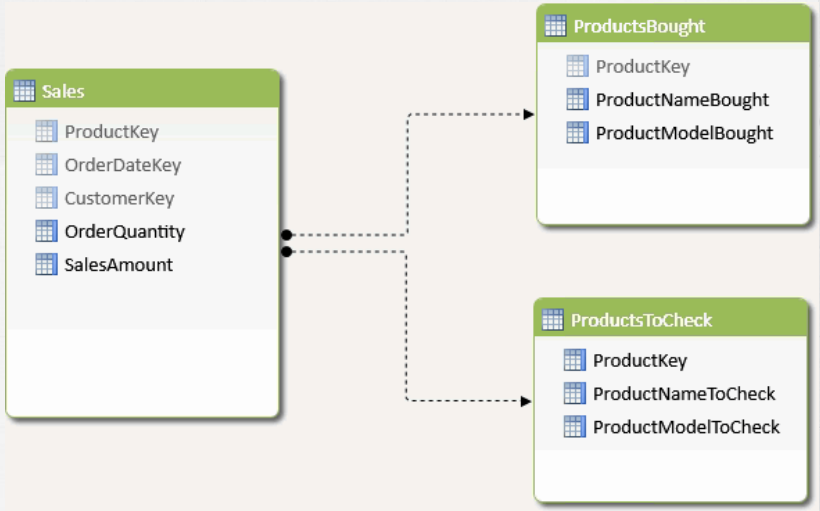
```
=COUNTROWS (
    CALCULATETABLE (
        FILTER (
            Customers;
            CALCULATE (
                COUNTROWS (Sales);
                USERELATIONSHIP (
                    Sales[ProductKey];
                    ProductsBought[ProductKey])
            ) > 0
            &&
            CALCULATE (
                COUNTROWS (Sales);
                USERELATIONSHIP (
                    Sales[ProductKey];
                    ProductsToCheck[ProductKey])
            ) > 0
        );
        FILTER (
            ALL (OrderDate);
            OrderDate[FullDate] <= MAX (OrderDate[FullDate])
        )
    )
)
```



sqlbi

Not HavingProducts

```
=COUNTROWS (
    CALCULATETABLE (
        FILTER (
            Customers;
            CALCULATE (
                COUNTROWS (Sales);
                USERELATIONSHIP (
                    Sales[ProductKey];
                    ProductsBought[ProductKey])
            ) > 0
            &&
            CALCULATE (
                COUNTROWS (Sales);
                USERELATIONSHIP (
                    Sales[ProductKey];
                    ProductsToCheck[ProductKey])
            ) = 0
        );
        FILTER (
            ALL (OrderDate);
            OrderDate[FullDate] <= MAX (OrderDate[FullDate])
        )
    )
)
```



sqlbi

Nice, and the many-to-many?

Where are the many-to-many patterns in this formula?

```
=COUNTROWS (
    CALCULATETABLE (
        FILTER (
            Customers;
            CALCULATE (
                COUNTROWS (Sales);
                USERELATIONSHIP (
                    Sales[ProductKey];
                    ProductsBought[ProductKey])
            ) > 0
            &&
            CALCULATE (
                COUNTROWS (Sales);
                USERELATIONSHIP (
                    Sales[ProductKey];
                    ProductsToCheck[ProductKey])
            ) = 0
        );
        FILTER (
            ALL (OrderDate);
            OrderDate[FullDate] <= MAX (OrderDate[FullDate])
        )
    )
)
```

1° M2M Pattern

2° M2M Pattern

sqlbi

Leveraging Tabular Features

- Multiple relationships between tables
  - Role Dimensions
  - Role Keys
- Active / Inactive relationships
- USERELATIONSHIP
  - New filter function
  - Selects a model relationship to use in calculations

sqlbi

Many to Many - Conclusions

- Learn with COUNTROWS
- Implement with SUMMARIZE
  - Faster
  - More concise
- If SUMMARIZE is not enough
  - Revert back to COUNTROWS
- Many powerful patterns with many-to-many
- Very fast in Tabular when compared with Multidimensional

sqlbi

Thank you!

Check daily our new articles on

www.sqlbi.com

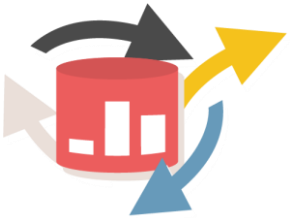
Oh, just another thing…

Are you sure to get the best out of your analytical environment?

Ask to SQLBI, we offer many services to help you:

Consulting

Assessment



Outsourcing

Technical Fellowship

Find out more on

www.sqlbi.com/consulting