# Introducing Data Analysis eXpressions (DAX)

William E. Pearson III
www.islandtechnologies.com
@Bill_Pearson

pluralsight
hardcore dev and IT training

# Objectives

- Introduce the Data Analysis Expressions (DAX) language
- Expose DAX functions within Calculations

# Introducing Data Analysis Expressions (DAX)

**DAX is a collection of**

- **Functions**
- **Operators**
- **Constants**

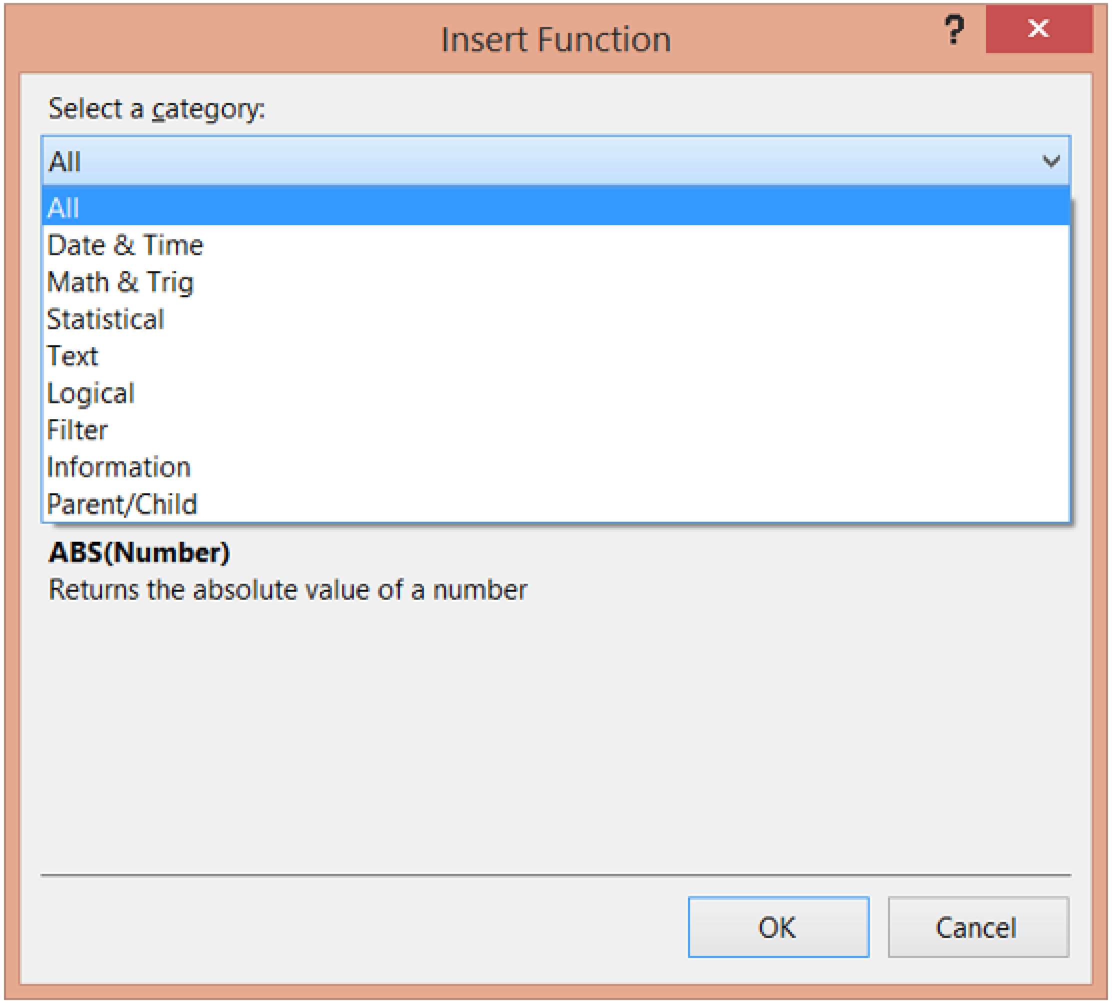
**… used to calculate and return one or more values**

***DAX helps us to create and present new information from our data***

# Data Analysis Expressions (DAX)

- **An expression language**
- **Used in Tabular calculated columns and measures**
- **Several types (with representative examples):**
  - Date & Time – MONTH(date), DAY(date)
  - Math & Trig – ROUND(number, num_digits)
  - Statistical – COUNT(value), AVERAGE(value)
  - Text – CONCATENATE(text, text2), TRIM(text)
  - Logical – IF(logical test, value if true, value if false)
  - Filter – RELATED(value)
  - Information – ISBLANK(value) return T/F
  - Parent / Child – PATH(ID_ColumnName, Parent_ColumnName)

# DAX Function Type Groupings



Select a category:

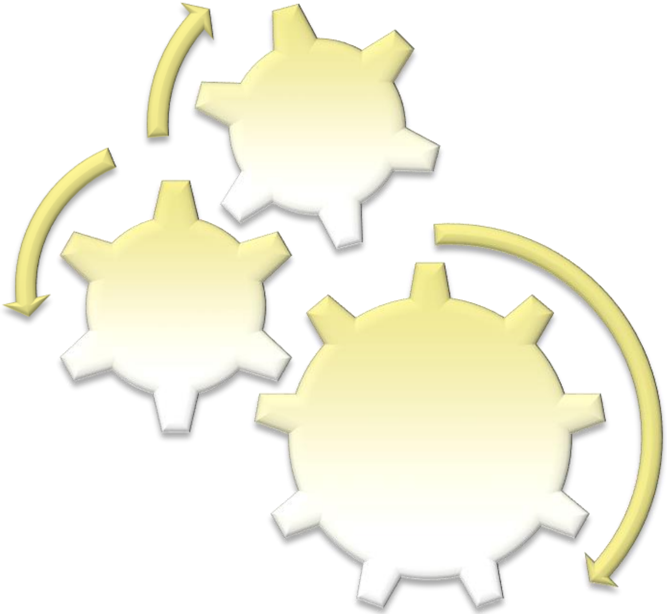
All

- All
- Date & Time
- Math & Trig
- Statistical
- Text
- Logical
- Filter
- Information
- Parent/Child

**ABS(Number)**

Returns the absolute value of a number

# Introducing DAX



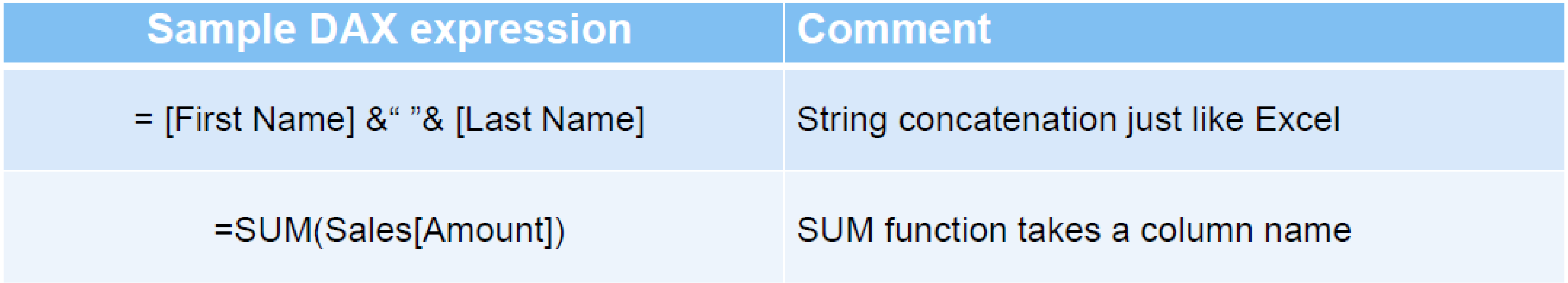
***Let's Take a Look:***

Examine the DAX Function Type Groupings in SSAS Tabular

# Data Analysis Expressions (DAX) …

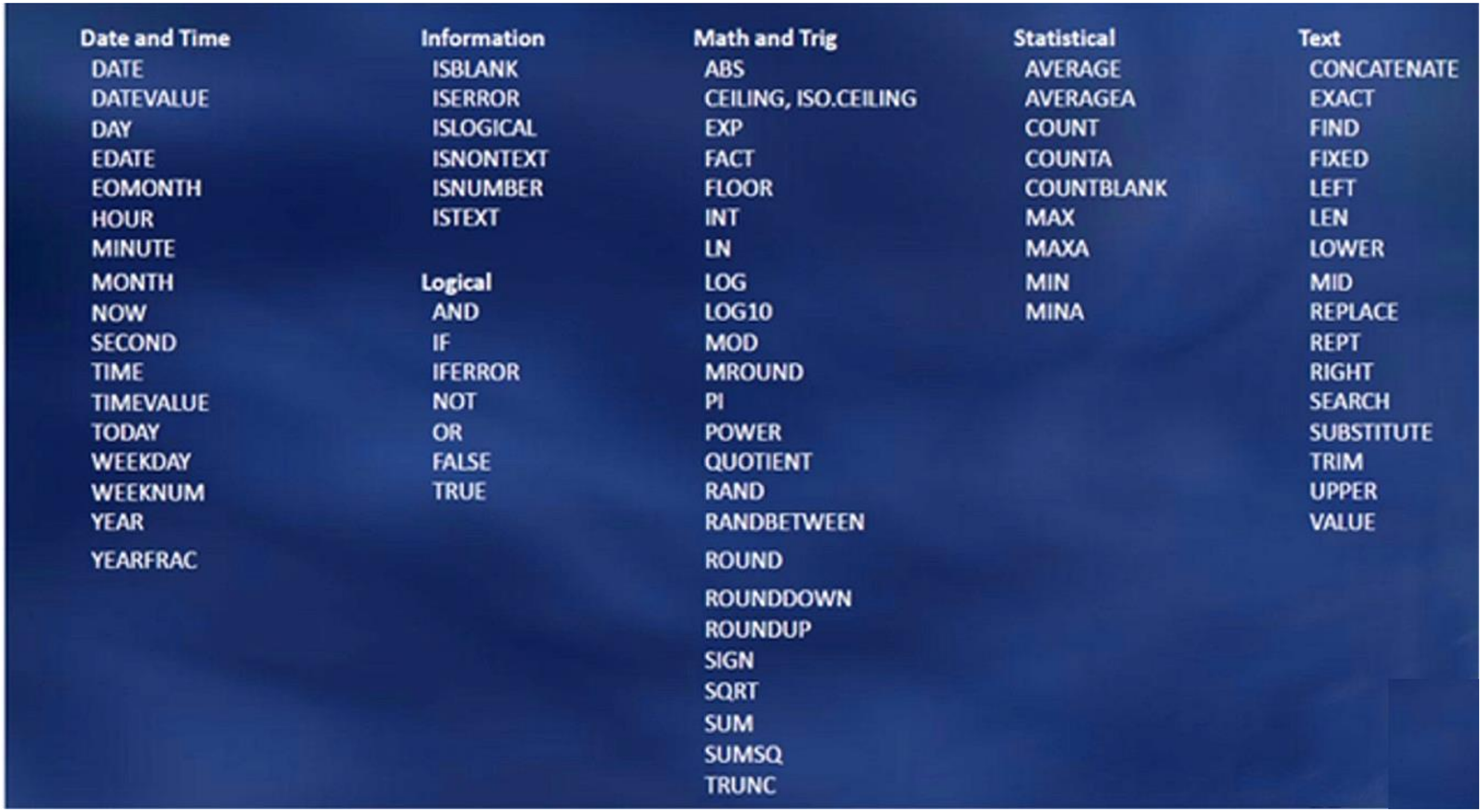
- **Use Excel syntax and many Excel functions**
- **Functions refer to columns, not cells / ranges**

**EXAMPLE: [Cost] or Product[Cost] or ‘Product Category’[Category Name]**



| Sample DAX expression | Comment |
|---|---|
| = [First Name] &“ ”& [Last Name] | String concatenation just like Excel |
| =SUM(Sales[Amount]) | SUM function takes a column name |

# DAX Included 80 Functions From Excel



### Date and Time

- DATE
- DATEVALUE
- DAY
- EDATE
- EOMONTH
- HOUR
- MINUTE
- MONTH
- NOW
- SECOND
- TIME
- TIMEVALUE
- TODAY
- WEEKDAY
- WEEKNUM
- YEAR
- YEARFRAC

### Information

- ISBLANK
- ISERROR
- ISLOGICAL
- ISNONTEXT
- ISNUMBER
- ISTEXT

### Logical

- AND
- IF
- IFERROR
- NOT
- OR
- FALSE
- TRUE

### Math and Trig

- ABS
- CEILING, ISO.CEILING
- EXP
- FACT
- FLOOR
- INT
- LN
- LOG
- LOG10
- MOD
- MROUND
- PI
- POWER
- QUOTIENT
- RAND
- RANDBETWEEN
- ROUND
- ROUNDDOWN
- ROUNDUP
- SIGN
- SQRT
- SUM
- SUMSQ
- TRUNC

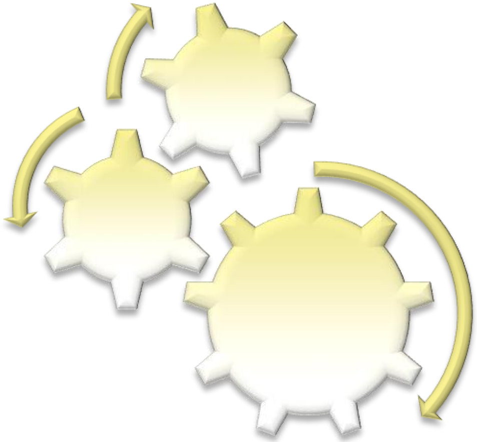
### Statistical

- AVERAGE
- AVERAGEA
- COUNT
- COUNTA
- COUNTBLANK
- MAX
- MAXA
- MIN
- MINA

### Text

- CONCATENATE
- EXACT
- FIND
- FIXED
- LEFT
- LEN
- LOWER
- MID
- REPLACE
- REPT
- RIGHT
- SEARCH
- SUBSTITUTE
- TRIM
- UPPER
- VALUE

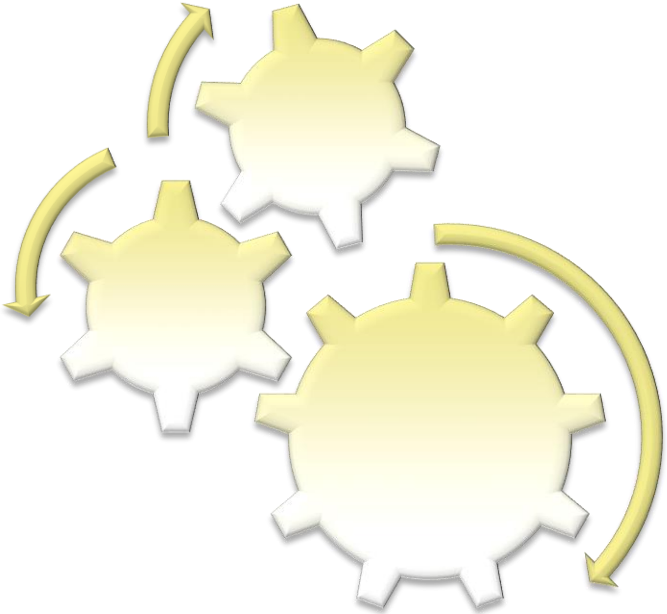
# Introducing DAX



## Preparation: Import  SQL Server 2012 Tables:

| **ADD:** | **RENAME:** |
|---|---|
| DimCustomer | Customer |
| DimDate | Date |
| DimGeography | Geography |
| DimProduct | Product |
| DimProductCategory | Product Category |
| DimProductSubCategory | Product Subcategory |
| FactInternetSales | Internet  Sales |

# Introducing DAX



***Let's Take a Look:***

Check Relationships for Accuracy and Completeness

# Introducing DAX

We’ll be using DAX in two places in Tabular:

- Calculated Columns
- Measures

# Calculated Columns - Review

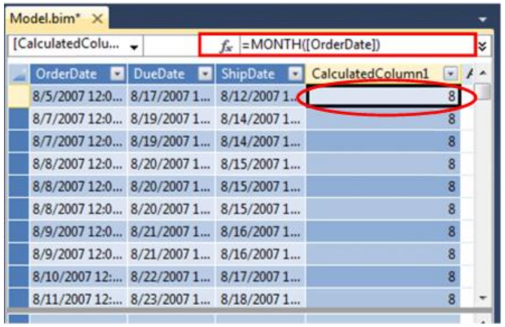
**Two ways to create Calculated Columns:**

**Type syntax to formula bar once we:**

- **Click row in right-most column, ("Add Column" )**

***or***

- **Right-click a column and select "Insert Column"**



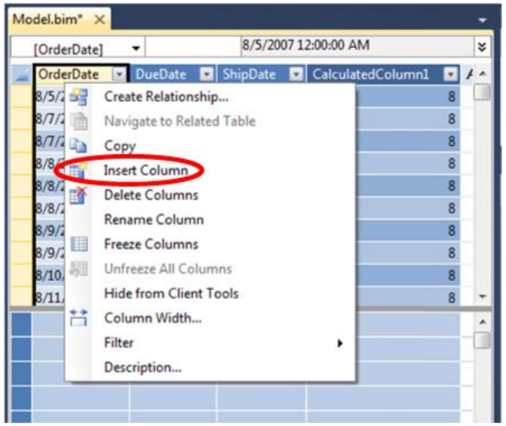
Model.bim*

[CalculatedColu...

=MONTH([OrderDate])

| OrderDate | DueDate | ShipDate | CalculatedColumn1 |
|---|---|---|---|
| 8/5/2007 12:0... | 8/17/2007 1... | 8/12/2007 1... | 8 |
| 8/7/2007 12:0... | 8/19/2007 1... | 8/14/2007 1... | 8 |
| 8/7/2007 12:0... | 8/19/2007 1... | 8/14/2007 1... | 8 |
| 8/8/2007 12:0... | 8/20/2007 1... | 8/15/2007 1... | 8 |
| 8/8/2007 12:0... | 8/20/2007 1... | 8/15/2007 1... | 8 |
| 8/8/2007 12:0... | 8/20/2007 1... | 8/15/2007 1... | 8 |
| 8/9/2007 12:0... | 8/21/2007 1... | 8/16/2007 1... | 8 |
| 8/9/2007 12:0... | 8/21/2007 1... | 8/16/2007 1... | 8 |
| 8/10/2007 12:... | 8/22/2007 1... | 8/17/2007 1... | 8 |
| 8/11/2007 12:... | 8/23/2007 1... | 8/18/2007 1... | 8 |

... OR ...



Model.bim*

[OrderDate]

8/5/2007 12:00:00 AM

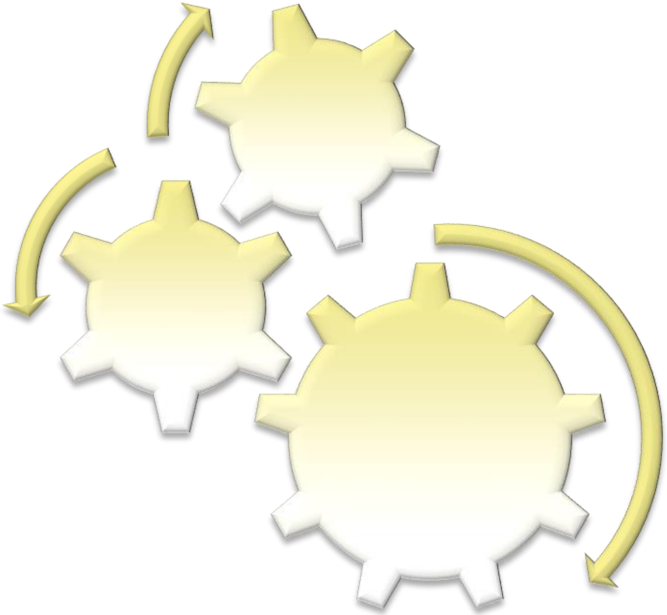
| OrderDate | DueDate | ShipDate | CalculatedColumn1 |
|---|---|---|---|
| 8/5/2 | | | 8 |
| 8/7/2 | | | 8 |
| 8/7/2 | | | 8 |
| 8/8/ | | | 8 |
| 8/8/2 | | | 8 |
| 8/8/2 | | | 8 |
| 8/9/2 | | | 8 |
| 8/9/2 | | | 8 |
| 8/10/ | | | 8 |
| 8/11/ | | | 8 |

# Calculated Columns

Calculated Columns:

- Single formula
- Always Row Context

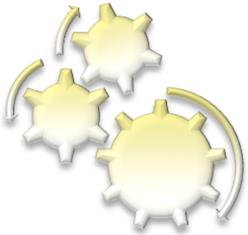
# Introducing DAX …



***Create Basic Calculated Columns in SSAS Tabular***

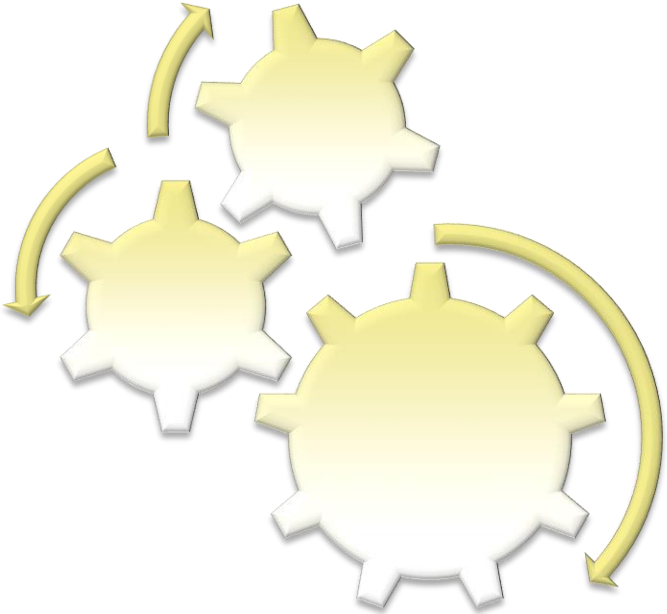
Using the Following List …

# Introducing DAX …



| CALCULATED COLUMN | SYNTAX |
|---|---|
| **Internet_Sales.Gross Margin** | ='Internet Sales'[SalesAmount] - 'Internet Sales'[TotalProductCost] |
| **Internet_Sales.Perc Gross Margin** | ='Internet Sales'[Gross Margin] /'Internet Sales'[SalesAmount]*100 |
| **Internet_Sales.Net Profit** | ='Internet Sales'[Gross Margin]-'Internet Sales'[TaxAmt]- 'Internet Sales'[Freight] |

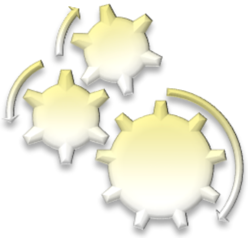
# Introducing DAX



*Create Basic Calculated Columns in SSAS Tabular*

Using the Following List …

# Introducing DAX …



| CALCULATED COLUMN | SYNTAX |
|---|---|
| **Internet_Sales.Customer Name** | =RELATED(Customer[LastName]) |
| **Internet_Sales.Customer Region** | =RELATED(Geography[EnglishCountryRegionName]) |
| **Internet_Sales.Product Name** | =RELATED(Product[EnglishProductName]) |

# Following Relationships … With the DAX RELATED() Function

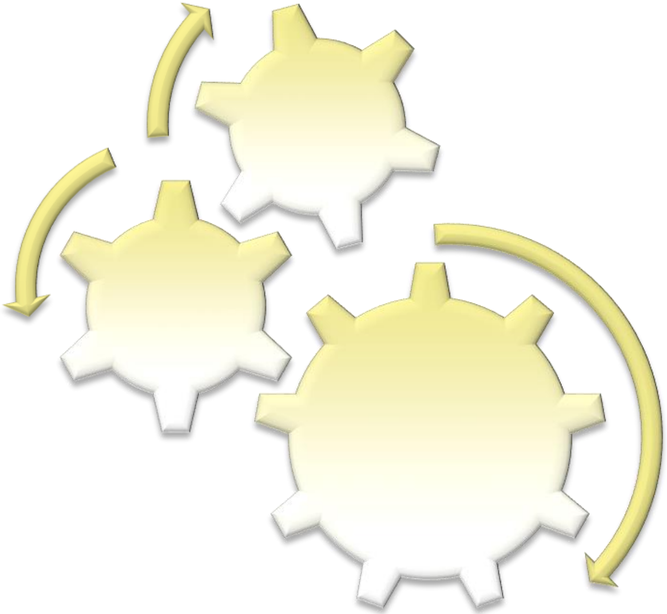
**RELATED(Column)**

- **Fetches value related table**
- **“Many” side to the “One” side**
- **Blank returned for missing values**
- **Can make multiple “hops”**

```
=RELATED(Customer[LastName])

=RELATED(Product[EnglishProductName])
```

# Introducing DAX



***Create a Calculated Column in SSAS Tabular***

Using a Text – type Function …

# Introducing DAX

| CALCULATED COLUMN | SYNTAX |
|---|---|
| **Customer.Full Name** | =Customer[LastName] & " , " & Customer[FirstName] |

# Concatenation … with “ & “ Operator or CONCATENATE() Function

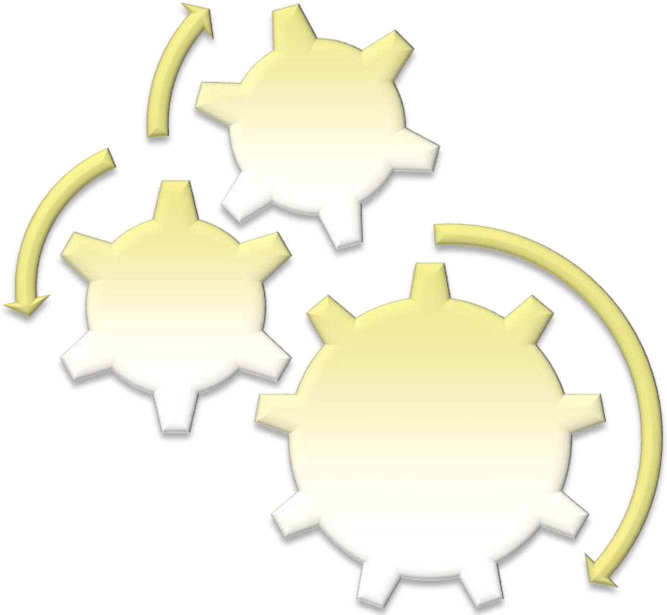
**Two ways to do basic concatenation:**

- **Ampersand (&) Operator**
- **CONCATENATE() Function**

```
=Customer[LastName] & " , " & Customer[FirstName]

=CONCATENATE(Customer[LastName]  , CONCATENATE(" , “ , Customer[FirstName]))
```

# Introducing DAX



***Create a Calculated Column in SSAS Tabular***

Using a Date & Time – type Function …

# Introducing DAX

| **CALCULATED COLUMN** | **SYNTAX** |
|---|---|
| **Customer.YrsAnniv** | =(TODAY() - Customer[DateFirstPurchase]) / 365 |

# Today's Date, Without Time Attached … With the TODAY() Function

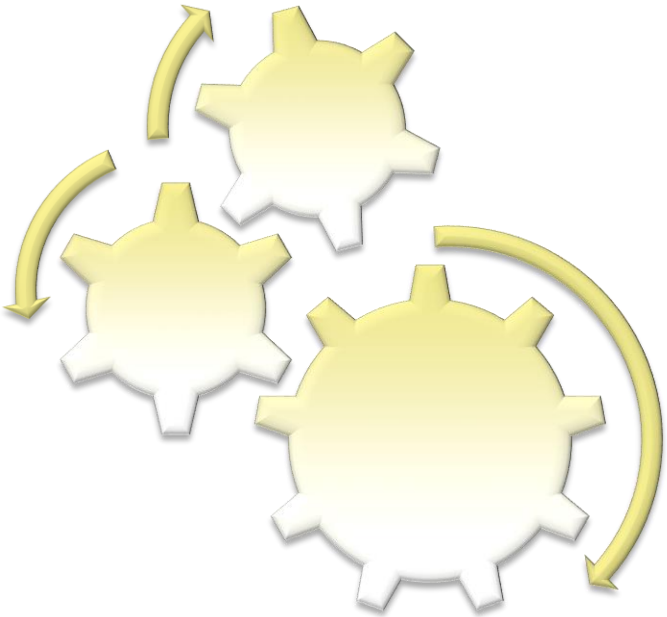
**TODAY(Datetime)**

- **Returns the current date**
- **Display current date upon opening the workbook**
- **Also useful for calculating intervals**

```
=(TODAY() - Customer[DateFirstPurchase]) / 365

=YEAR(TODAY()) - 1963
```

# Introducing DAX



***Create a Calculated Column in SSAS Tabular***

Using a Logical – type Function …

# Introducing DAX

| CALCULATED COLUMN | SYNTAX |
|---|---|
| **Customer.Customer Class** | =IF(Customer[YrsAnniv]>=8,"Gold",<br>IF(Customer[YrsAnniv]>=6,"Silver",<br>IF(Customer[YrsAnniv]>=2,"Bronze",<br>"Regular"))) |

# Conditional Logic …
# With the DAX IF() Function

IF(Logical_Test, Value_If_True, [Value_If_False])

- Checks if first argument is met
- Returns one value if TRUE, another value if FALSE
- IF() treats empty return value(s) as empty string ("")
- If column value referenced, IF() returns value for current row
- IF() attempts to return single data type
- If  return values are different data types,  IF() implicitly converts

# DAX Aggregation Functions

- **SUM, AVERAGE, MIN, MAX, COUNT take column references**
- **Extensions (“X” iterators) aggregate expression over a table’s rows**
  - □ SUMX(Table, Expression)
  - □ AVERAGEX(Table, Expression)
  - □ COUNTX(Table, Expression)
  - □ MINX(Table, Expression)
  - □ MAXX(Table, Expression)
  - □ COUNTROWS(Table)

# Some DAX Table Functions

| Function | Output |
|---|---|
| **FILTER(**Table, condition**)** | Table filtered to rows where condition true |
| **ALL(**Table)<br>**ALL(**Column) | All data for specified object, ignoring:<br>• context filters<br>• duplicates |
| **ALLEXCEPT(**Table, Col1, Col2, …) | All data for specified table, ignoring:<br>• context filters<br>… EXCEPT …<br>• filters for the specified columns retained |

# Time Intelligence Functions

- **Built-in 35 functions:**
  - Update filter context (***thus Measures only***)
  - For time-based analysis
- **Require Date column in data joined to Date table**
- **Work with intervals: days, months, quarters, and years**
- **Functions that:**
  - Return a single date
  - Return a table of dates
  - Evaluate expressions over the time period

# Functions That Return a Single Date …

- FirstDate (Date_Column)
- LastDate (Date_Column)
- FirstNonBlank (Date_Column, Expression)
- LastNonBlank (Date_Column, Expression)

- StartOfMonth (Date_Column)
- StartOfQuarter (Date_Column)
- StartOfYear (Date_Column [, YE_Date])

- EndOfMonth (Date_Column)
- EndOfQuarter (Date_Column)
- EndOfYear (Date_Column [, YE_Date])

# Functions That Return a Table of Dates …

- DateAdd (Date_Column, Number_of_Intervals, Interval)
- DatesBetween(Date_Column, Start_Date, End_Date)
- DatesInPeriod(Date_Column, Start_Date, Number_of_Intervals, Interval)
- ParallelPeriod (Date_Column, Number_of_Intervals, Interval)

- PreviousDay (Date_Column)
- PreviousMonth (Date_Column)
- PreviousQuarter (Date_Column)
- PreviousYear (Date_Column [, YE_Date])

- NextDay (Date_Column)
- NextMonth (Date_Column)
- NextQuarter (Date_Column)
- NextYear (Date_Column [, YE_Date])

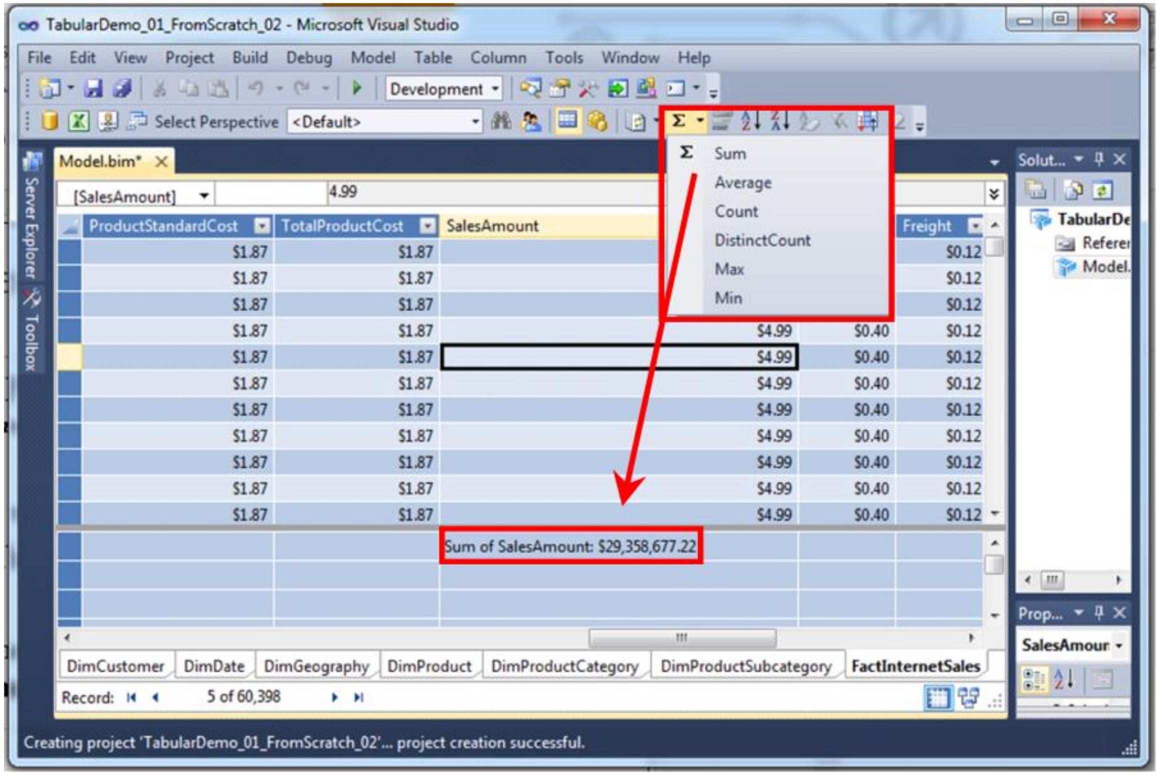
- DatesMTD (Date_Column)
- DatesQTD (Date_Column)
- DatesYTD (Date_Column [, YE_Date])
- SamePeriodLastYear (Date_Column)

# Measures - Review

- Measures are *pre-defined* data aggregations
- Creating Measures is simple and fast

# Measures - Review

## Simplest way to create Measures:



TabularDemo_01_FromScratch_02 - Microsoft Visual Studio

[SalesAmount] 4.99

| ProductStandardCost | TotalProductCost | SalesAmount | | Freight |
|---|---|---|---|---|
| $1.87 | $1.87 | | | $0.12 |
| $1.87 | $1.87 | | | $0.12 |
| $1.87 | $1.87 | | | $0.12 |
| $1.87 | $1.87 | $4.99 | $0.40 | $0.12 |
| $1.87 | $1.87 | $4.99 | $0.40 | $0.12 |
| $1.87 | $1.87 | $4.99 | $0.40 | $0.12 |
| $1.87 | $1.87 | $4.99 | $0.40 | $0.12 |
| $1.87 | $1.87 | $4.99 | $0.40 | $0.12 |
| $1.87 | $1.87 | $4.99 | $0.40 | $0.12 |
| $1.87 | $1.87 | $4.99 | $0.40 | $0.12 |
| $1.87 | $1.87 | $4.99 | $0.40 | $0.12 |

Sum of SalesAmount: $29,358,677.22

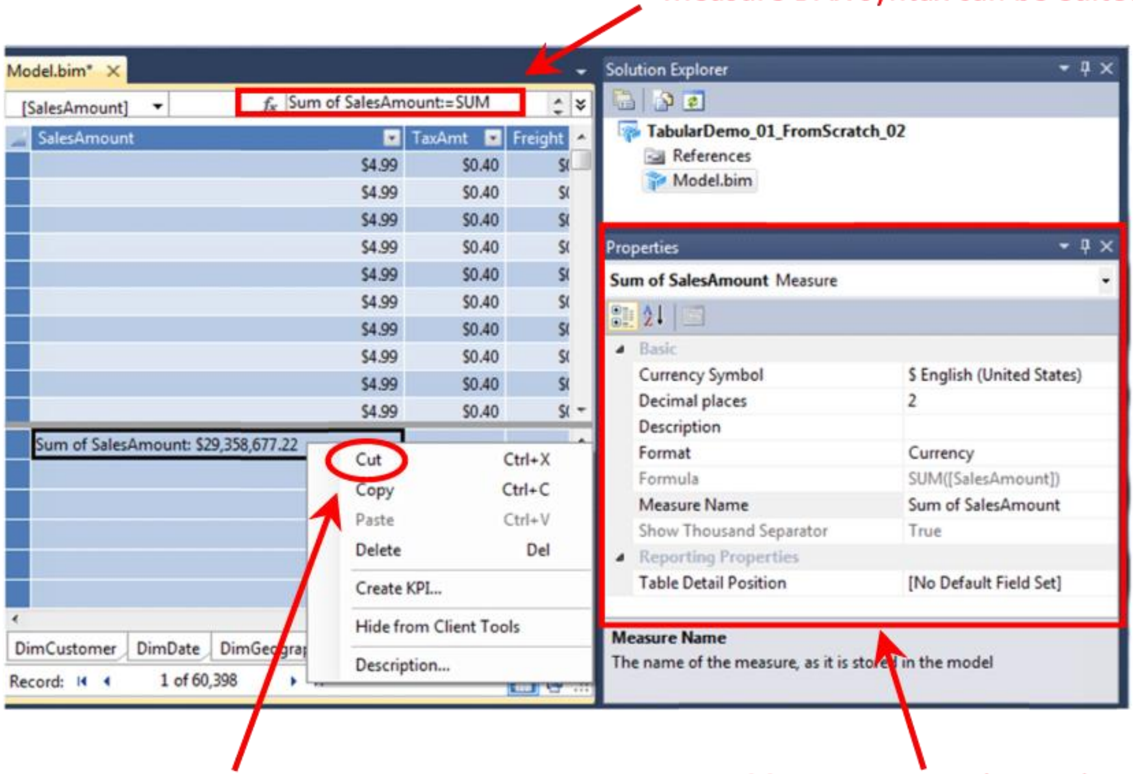
Record: 5 of 60,398

Creating project 'TabularDemo_01_FromScratch_02'... project creation successful.

# Measures - Review

## More about Measures in SSAS 2012 Tabular:

Measure DAX syntax can be edited ...



Sum of SalesAmount:=SUM

| SalesAmount | TaxAmt | Freight |
|---|---|---|
| $4.99 | $0.40 | $( |
| $4.99 | $0.40 | $( |
| $4.99 | $0.40 | $( |
| $4.99 | $0.40 | $( |
| $4.99 | $0.40 | $( |
| $4.99 | $0.40 | $( |
| $4.99 | $0.40 | $( |
| $4.99 | $0.40 | $( |
| $4.99 | $0.40 | $( |
| $4.99 | $0.40 | $( |

Sum of SalesAmount: $29,358,677.22

**Sum of SalesAmount** Measure

| Basic | |
|---|---|
| Currency Symbol | $ English (United States) |
| Decimal places | 2 |
| Description | |
| Format | Currency |
| Formula | SUM([SalesAmount]) |
| Measure Name | Sum of SalesAmount |
| Show Thousand Separator | True |
| **Reporting Properties** | |
| Table Detail Position | [No Default Field Set] |

**Measure Name**
The name of the measure, as it is stored in the model

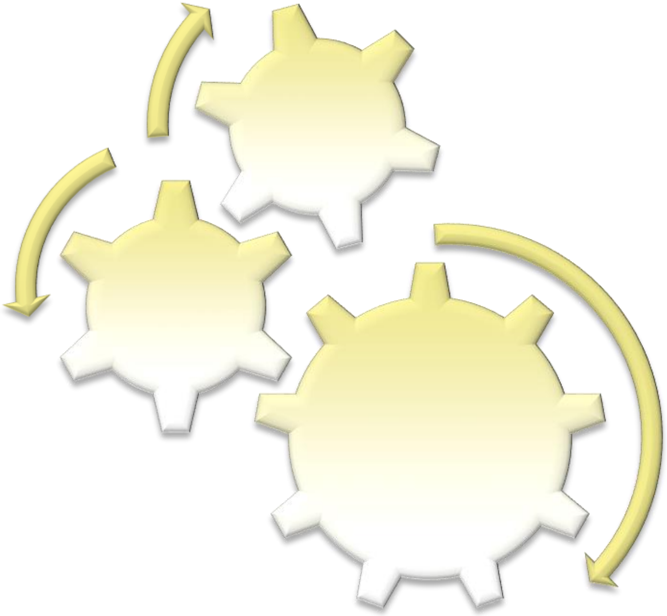
Measures can be relocated to a single region
of measure grid for centralized maintenance ...

Measure properties settings

# Measures – Context Considerations

- **Dynamic formulas: results depend upon context**
- **Used in reporting formats that:**
  - Combine / filter model data
  - Examples:  Excel PivotTable / PivotChart or Power View report
- **In Tabular, Measures are:**
  - Defined by model designer …
  - … Using SSDT measure grid / formula bar
- **Filtered results of a calculation:**
  - Cannot be seen immediately
  - Cannot be determined without context
  - Can only be seen via a reporting client that:
    - Retrieves data for each cell
    - Evaluates expression for each cell

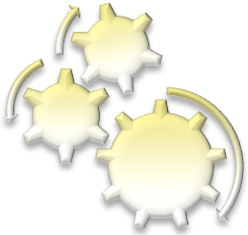
# Introducing DAX



*Create Measures in SSAS Tabular*

Using the CALCULATE() Function …

# Introducing DAX



| MEASURE | SYNTAX |
|---|---|
| **Product Subcategory.Total No SubCats** | Total No SubCats:=<br>CALCULATE( COUNTROWS('Product Subcategory'),<br>ALL( 'Product Subcategory'[EnglishProductSubcategoryName]) ) |
| **Product Subcategory. No Bike SubCats** | No Bike SubCats:=<br>CALCULATE( COUNTROWS('Product Subcategory'),<br>ALL( 'Product Subcategory'[EnglishProductSubcategoryName]),<br>'Product Subcategory'[ProductCategoryKey] = 1 ) |

# “Any Port in a (Contextual) Storm…” The DAX CALCULATE() Function

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

**If data has been filtered (PivotTable a typical Filter Context), CALCULATE():**

- **Changes context of filtering**
- **Evaluates expression in new context**

**For each column used in a filter argument:**

- **Existing filters on that column are removed**
- **Filter used in function filter argument is applied instead**

```
CALCULATE( COUNTROWS('Product Subcategory'),
  ALL( 'Product Subcategory'[EnglishProductSubcategoryName]),
    'Product Subcategory'[ProductCategoryKey] = 1  )
```

# Summary

- Introduced the Data Analysis Expressions (DAX) language
- Exposed DAX functions within Calculations