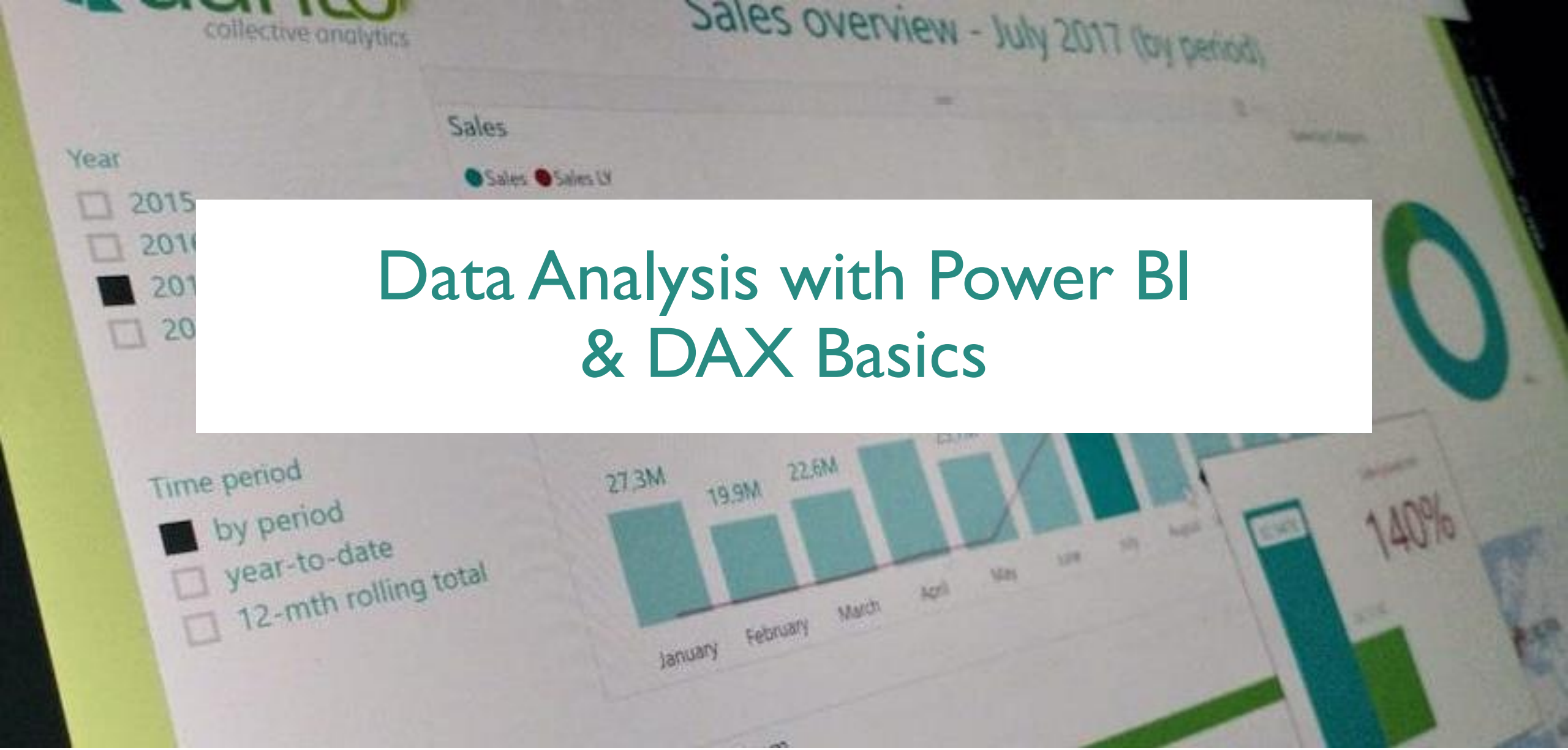


Data Analysis with Power BI & DAX Basics

Quanto
collective analytics

Day Schedule – Day 2

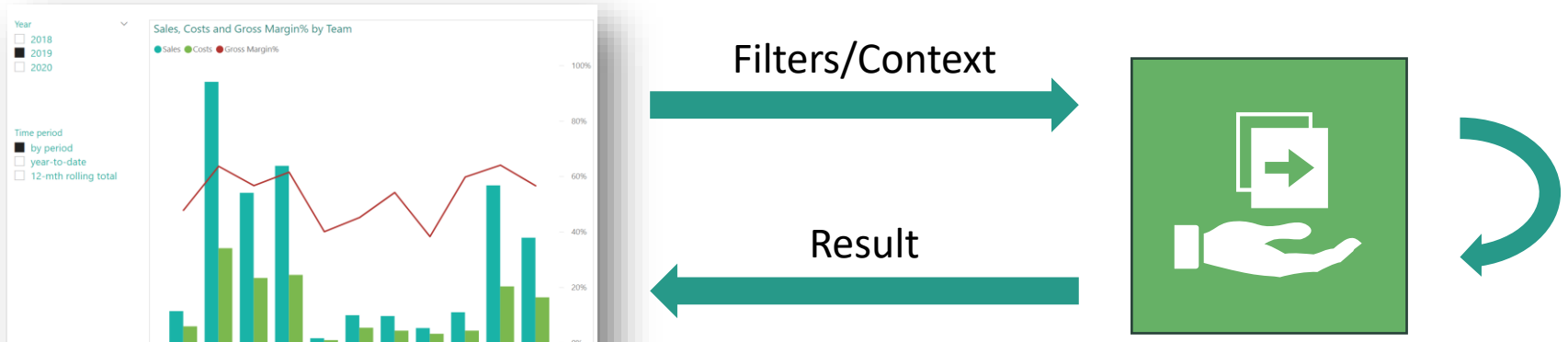
9:30	Module 1 – 1:30 hr	Recap: Modeling, Context, Filters
11:00	Break	
11:15	Module 2 – 1 hr	Storage modes, relationships (advanced)
12:15	Lunch	
13:15	Module 3 – 1:45 hr	Special filter arguments in CALCULATE
15:00	Break	
15:15	Module 4 – 1:45 hr	Time Intelligence
17:00	Closure	

Quanto
collective analytics

Filters in Visual Reports

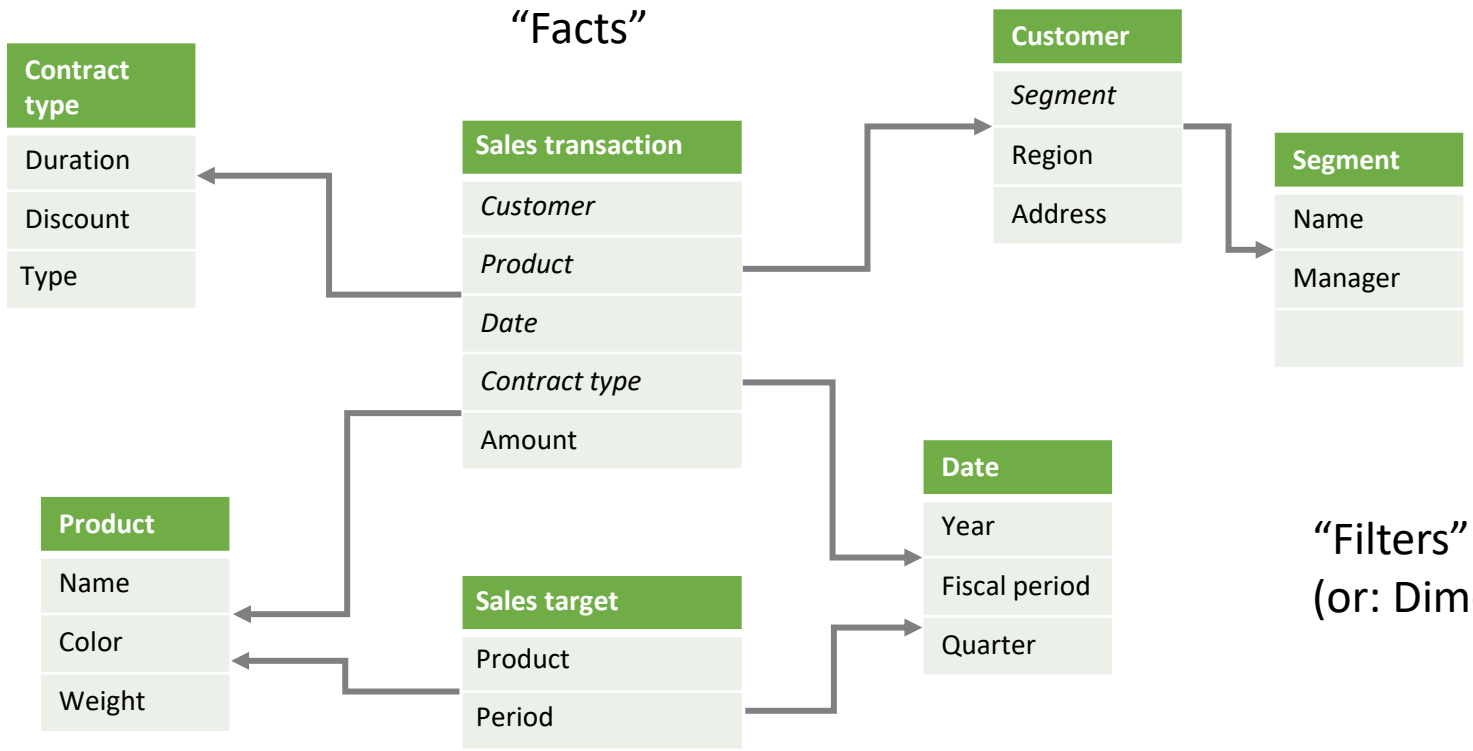
Filters are one of the most fundamental concepts in Power BI

- DAX is all about filters
- Visuals in a report provide filters, forming the context for DAX calculations



Quanto
collective analytics

Structure of a model

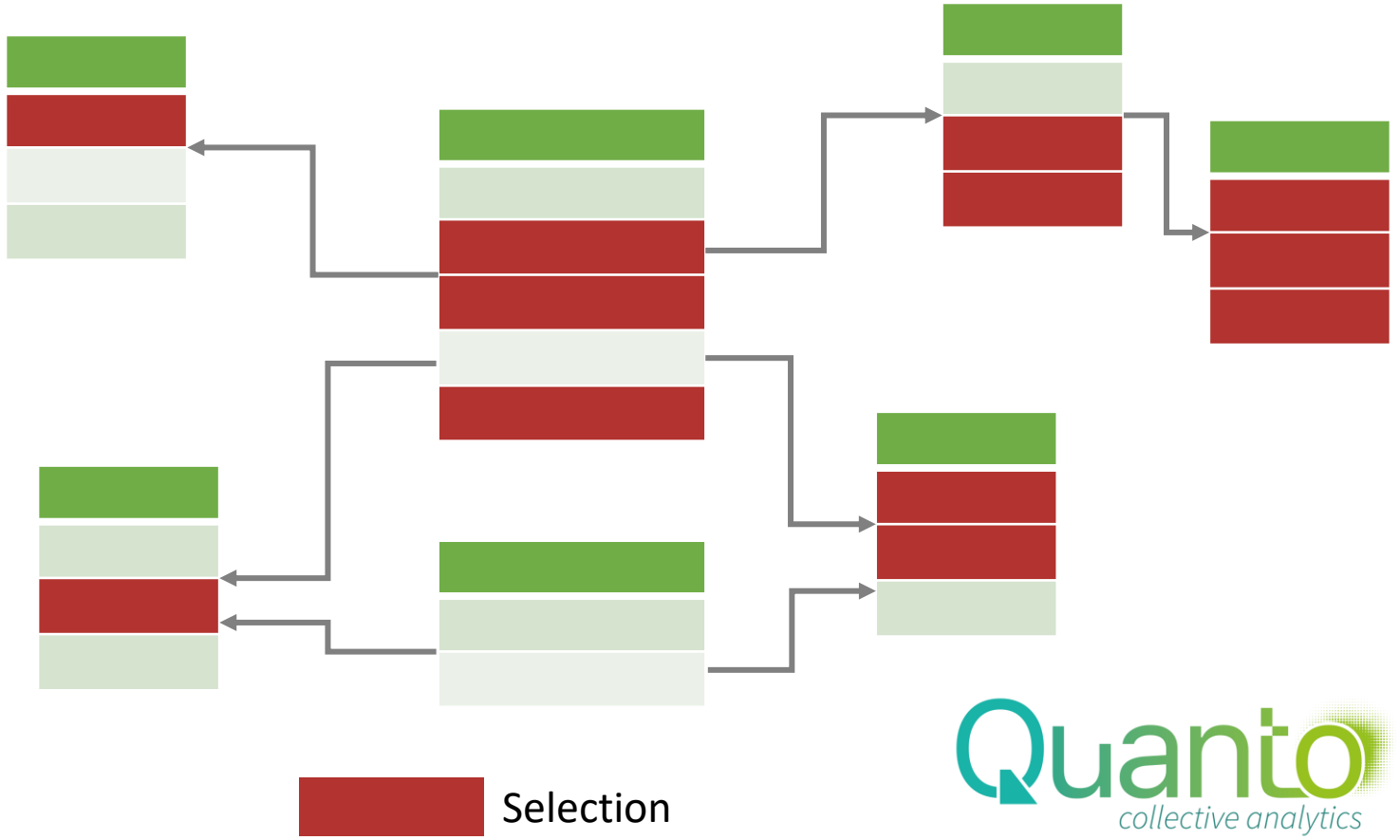


Quanto
collective analytics

Query context: selection

Selection in the report (slicers, page filters, row- or axis labels) correspond to rows in filter tables being selected

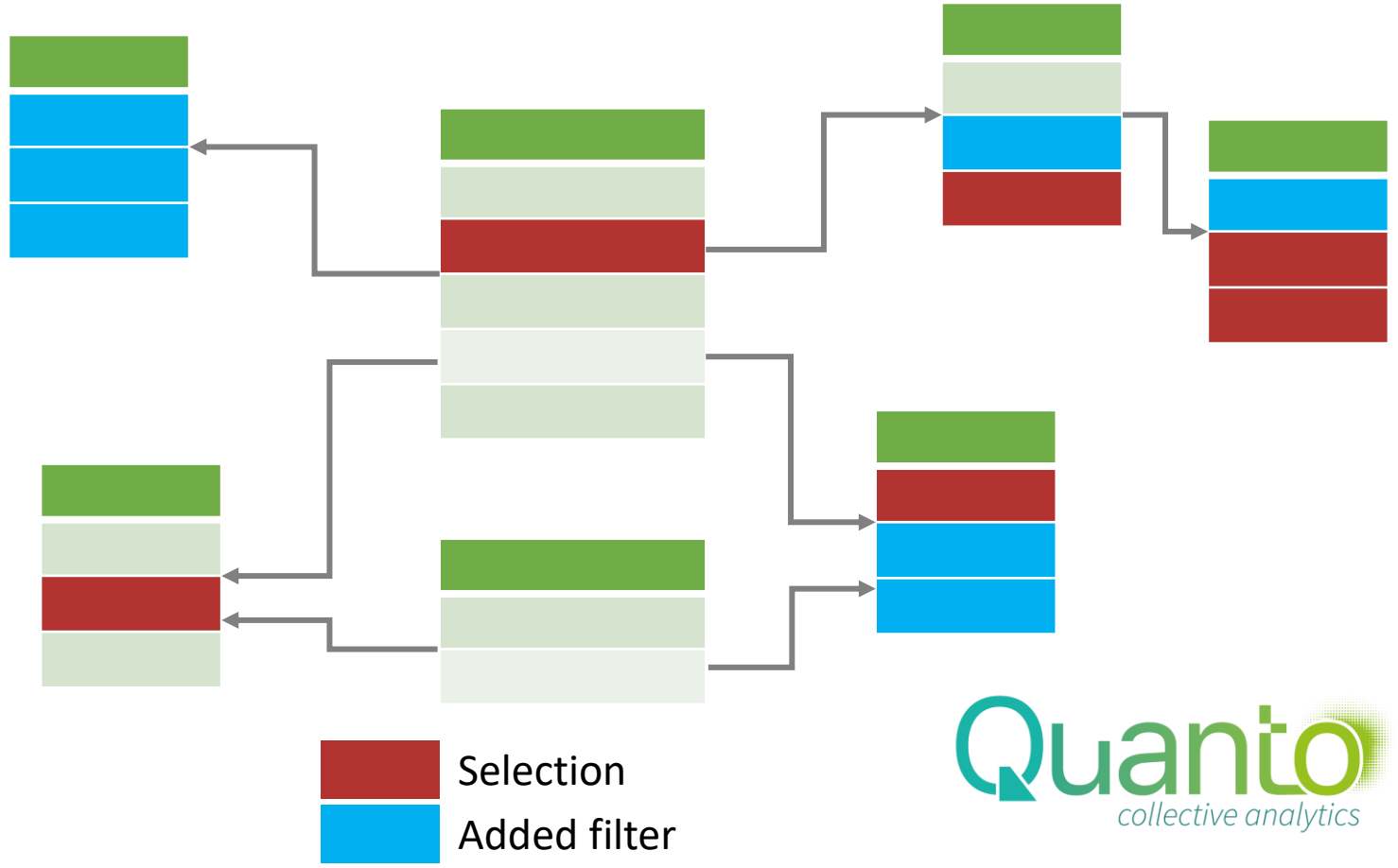
Relationships propagate selection resulting in selection of a number of rows in the fact tables



Quanto
collective analytics

Filter context

- Filter context is a context transformed with DAX
- Filter context behaves like query context
- But: you cannot derive filter context from what you see in the report



Quanto
collective analytics

Filtering with DAX: CALCULATE()

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

SalesWest = CALCULATE([Sales], Customer[Region]="West")

CALCULATE does four things:

1. Create a filter context
2. Remove existing filters from columns or tables that are referenced in the filter arguments: Customer[Region]
3. Add new filters: Customer[Region]="West"
4. Compute <expression> in the new filter context

Quanto
collective analytics

Exercise

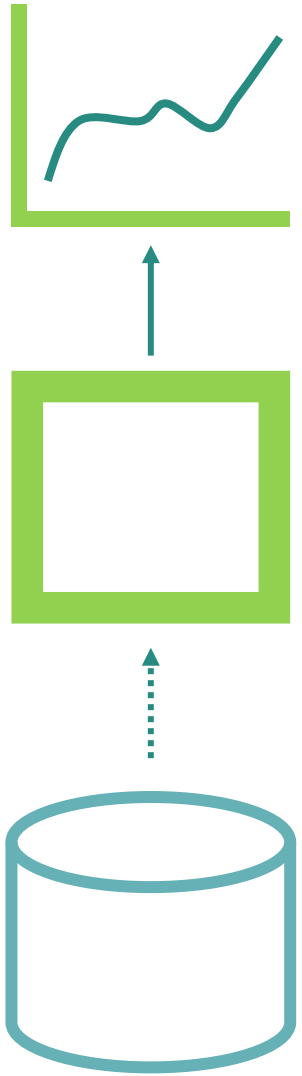
Filtering with CALCULATE()

1. *Create a measure to compute, in any context, the revenue on the product with ProductNr 303*

Power BI Storage Modes

Quanto
collective analytics

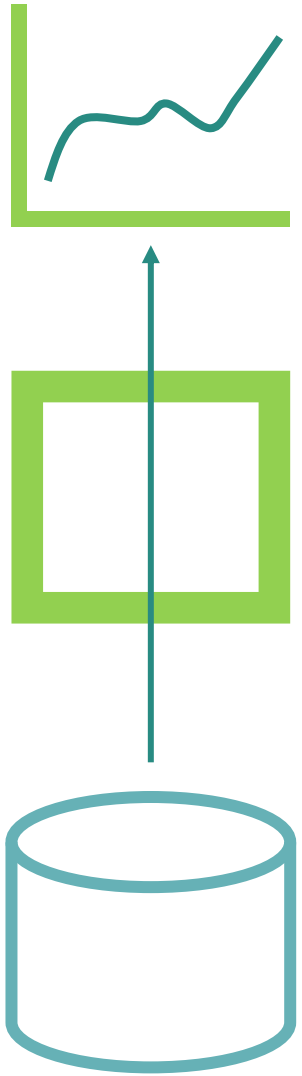
Power BI Data Model options (1)



Import

Quanto
collective analytics

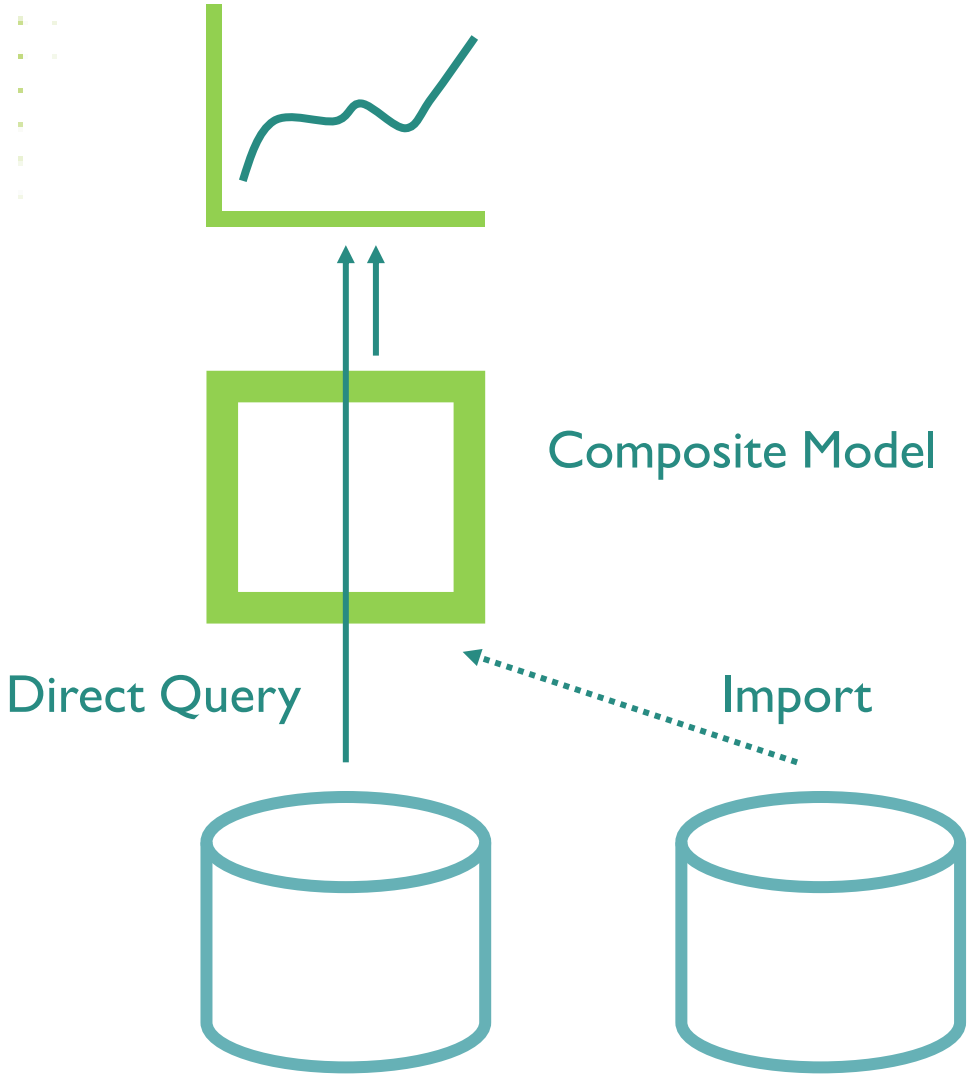
Power BI Data Model options (2)



Direct Query

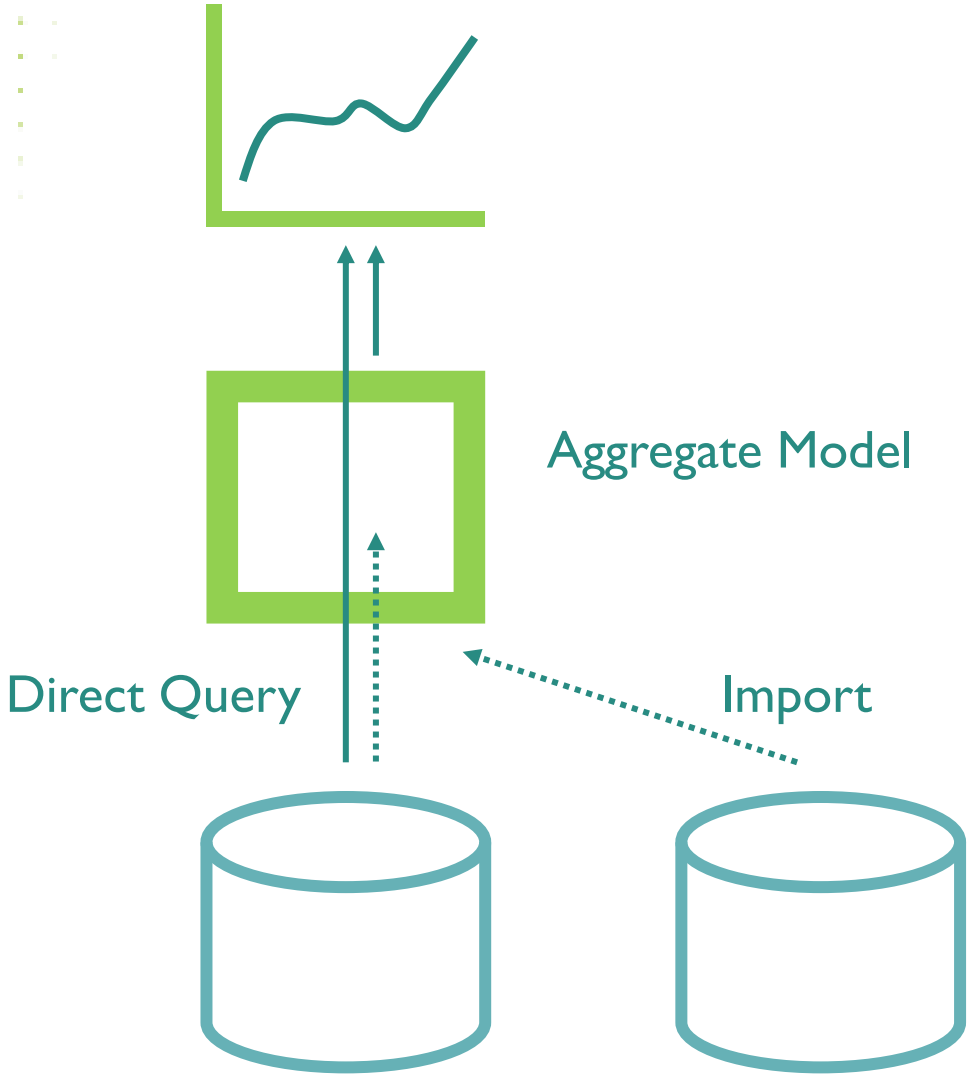
Quanto
collective analytics

Power BI Data Model options (3)



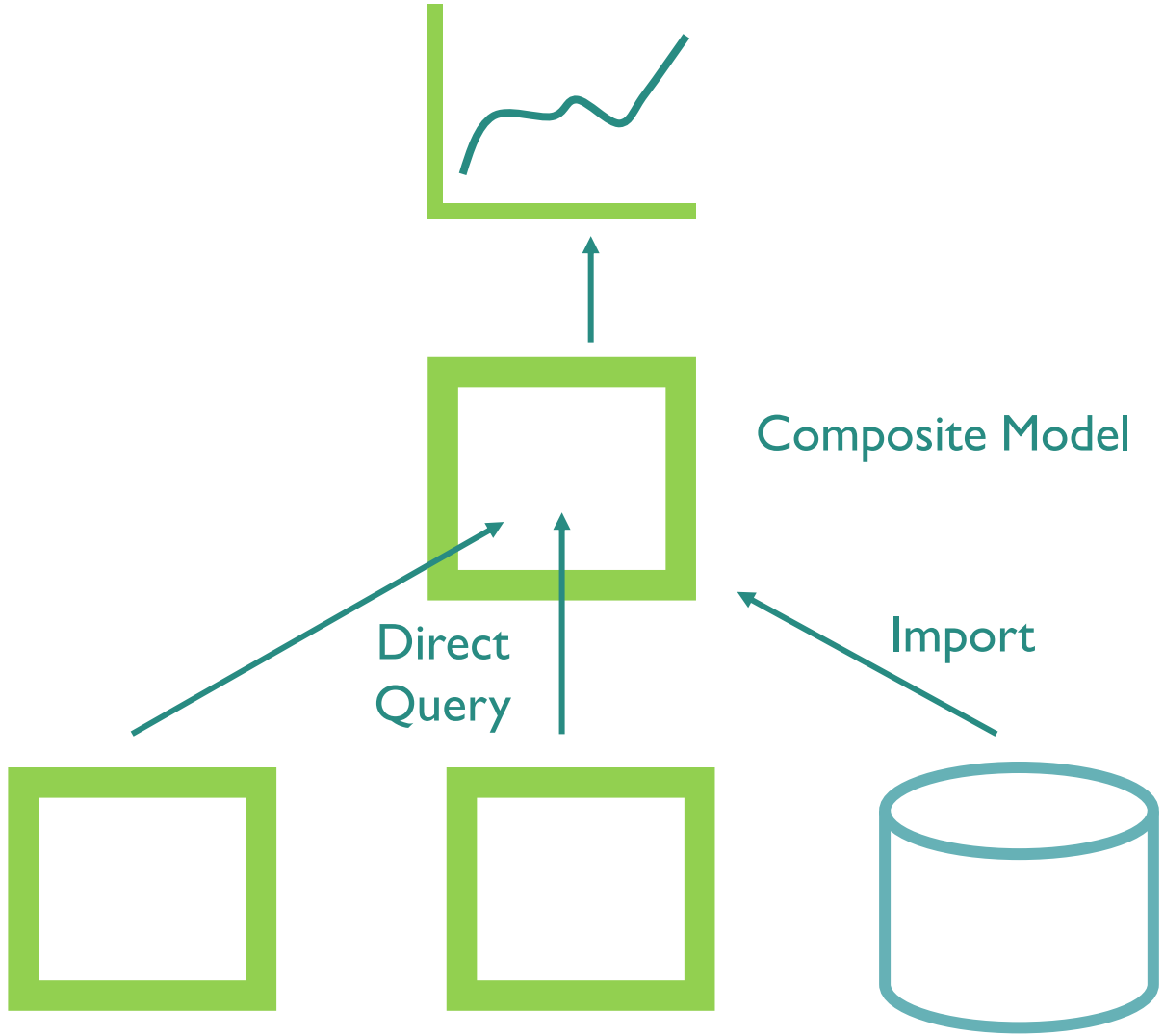
Quanto
collective analytics

Power BI Data Model options (4)



Quanto
collective analytics

Power BI Data Model options (5)



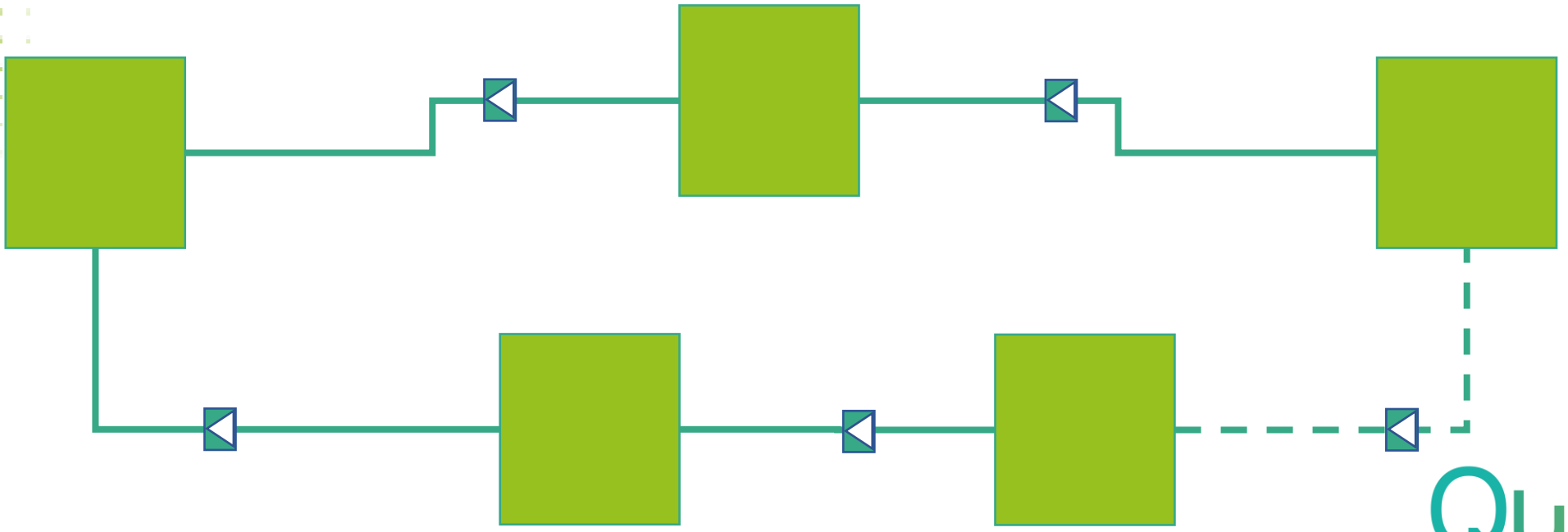
Quanto
collective analytics

Relationships

Quanto
collective analytics

Active and inactive relationships

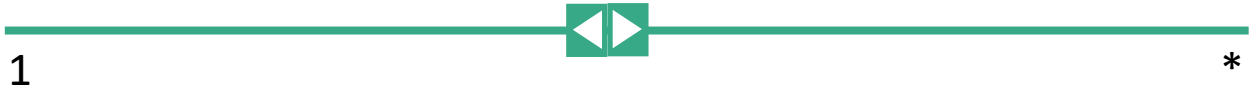
- Because relationships join table automatically, only one active path of relationships can exist between any two tables
  - Enable an inactive relationship/path with USERELATIONSHIP function



Quanto
collective analytics

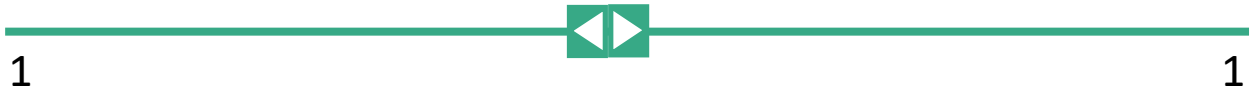
Other types of relationships

- Relationships with double crossfilter direction



- Many-to-many

- One-to-one

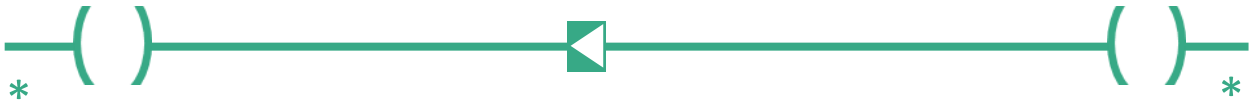


Use these only for specific reasons and when you know what you're doing!

Quanto
collective analytics

Limited relationships

- Many-to-many Relationships are limited



- Relationships between tables in different source groups are always limited
- Limited relationships behave a bit different on filter propagation

Quanto
collective analytics

Considerations for relationships in Power BI

- Relationships are between single columns only
- For relationships, less key values is better (keep it < 100k)
- Use ‘good’ data types for relationship keys when possible
- Don’t load data you don’t need
  - E.g. Calendar running from 1900 to 2100 ?!
- Double crossfilter direction is slow, use only on small tables
  - Or even better, use it only in DAX measures with CROSSFILTER

Quanto
collective analytics

Relationships in the data model

- Go from a source table to a lookup table, and from a source column to a lookup column
  - This is comparable to foreign keys and primary keys in relational databases
- The data model supports relationships on single columns only
- The lookup column must contain unique values, even if they're blank
- Not every value in the source table has to exist in the lookup table – missing values are linked to a blank row
  - So relationships are not constraints as in relational databases
- It is possible to have multiple relationships between tables, but only one can be active
- Best practice: hide the source column!

Quanto
collective analytics

ALL functions

These functions remove existing filters:

- ALL(Table[Column]) removes a filter from Table[Column]
  - Works with multiple columns as well
- ALL(Table) removes all filters from Table (all columns)
- ALLEXCEPT(Table, Table[Column]) removes all filters on columns in Table, except those on Table[Column]
  - Works with multiple columns as well

ALL functions (2)

- ALLNOBLANKROW(Table) removes all filters from Table, but does not include the blank row resulting from incomplete relationships
- ALLNOBLANKROW(Table[Column]) does the same for Table[Column]

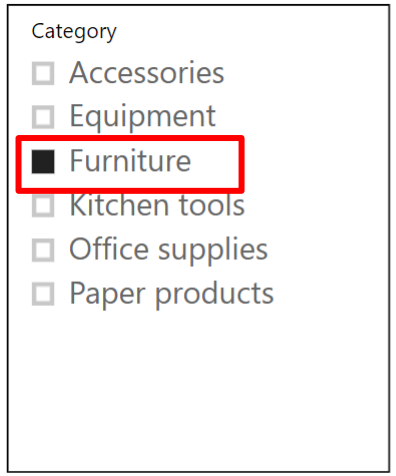
ALL functions (3)

More ALL options:

- ALL() removes all filters from all tables
- ALLCROSSFILTERED(Table) removes all filters from Table, including filters on tables with a many-to-many relationship to Table

Stacked filters

There can be more filters on the same column



Category	Product	Number Sold	Number sold all products
Furniture	Coat rack	189.145	20.129.846
Furniture	Conference chair	234.190	20.129.846
Furniture	Conference table	48.305	20.129.846
Furniture	Desk	141.421	20.129.846
Furniture	Desk lamp	237.316	20.129.846
Furniture	Drawer	190.845	20.129.846
Furniture	Office chair	139.686	20.129.846
Total		**1.180.908**	**20.129.846**

Most DAX functions cannot distinguish between these

```
Number sold all products = CALCULATE([Number Sold], ALL('Product'))
```

Quanto collective analytics

Stacked filters (2)

But there's one DAX function that does: ALLSELECTED()

```
Number sold all products in visual =
CALCULATE([Number Sold], ALLSELECTED('Product'))
```

Category

- ☐ Accessories
- ☑ Equipment
- ☑ Furniture
- ☐ Kitchen tools
- ☐ Office supplies
- ☐ Paper products

Category	Product	Number Sold	Number sold all products	Number sold ALLSELECTED
Equipment	Calculator	360.169	20.129.846	2.126.884
Furniture	Coat rack	189.145	20.129.846	2.126.884
Furniture	Conference chair	234.190	20.129.846	2.126.884
Furniture	Conference table	48.305	20.129.846	2.126.884
Equipment	Copier	76.905	20.129.846	2.126.884
Furniture	Desk	141.421	20.129.846	2.126.884
Furniture	Desk lamp	237.316	20.129.846	2.126.884
Furniture	Drawer	190.845	20.129.846	2.126.884
Equipment	Fax system	112.693	20.129.846	2.126.884
Equipment	Laser printer	112.859	20.129.846	2.126.884
Equipment	Latop	255.393	20.129.846	2.126.884
Furniture	Office chair	139.686	20.129.846	2.126.884
Equipment	PoS system	27.957	20.129.846	2.126.884
Total		**2.126.884**	**20.129.846**	**2.126.884**

Quanto collective analytics

ALL functions (3)

- ALLSELECTED() removes all filters that come from row or column labels, retaining filters from slicers and report filters
- ALLSELECTED(Table) does the same, but only for filters on Table
- ALLSELECTED(Table[Column]) does the same, but only for filters on Table[Column]

Exercise
ALLSELECTED()

1. *Create a measure to compute the percentage of the revenue for a product (or group of products) compared to all products in the visual*

Some special filter functions

USERELATIONSHIP()

- Is used to activate an inactive relationship
- E.g. CALCULATE([Revenue], USERELATIONSHIP(fSales[Payment date], Date[Date]))

KEEPFILTERS()

- This changes the behavior of CALCULATE: instead of *replacing* existing filters, filters in the CALCULATE statement are *added* to the existing context
- E.g. CALCULATE(<expression>, KEEPFILTERS(<filter>))

Quanto
collective analytics

Some special filter functions (2)

CROSSFILTER()

- Is used to change the filter propagation behaviour of a relationship
- CROSSFILTER(Table1[Column1], Table2[Column2], <Mode>)
  - Both: relationship uses two-way crossfiltering
  - None: relationship uses no crossfiltering
  - OneWay: relationship uses default, one-way crossfiltering
- E.g. CALCULATE([Revenue], CROSSFILTER(fSales[Payment date], Date[Date], Both))
- Mostly used to enable bi-directional crossfiltering, but can be used as an alternative to ALL() without loosing the context on the filter table

Quanto
collective analytics

Exercise
CALCULATE()

1. *Create a measure to compute the value of goods delivered*
2. *Create a measure that returns the revenue on product 304, only if that product is in the context*

Time Intelligence functions

Quanto
collective analytics

Time intelligence functions

- Allow for intelligent filtering over time
- Prerequisite: the Date or Calendar table
  - One row per day
  - Other columns for filtering
  - All fact tables relating to the Date table
  - Use TRUNC([Date]) when needed
  - Use the ‘Mark as Date table’ feature
  - Date[Date] is an argument of every time intelligence function

Quanto
collective analytics

What TOTALYTD() does

TOTALYTD() is a shortcut for a YTD filter:

```
TOTALYTD([Revenue], Date[Date])
```

is equivalent to

```
CALCULATE([Revenue], DATESYTD(Date[Date]))
```

DATESYTD replace the current context on the Date table by the YTD period (from 1st day of the year to last day in the current context)

Note that TOTALYTD() can be used with any calculation, not only SUM(), e.g.

```
TOTALYTD([AvgPrice], Date[Date]))
```

returns the year-to-date average price

Quanto
collective analytics

About ‘Mark as Date Table’

- Time Intelligence functions have special filtering behaviour: they add an implicit ALL(<Date Table>) as a filter.
- This will be done automatically when the relationship between a fact table and the Date table is created on a column with the Date data type.
- For other relationships (like ‘20210825’), this does not happen, unless the table is marked as date table.
  - Note that using real dates is better from an encoding perspective!

Time intelligence filters

- **SAMEPERIODLASTYEAR()** – returns the existing context, but one year earlier
- STARTOFYEAR(), ENDOFYEAR() – return the first day of the year, and the last day, respectively
  - STARTOFQUARTER(), STARTOFMONTH(), ENDOFQUARTER(), ENDOFMONTH()
- FIRSTDATE(), LASTDATE() – return the first day of the current context, and the last day, respectively
- DATEADD() – returns the current context, shifted forward or backward by a number of periods
  - E.g. DATEADD(Date[Date],-2,quarter)
- PARALLELPERIOD() – same as DATEADD(), but returns complete periods
- **DATESINPERIOD()** – returns a period extending from a start date to a number of intervals earlier or later
  - E.g. DATESINPERIOD(Date[Date],TODAY(),-6,month)
- **DATESBETWEEN()** – returns a period between two dates
  - E.g. DATESBETWEEN(Date[Date],[Start_Date],[End_Date])

Quanto
collective analytics

Time intelligence filters (2)

- PREVIOUSYEAR(), PREVIOUSQUARTER(), PREVIOUSMONTH(), PREVIOUSDAY() – return whole periods before the current context
- NEXTYEAR(), NEXTQUARTER(), NEXTMONTH(), NEXTDAY() – return whole periods after the current context

Note: the reference date to determine the next or previous period differs between functions – always double check!

Quanto
collective analytics

Shortcut time intelligence functions

- TOTALYTD(), TOTALQTD(), TOTALMTD()

- OPENINGBALANCEYEAR() etc.
- CLOSINGBALANCEYEAR() etc.

Exercise
Time intelligence

1. *Create a measure to compute the year-to-date revenue*
2. *Create a measure to compute the year-on-year revenue growth percentage*
3. *Create a measure to compute the 12-month rolling total revenue – DATESINPERIOD()*
4. *Create a measure to compute the cumulative total revenue – since the beginning of time – DATESBETWEEN()*