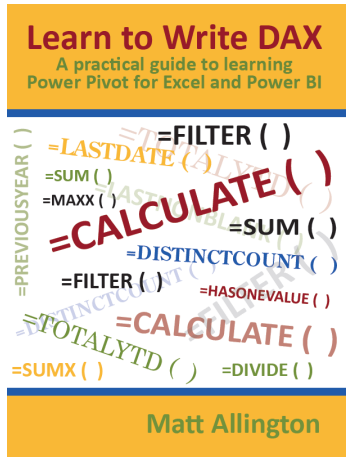
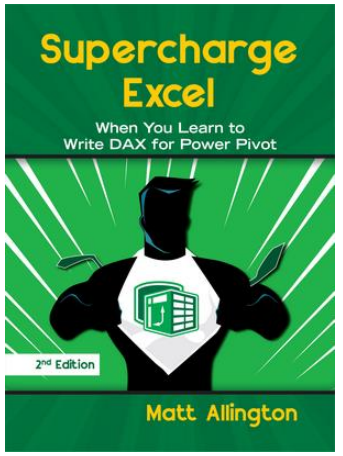
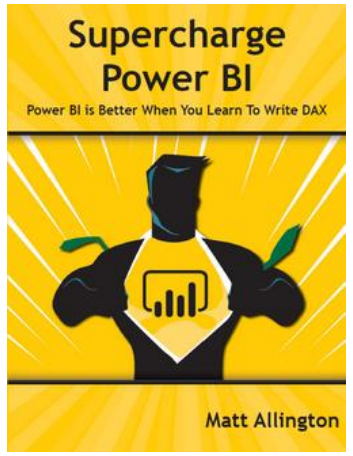
Time Intelligence Using DAX

About me

- 25 year career working at Coca-Cola in both Sales and IT
- Now running a Power BI consultancy in Sydney Australia
  - Self Service BI Consulting
  - Live Power BI Training
  - Online Training
  - Blogger http://xbi.com.au/blog
- Author of the book “Supercharge Power BI” and its derivations.
- Microsoft MVP







http://xbi.com.au/scpbib

Agenda

- Auto Time Intelligence
- Inbuilt Time Intelligence Functions
- Custom Time Intelligence Pattern
- Questions

3 Types of Time Intelligence

Auto Time Intelligence

- Automatically creates a calendar table for every date field in the data model.
- Easy to get started but limited capability (e.g. no weeks).
- Can make your files very large.
- Lacks a common lookup table to apply to many data tables.

Inbuilt Intelligence

- You must have a day level calendar table in your data model.
- Only works with standard calendar months (not 445).
- Not great for week and month level data (although you can hack it).
- There is a defined set of standard time intelligence functions that cover most scenarios

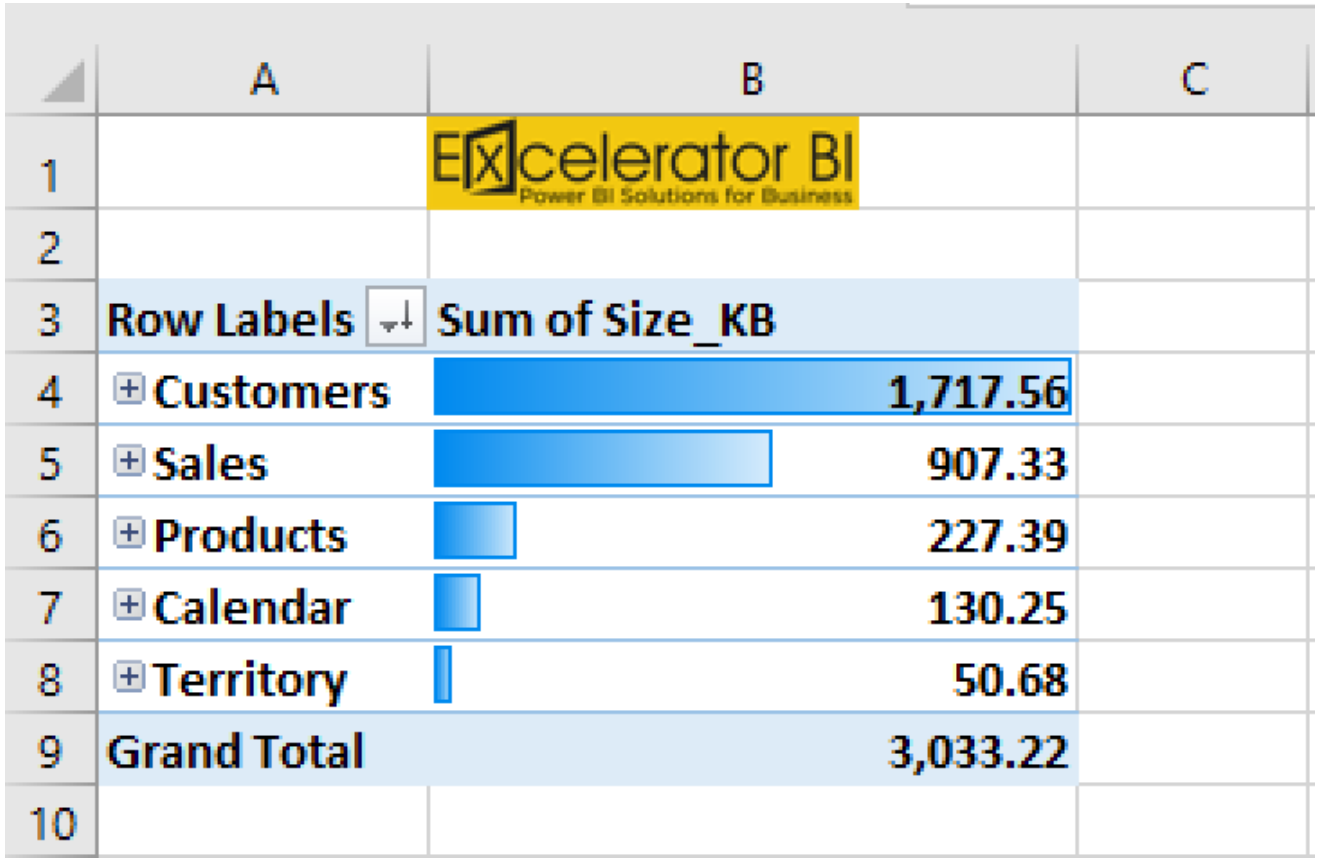
Custom Intelligence

- You should use a calendar table. It can be any granularity (e.g. from milliseconds to decades).
- Most flexible and can do anything you want.
- Hardest to learn, but you can learn it and it will sharpen you skills.

Demo

Auto Time Intelligence

Link to Local Host Workbook



	A	B	C
1		Excelerator BI Power BI Solutions for Business	
2			
3	Row Labels	Sum of Size_KB	
4	⊞ Customers	1,717.56	
5	⊞ Sales	907.33	
6	⊞ Products	227.39	
7	⊞ Calendar	130.25	
8	⊞ Territory	50.68	
9	Grand Total	3,033.22	
10			

http://xbi.com.au/localhost

Inbuilt Time Intelligence

Rules of a date table

Rules only apply if you want to use inbuilt Time Intelligence

- Must have a calendar table (or use the Auto Time Intelligence)
- Calendar table must have contiguous date range
  - Don’t skip any days
  - It “should” continue to the end of a full year (I rarely do this)
- Good date tables
  - Have all the columns that you will want to use in your filters and formulas.
  - Have numeric sort columns to control the way text is displayed.
- Gregorian Calendar only
  - Can have any financial year you specify, but no 445 calendars

http://exceleratorbi.com.au/power-pivot-calendar-tables/

Building Calendar Tables

- Excel
  - Very flexible but can be quite manual
- DAX Calendar Tables
  - https://github.com/sql-bi/DaxDateTemplate/blob/master/DAX Date Template.pbit
- Power Query (my preference)
  - https://exceleratorbi.com.au/build-reusable-calendar-table-power-query/

Demo

Inbuilt Time Intelligence

Reference Guide for Inbuilt Time Intelligence

Download a free copy of my quick reference guide from my online shop at http://xbi.com.au/ebooks

DAX Time Intelligence Functions

DAX Time Intelligence Functions support the needs of Business Intelligence analysis by enabling you to manipulate data using time periods, including days, months, quarters and years, and then build and compare calculations over those periods.

Function	Notes
CLOSINGBALANCEMONTH (expression, dates, [filter])	Evaluates the expression at the last date of the month in the current context.
CLOSINGBALANCEQUARTER (expression, dates, [filter])	Evaluates the expression at the last date of the quarter in the current context.
CLOSINGBALANCEYEAR (expression, dates, [filter], [year_end_date])	Evaluates the expression at the last date of the year in the current context.
DATEADD (dates, number_of_intervals, interval)	Returns a table that contains a column of dates, shifted either forward or backward in time by the specified number of intervals from the dates in the current context.
DATESBETWEEN (dates, start_date, end_date)	Returns a table that contains a column of dates that begins with the **start_date** and continues until the **end_date**.
DATESINPERIOD (dates, start_date, number_of_intervals, interval)	Returns a table that contains a column of dates that begins with the **start_date** and continues for the specified **number_of_intervals**.
DATESMTD (dates)	

Custom Time Intelligence

General Advice – Custom Time Intelligence

- Create an integer column that increments continuously for every time dimension that you want to use inside formulas.
- Don’t embed inbuilt time intelligence functions inside custom time intelligence functions.
  - Can create unexpected behaviour
  - Probably OK if you are using VAR/RETURN syntax
- Use the VAR/RETURN syntax where possible
  - Easier to write
  - Easier to read and debug

Custom time intelligence pattern

```
Sales YTD =  CALCULATE(
                [Total Sales],
                FILTER(
                 All(Calendar),
                    Calendar[Date] <=MAX(Calendar[Date]) &&
                    Calendar[Year] =MAX(Calendar[Year])
                )
             )
```

CalendarYear	2002	
Row Labels	**Total Sales**	**Total Sales YTD**
January	$596,747	$596,747
February	$550,817	$1,147,563
March	$644,135	$1,791,698
April	$663,692	$2,455,391
May	$673,556	$3,128,947
June	$676,764	$3,805,711
July	$500,365	$4,306,076
August	$546,001	$4,852,077
September	$350,467	$5,202,544
October	$415,390	$5,617,934
November	$335,095	$5,953,030
December	$577,314	$6,530,344
Grand Total	**$6,530,344**	**$6,530,344**

1. You need ALL otherwise you can't access periods that are pre-filtered by the initial filter context
2. Naked Columns mean => apply the filter to this column in the table
3. Aggregators (like MAX) mean => read from the initial filter context

Custom Time Intelligence using Variables

```
Total Sales YTD Custom VAR Syntax =
VAR UnfilteredCalendar =
    ALL ( 'Calendar' ) //returns an unfiltered copy of the calendar table
VAR LastDateInFilter =
    MAX ( 'Calendar'[Date] ) //max returns the last date in the filtered table
VAR FilteredYear =
    MAX ( 'Calendar'[Year] ) // max returns the only year that is filtered
RETURN
    CALCULATE (
        [Total Sales],
        FILTER (
            UnfilteredCalendar,
            'Calendar'[Date] <= LastDateInFilter
                && 'Calendar'[Year] = FilteredYear
        )
    )
```

Custom Time Intelligence VAR Syntax

```
Total Sales YTD Custom VAR Syntax =
    VAR UnfilteredCalendar =
        ALL ( 'Calendar' ) //returns an unfiltered copy of the calendar table
    VAR LastDateInFilter =
        MAX ( 'Calendar'[Date] ) //max returns the last date in the filtered table
    VAR FilteredYear =
        MAX ( 'Calendar'[Year] ) // max returns the only year that is filtered
RETURN
  CALCULATE (
    [Total Sales],
    FILTER (
      UnfilteredCalendar,
      'Calendar'[Date] <= LastDateInFilter
        && 'Calendar'[Year] = FilteredYear
    )
  )
```

Demo

Rolling 90 Day Average

Questions?

Special Webinar Offer

20% discount for online training using coupon code* **webinar**

Online Power Query Training
http://xbi.com.au/pq

Online DAX Training
http://xbi.com.au/scpbiot

Excelerator BI
Power BI Solutions for Business

* Expires 14th June 2018

445 Calendars

- Many different approaches across different businesses

- One good approach
  - Identify your weeks/days each year in a column
  - Use VALUES to detect which are filtered for current year.
  - Complete the time shift
    - Go back in time to a previous year
    - YTD calculations etc

445 Calendars

DAX FORMATTER

```
Same Week Prior Year =
VAR SelectedYear =
    MAX ( Calendar445[Year] )
VAR PriorYear = SelectedYear - 1
VAR Weeks =
    VALUES ( Calendar445[Week of Year] )
RETURN
    CALCULATE (
        [Total Sales],
        FILTER ( ALL ( Calendar445 ), Calendar445[Year] = PriorYear ),
        Weeks
    )
```

Questions?

Follow up materials on my blog
http://xbi.com.au/blog

Special Webinar Offer

20% discount for online training using coupon code* **webinar**

Online Power Query Training
http://xbi.com.au/pq

Online DAX Training
http://xbi.com.au/scpbiot

Excelerator BI
Power BI Solutions for Business

* Expires 14th June 2018