



Combining Tables

Combining Tables – Strategies



- Append
- Merge
- Relationships

Appending Tables



- Columns are stacked based on matching Column Names and Data Types
- Column names must match
- Data Types must match or lesser (*i.e., more generic*) data type is assigned

Appending Tables *(cont)*

SalesRep	Month	Product	Purchaser	Sale
Frank Killough	Jan	Stepper Machines	World of Sports	\$3,498.00
Sandy Brady	Jan	Tennis Balls	Tennis Joint	\$1,796.00
Ernest Feldgus	Jan	Rowing Machines	Sportsman's Den	\$2,170.00
Ernest Feldgus	Jan	Baseballs	Sportsman's Den	\$2,591.00
Alice Abramas	Jan	Gloves	SportsCity	\$3,097.00

Product	Month	SalesRep	Sale	Purchaser
Stepper Machines	Feb	John Carpenter	\$4,369.00	Sports Emporium
Footballs	Feb	Fred Edwards	\$3,444.00	SportsWorld
Baseballs	Feb	Frank Mann	\$3,289.00	Sports Emporium
Rowing Machines	Feb	Fred Edwards	\$3,349.00	SportsWorld
Tennis Balls	Feb	Susan Edwards	\$1,912.00	Tennis Joint

SalesRep	Purchaser	Product	Sale	Month
Terry Caracio	Athlete's Dream	Rowing Machines	\$3,053.00	Mar
Terry Caracio	Athlete's Dream	Golf Balls	\$3,034.00	Mar
Frank Killough	World of Sports	Footballs	\$2,334.00	Mar
Perry Weinstein	Specialty Sports	Stepper Machines	\$4,253.00	Mar
Fred Edwards	SportsWorld	Gloves	\$2,890.00	Mar



SalesRep	Month	Product	Purchaser	Sale
Frank Killough	Jan	Stepper Machines	World of Sports	\$3,498.00
Sandy Brady	Jan	Tennis Balls	Tennis Joint	\$1,796.00
Ernest Feldgus	Jan	Rowing Machines	Sportsman's Den	\$2,170.00
Ernest Feldgus	Jan	Baseballs	Sportsman's Den	\$2,591.00
Alice Abramas	Jan	Gloves	SportsCity	\$3,097.00
John Carpenter	Feb	Stepper Machines	Sports Emporium	\$4,369.00
Fred Edwards	Feb	Footballs	SportsWorld	\$3,444.00
Frank Mann	Feb	Baseballs	Sports Emporium	\$3,289.00
Fred Edwards	Feb	Rowing Machines	SportsWorld	\$3,349.00
Susan Edwards	Feb	Tennis Balls	Tennis Joint	\$1,912.00
Terry Caracio	Mar	Rowing Machines	Athlete's Dream	\$3,053.00
Terry Caracio	Mar	Golf Balls	Athlete's Dream	\$3,034.00
Frank Killough	Mar	Footballs	World of Sports	\$2,334.00
Perry Weinstein	Mar	Stepper Machines	Specialty Sports	\$4,253.00
Fred Edwards	Mar	Gloves	SportsWorld	\$2,890.00

Merging Tables



- Columns from one table are “copy/pasted” to another table
- Like performing a VLOOKUP or XLOOKUP
- Each table must contain a common column that has the same data type
- Can be performed using Power Query’s **Merge Tables** feature
- Can be performed using the DAX **RELATED** function (*flows Many to 1*)

Class = RELATED(tblCatalog[Class Name])

Merging Tables *(cont.)*

TransID	Product	PurchID	Sale
101	Tennis Balls	AA1	\$ 1,912.00
102	Footballs	BB2	\$ 2,334.00
103	Basketballs	CC3	\$ 2,004.00
104	Golf Balls	BB2	\$ 1,553.00
105	Rowing Machines	CC3	\$ 2,170.00

PurchID	Purchaser
AA1	Tennis Joint
BB2	World of Sports
CC3	Athlete's World

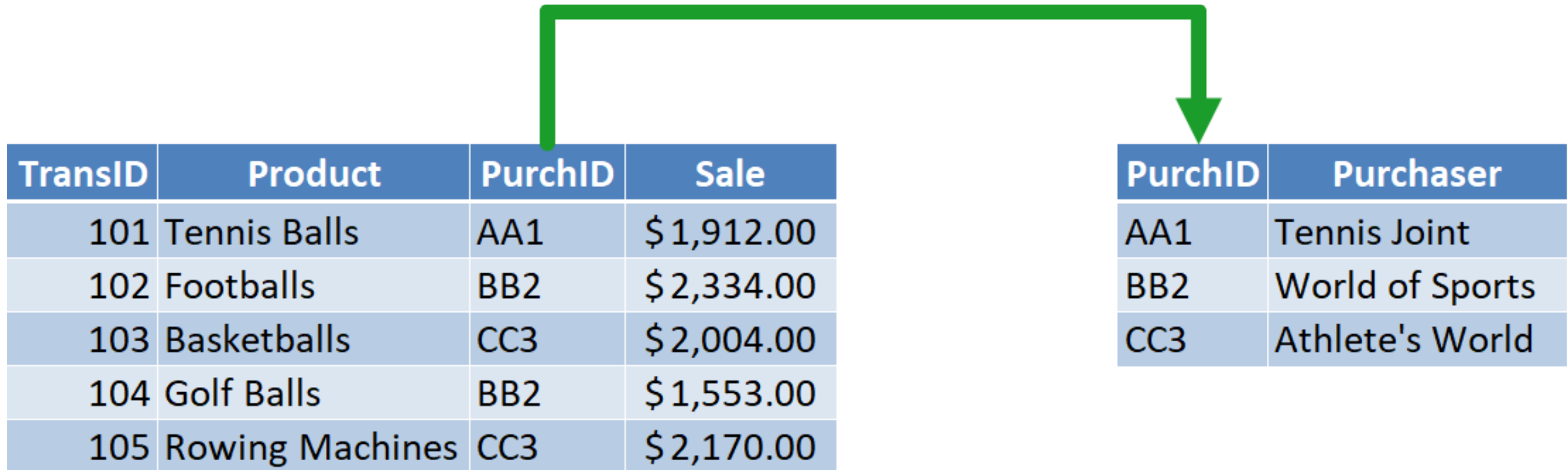
TransID	Product	PurchID	Sale	Purchaser
101	Tennis Balls	AA1	\$ 1,912.00	Tennis Joint
102	Footballs	BB2	\$ 2,334.00	World of Sports
103	Basketballs	CC3	\$ 2,004.00	Athlete's World
104	Golf Balls	BB2	\$ 1,553.00	World of Sports
105	Rowing Machines	CC3	\$ 2,170.00	Athlete's World

Creating Relationships



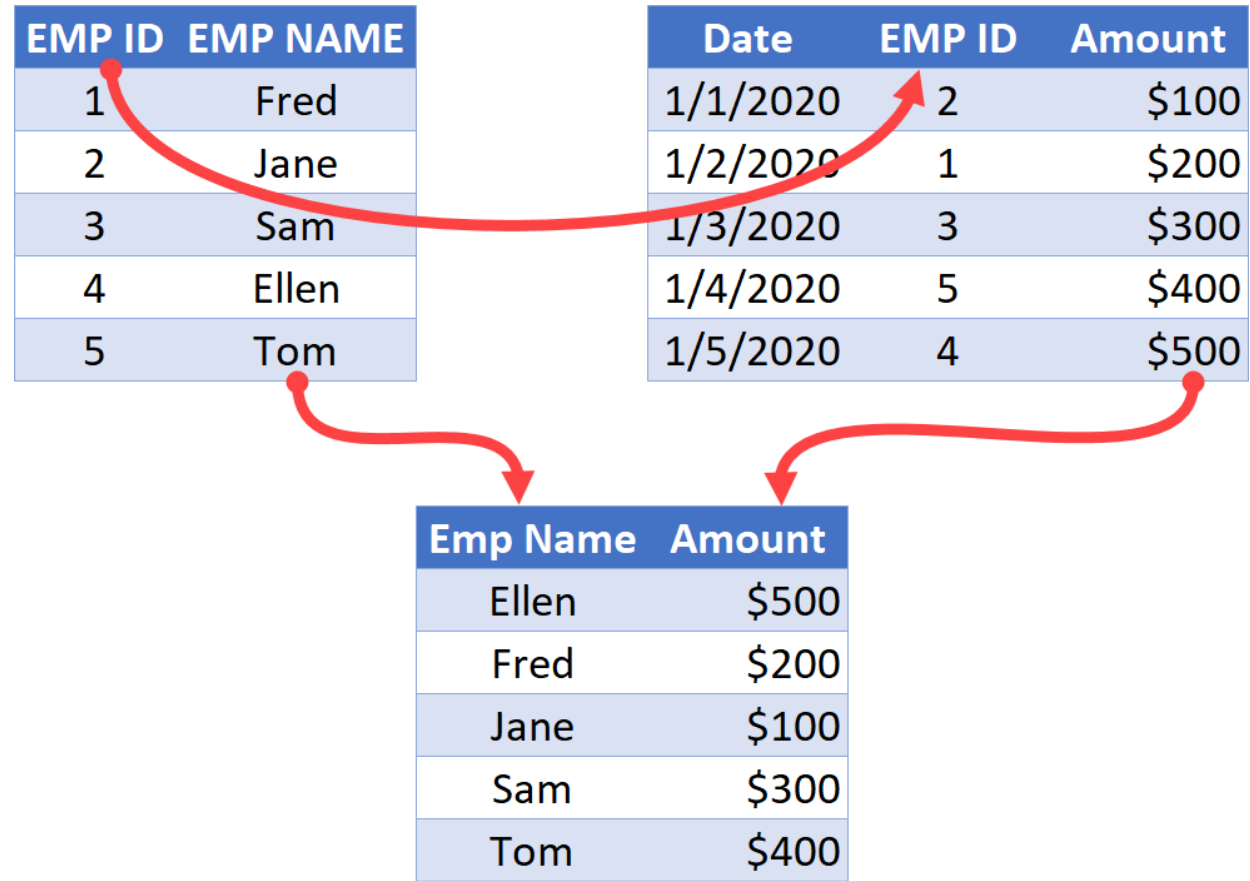
- Each table must contain a common column that has the same data type (*names can be different*)
- An active connection is created between the common columns (*key column*)
- Lookups are performed “live” on an as-needed basis

Creating Relationships *(cont)*



Relationships in Action

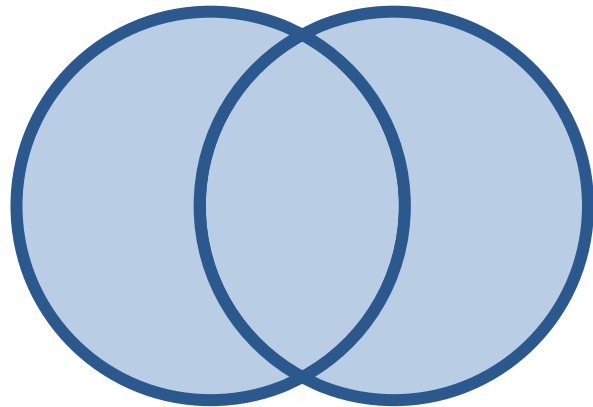
- Allows you to keep the tables separate yet still coordinate a conversation (*lookup*) between the tables
- Requires a “key” column in each table
- Can be 1:1 or 1:Many



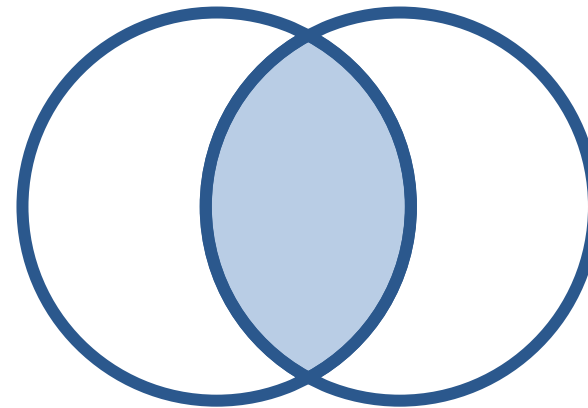


Join Types

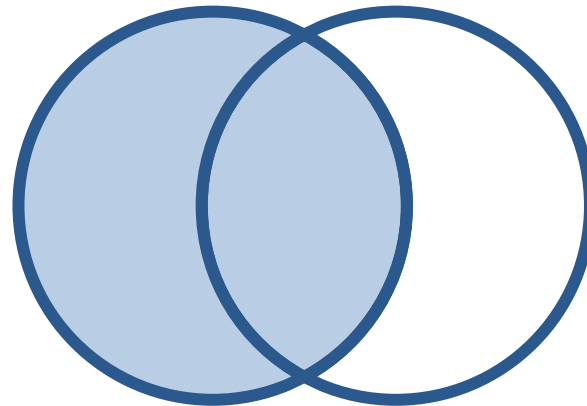
JOIN types



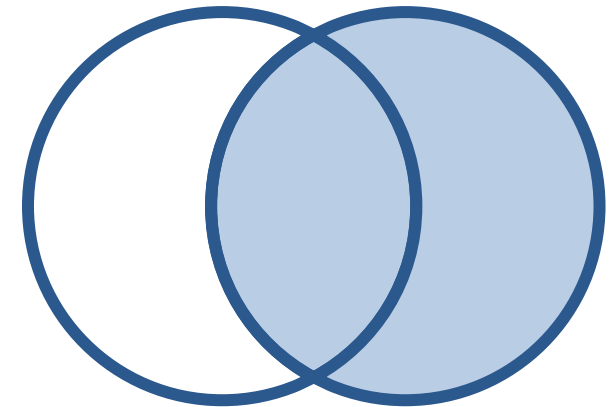
Outer Join



Inner Join



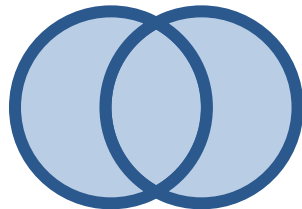
Left Join



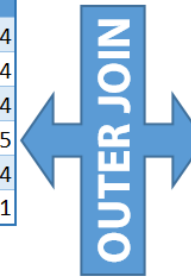
Right Join

JOIN types

- Outer Join
 - All rows will be kept in the result
 - Rows that do not have a corresponding value in the opposing table will receive nulls for fields that are unique to that table



Emp_ID	F_Name	L_Name	Dept	Salary	Status
217	Jane	Henderson	Accounting	75000	4
342	Tim	Biggs	Research	120000	4
477	Bob	Jones	Sales	85000	4
604	Susan	Franklin	Marketing	90000	5
756	Ellen	Frye	Support	55000	4
764	Fred	Smith	Sales	80000	1



Emp_ID	Title	Years
100	Intern	1
217	V.P.	9
342	Technician	2
477	Associate	4
555	Chief	20
764	Manager	12
778	Lead	9
900	Investigator	7

RESULT							
Emp_ID	F_Name	L_Name	Dept	Salary	Status	Title	Years
100	-	-	-	-	-	Intern	1
217	Jane	Henderson	Accounting	75000	4	V.P.	9
342	Tim	Biggs	Research	120000	4	Technician	2
477	Bob	Jones	Sales	85000	4	Associate	4
555	-	-	-	-	-	Chief	20
604	Susan	Franklin	Marketing	90000	5	-	-
756	Ellen	Frye	Support	55000	4	-	-
764	Fred	Smith	Sales	80000	1	Manager	12
778	-	-	-	-	-	Lead	9
900	-	-	-	-	-	Investigator	7

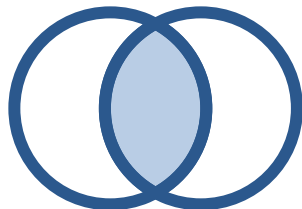
JOIN types

- Inner Join
 - Only rows that can be matched between both tables will be kept in the result

Emp_ID	F_Name	L_Name	Dept	Salary	Status
217	Jane	Henderson	Accounting	75000	4
342	Tim	Biggs	Research	120000	4
477	Bob	Jones	Sales	85000	4
604	Susan	Franklin	Marketing	90000	5
756	Ellen	Frye	Support	55000	4
764	Fred	Smith	Sales	80000	1

INNER JOIN

Emp_ID	Title	Years
100	Intern	1
217	V.P.	9
342	Technician	2
477	Associate	4
555	Chief	20
764	Manager	12
778	Lead	9
900	Investigator	7

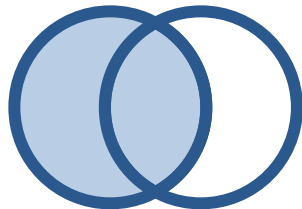


RESULT							
Emp_ID	F_Name	L_Name	Dept	Salary	Status	Title	Years
217	Jane	Henderson	Accounting	75000	4	V.P.	9
342	Tim	Biggs	Research	120000	4	Technician	2
477	Bob	Jones	Sales	85000	4	Associate	4
764	Fred	Smith	Sales	80000	1	Manager	12

JOIN types

- Left Join

- All rows from the first table (left) are retained
- Only rows in the second table that have a corresponding key in the first table are retained
- When no match is found, null values are supplied to unique columns in the second table



Emp_ID	F_Name	L_Name	Dept	Salary	Status
217	Jane	Henderson	Accounting	75000	4
342	Tim	Biggs	Research	120000	4
477	Bob	Jones	Sales	85000	4
604	Susan	Franklin	Marketing	90000	5
764	Fred	Smith	Sales	80000	1

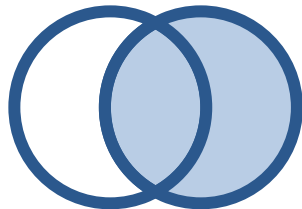
LEFT JOIN

Emp_ID	Title	Years
100	Intern	1
217	V.P.	9
342	Technician	2
477	Associate	4
764	Manager	12
778	Lead	9
900	Investigator	7

RESULT							
Emp_ID	F_Name	L_Name	Dept	Salary	Status	Title	Years
217	Jane	Henderson	Accounting	75000	4	V.P.	9
342	Tim	Biggs	Research	120000	4	Technician	2
477	Bob	Jones	Sales	85000	4	Associate	4
604	Susan	Franklin	Marketing	90000	5	-	-
764	Fred	Smith	Sales	80000	1	Manager	12

JOIN types

- Right Join
 - All rows from the second table are retained
 - Only rows in the first table that have a corresponding key in the second table are retained
 - When no match is found, null values are supplied to unique columns in the first table



Emp_ID	F_Name	L_Name	Dept	Salary	Status
217	Jane	Henderson	Accounting	75000	4
342	Tim	Biggs	Research	120000	4
477	Bob	Jones	Sales	85000	4
604	Susan	Franklin	Marketing	90000	5
764	Fred	Smith	Sales	80000	1

RIGHT JOIN

Emp_ID	Title	Years
100	Intern	1
217	V.P.	9
342	Technician	2
477	Associate	4
764	Manager	12
778	Lead	9
900	Investigator	7

RESULT							
Emp_ID	F_Name	L_Name	Dept	Salary	Status	Title	Years
100	-	-	-	-	-	Intern	1
217	Jane	Henderson	Accounting	75000	4	V.P.	9
342	Tim	Biggs	Research	120000	4	Technician	2
477	Bob	Jones	Sales	85000	4	Associate	4
764	Fred	Smith	Sales	80000	1	Manager	12
778	-	-	-	-	-	Lead	9
900	-	-	-	-	-	Investigator	7



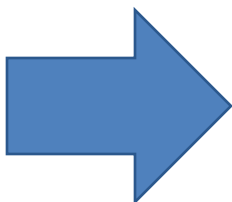


Calendar Tables

Calendar Tables

Date columns are difficult to model

Date	Sales
Monday, January 1, 2018	\$12,418
Tuesday, January 2, 2018	\$26,752
Wednesday, January 3, 2018	\$27,364
Thursday, January 4, 2018	\$9,592
Friday, January 5, 2018	\$2,202
Saturday, January 6, 2018	\$29,934
Sunday, January 7, 2018	\$19,940
Monday, January 8, 2018	\$1,470
Wednesday, January 10, 2018	\$57,216
Sunday, January 14, 2018	\$28,036
Tuesday, January 16, 2018	\$546
Wednesday, January 17, 2018	\$23,428
Thursday, January 18, 2018	\$29,138
Sunday, January 21, 2018	\$11,379
Tuesday, January 23, 2018	\$59,265
Wednesday, January 24, 2018	\$47,560
Thursday, January 25, 2018	\$13,706
Friday, January 26, 2018	\$10,320
Sunday, January 28, 2018	\$8,032
Monday, January 29, 2018	\$28,638
Tuesday, January 30, 2018	\$36,967
Wednesday, January 31, 2018	\$8,190



Month Name	Sales
January	\$1,204,646
February	\$1,301,112
March	\$1,204,598
April	\$1,403,875
May	\$1,487,693
June	\$1,341,493
July	\$1,373,399
August	\$1,367,852
September	\$1,097,312
October	\$1,766,959
November	\$1,062,991
December	\$1,221,133
Total	\$15,833,063

Day Name	Sales
Sunday	\$2,127,919
Monday	\$2,163,326
Tuesday	\$2,604,383
Wednesday	\$2,306,117
Thursday	\$2,380,711
Friday	\$2,051,168
Saturday	\$2,199,439
Total	\$15,833,063

Why use a Calendar Table?



- If you have a date field anywhere in the model, create a Calendar Table
- Needed if you plan to use Time Intelligence functions
 - *DATESQTD()*
 - *DATESINPERIOD()*
 - *DATEADD()*
 - *DATESYTD()*
 - *ENDOFMONTH()*
 - *PREVIOUSMONTH()*
 - *TOTALMTD()*
 - *TOTALYTD()*

Why use a Calendar Table?

- Serves as a lookup table for all date inquiries
- Reduces the amount of repetitive data

Date	A ^B _C Product	A ^B _C State	A ^B _C Supplier	1 ² ₃ Sales	1 ² ₃ Year	1 ² ₃ Month	1 ² ₃ Day	A ^B _C Day Name
6/9/2020	Basketballs	Tennessee	World of Sports	4386	2020	6	9	Tuesday
6/9/2020	Gloves	West Virginia	SportsWorld	14736	2020	6	9	Tuesday
6/9/2020	Basketballs	Nebraska	SportsWorld	2148	2020	6	9	Tuesday
6/9/2020	Golf Balls	Tennessee	Specialty Sports	6678	2020	6	9	Tuesday
6/9/2020	Stepper Machines	New Hampshire	SportsCity	12444	2020	6	9	Tuesday
6/9/2020	Baseballs	Tennessee	Specialty Sports	1647	2020	6	9	Tuesday
6/9/2020	Baseballs	Michigan	Specialty Sports	4528	2020	6	9	Tuesday
6/9/2020	Gloves	Louisiana	Specialty Sports	8164	2020	6	9	Tuesday
6/9/2020	Basketballs	Louisiana	Specialty Sports	8750	2020	6	9	Tuesday
6/9/2020	Tennis Balls	Delaware	Tennis Joint	8572	2020	6	9	Tuesday
6/9/2020	Rowing Machines	North Dakota	World of Sports	10424	2020	6	9	Tuesday
6/9/2020	Exercise Machines	Tennessee	Sports Emporium	9152	2020	6	9	Tuesday
6/9/2020	Golf Balls	Maine	Athlete's World	27748	2020	6	9	Tuesday
6/9/2020	Baseballs	Massachusetts	Sportsman's Den	29624	2020	6	9	Tuesday
6/9/2020	Tennis Balls	North Carolina	Athlete's Dream	24920	2020	6	9	Tuesday
6/9/2020	Tennis Balls	Nebraska	Sportsman's Den	16821	2020	6	9	Tuesday
6/9/2020	Gloves	Delaware	Specialty Sports	11920	2020	6	9	Tuesday

Calendar Table Requirements



- You need a DATE column that is set to a DATE data type
- Dates must be contiguous
 - No skipped days between start date and end date
 - Can cover only the date range of the data, but it's best to cover from Jan 1 of oldest year to Dec 31 of latest year (*or beginning to end of fiscal year*)
- Needs to be marked in Power Pivot as a Date Table

Pro Tip: If you have Date & Time information stored in a single column, consider splitting the dates and times into separate columns. This will decrease cardinality (uniqueness) in those columns and provide for greater compression of the Data Model.

Low-Cardinality – Dates and Time



- If you were to load a 1 year's worth of dates and time information
 - 31,536,000 combinations
 - $365 \text{ days} \times 24 \text{ hours} \times 60 \text{ minutes} \times 60 \text{ seconds}$
- Split the Date and Time into two separate fields
 - 86,765 combinations (*0.28% of the original list*)
 - 365 combinations for dates
 - 86,400 combinations for time
- If time component is not needed, remove it from the model

Low-Cardinality – Big Numbers



- With 100,000,000 numbers there are 100,000,000 combinations
- Split the number into two 10,000 number ranges (Sqrt of 100M)
- Use the `IntegerDivide()` and `Mod()` functions

```
Number.IntegerDivide([BigNumber], 10000)  
Number.Mod([BigNumber], 10000)
```

- First expression performs integer division divided by 10,000
- Second expression performs a modulo operation divided by 10,000

Low-Cardinality – Big Numbers (cont.)



- Both fields have a potential of 10,000 unique values
- 20,000 unique values when combined
- 0.01% the size of the original 100,000,000 combination list
- Restore the original number by

$$(\text{BigNumber1} \times 10000) + \text{BigNumber2}$$

- This process is more processor-intensive

Ways to create a Calendar Table



Many ways to create a Calendar Table

- Excel
- SQL
- Azure
- DAX (*CalendarAuto or Calendar functions*)
- Power Query

Creating a Dynamic Calendar Table



Using DAX

=CALENDAR()

=CALENDARAUTO()

- Creates a table with a single column named “Date”
- Calculated from Jan 1 to Dec 31 based on actual data in the model
(*not from calculated columns*)

=CALENDARAUTO(9)

- Fiscal year ends in September

Beware of CALENDARAUTO()



CALENDARAUTO sounds like the go-to calendar creator, but remember, it looks at ALL date fields in the data model.

Scenario – Imagine a data model with **[Sale Date]** and **[Birth Date]** and the sales are only for the last 3 years.

The calendar table would start at 1/1 of the oldest persons birth year and span to 12/31 of the last sale's year. This has the potential of severely bloating the calendar table.

Creating a Dynamic Calendar Table



Using DAX

= CALENDAR(DATE(2018, 1, 1), DATE(2020, 12, 31))

= CALENDAR(TODAY() - 365, TODAY() + 365)

= CALENDAR(MIN(Sales[OrderDate]), MAX(Sales[OrderDate]))

= CALENDAR(FIRSTDATE(Sales[OrderDate]), LASTDATE(Sales[OrderDate]))

Potential Ideal Date Field



To achieve the full years dates for all sales, consider discovering the oldest and youngest years but hard-coding the start and end days.

Dates =

```
CALENDAR (  
    DATE ( YEAR ( MIN ( Sales[Order Date] ) ), 1, 1 ),  
    DATE ( YEAR ( MAX ( Sales[Order Date] ) ), 12, 31 ))
```

Dates =

```
CALENDAR (  
    DATE ( YEAR ( FIRSTDATE ( Sales[Order Date] ) ), 1, 1 ),  
    DATE ( YEAR ( LASTDATE ( Sales[Order Date] ) ), 12, 31 ))
```





DAX

(Data Analysis Expressions)

What is DAX



- Data Analysis Expressions
- The language of
 - Power Pivot
 - Power BI
 - SQL Server Analysis Services (SSAS)
- Resembles Excel functions (*born with Power Pivot 2010*)

DAX *(as opposed to Excel)*



- Has no concept of rows and columns
- Works with tables and fields
- To locate an item, you must search for it using functions
- Many new functions not available in Excel
- Designed for data models

DAX is a Functional Language



- Everything is written as a function that returns either a:
 - Scalar value (*i.e., single value*)
 - A column of values
 - A table of columns
- Functions have arguments, and arguments can be other functions

MyData =

```
SUMX (  
    FILTER ( VALUES ( 'Date'[Year] ), 'Date'[Year] > 2018 ),  
    IF ( 'Date'[Year] < 2020, [Sales Amount] * .09, [Sales Amount] * .1 ) )
```


Create Readable Code



www.daxformatter.com

```
Calendar=VAR BaseCalendar=CALENDAR(DATE(
2016, 1, 1),DATE(2022, 12, 31 )) RETURN
GENERATE(BaseCalendar, VAR BaseDate=[Date]
VAR YearDate=YEAR(BaseDate) VAR
MonthNumber=MONTH(BaseDate) VAR
MonthName=FORMAT(BaseDate, "mmm") VAR
YearMonthName=FORMAT(BaseDate, "mm yy")
VAR YearMonthNumber=YearDate*12+
MonthNumber-1 RETURN ROW("Day",BaseDate,
"Year",YearDate,"Month Number",MonthNumber,
"Month",MonthName,"Year Month Number",
YearMonthNumber,"Year Month",YearMonthName))
```

```
Calendar =
VAR BaseCalendar =
    CALENDAR ( DATE ( 2016, 1, 1 ), DATE ( 2022, 12, 31 ) )
RETURN
    GENERATE (
        BaseCalendar,
        VAR BaseDate = [Date]
        VAR YearDate =
            YEAR ( BaseDate )
        VAR MonthNumber =
            MONTH ( BaseDate )
        VAR MonthName =
            FORMAT ( BaseDate, "mmm" )
        VAR YearMonthName =
            FORMAT ( BaseDate, "mm yy" )
        VAR YearMonthNumber = YearDate * 12 + MonthNumber - 1
        RETURN
            ROW (
                "Day", BaseDate,
                "Year", YearDate,
                "Month Number", MonthNumber,
                "Month", MonthName,
                "Year Month Number", YearMonthNumber,
                "Year Month", YearMonthName
            )
    )
```

Create Readable Code



www.daxformatter.com

```
=SUMX(FILTER(VALUES('Date'[Year]),'Date'[Year]>2018),IF('Date'[Year]<=2020,[Sales Amount]*.09,[Sales Amount]*.1))
```

```
=  
SUMX (  
    FILTER (  
        VALUES ( 'Date'[Year] ),  
        'Date'[Year] > 2018  
    ),  
    IF (  
        'Date'[Year] <= 2020,  
        [Sales Amount] * .09,  
        [Sales Amount] * .1  
    )  
)
```

Name of Thing = Expression that defines thing

example:

```
Total Revenue =  
SUM ( Sales[SalesAmount] )
```

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

DAX Typing Conventions



- Fully-qualified field names (ex: ***Customer[FullName]*** instead of ***[FullName]***)
- Field name only for Measures (ex: ***[Profit]***)
- Liberal use of spaces (ex: ***SUM (Sales[LineTotal])***)
- Add line break between arguments (see *DAXFORMATTER.com*)



DAX Calculations

DAX Creates



- Calculated Columns
- Calculated Measures (Measures)
- Calculated Tables

Calculated Columns



- Evaluates each row individually, like “helper columns” in Excel
- Cannot access other rows from same table or other tables
- Product[Price]
 - [Price] field name (*explicit / required*)
 - Product (*explicit / optional, but you should always reference*)
 - For the current row (*implicit*)
 - Different result (*calculation*) for each row
- No additional aggregation is needed
- Computed at refresh time
- Stored in model (uses memory ☹)

Calculated Columns *(example)*

$$\text{Profit} = \text{Sales}[\text{SaleAmount}] - \text{Sales}[\text{Cost}]$$

“Sales”
table

Date	Product	Cost	SaleAmount	Profit
1/3/2016	Tennis Balls	1,516	1,912	396
2/17/2016	Footballs	1,709	2,334	625
3/13/2016	Basketballs	1,579	2,004	425
4/25/2016	Golf Balls	1,443	1,553	110
6/7/2016	Rowing Machines	1,090	2,170	1,080
6/11/2016	Baseballs	2,112	3,012	900
9/16/2016	Gloves	1,804	2,784	980
11/15/2016	Golf Balls	2,906	3,356	450
12/21/2016	Stepper Machines	2,224	4,369	2,145
2/15/2017	Gloves	2,140	2,890	750

- Starts with the entire data model
- Applies all relevant filters
 - Pivot Table Rows & Columns
 - Slicers
 - Visuals
 - Functions (e.g., *FILTER*, *CALCULATE*, *ALL*, etc.)
- Performs an expression on the remaining post-filtered data
- Requires an aggregator to process the initial results into a scalar value

Advantage of Measures



- Measures are computed at the aggregate level
- Does not work at row-by-row level
- Results are NOT stored in memory 😊
- Performed when needed (*lazy evaluation*)

Calculated Column (*current row only*)

Profit = Sales[SalePrice] - Sales[Cost]

Measure (*all values in stated fields*)

Profit = SUM (Sales[SalePrice]) - SUM (Sales[Cost])

Measure Naming Conventions



- Avoid placing table name in front of used measure;
helps distinguish a referenced Measure from a referenced field
ex: **[Profit]** instead of **Sales[Profit]**
- Makes it easy to move the measure to a different table
 - Calculated Column → Table[Column]
 - Measure → [Measure]
- Consider storing all Measures in a folder:
Model View → Measure → Properties → Display Folders

Calculated Columns vs Measures



- Calculated Column
 - When you need to slice or filter on a value
- Measure
 - Calculate percentages
 - Calculate ratios
 - Need complex calculations
- Memory vs CPU
 - Calculated Columns consume memory
 - Measures consume CPU

Calculated Table



- Returns a collection of objects rather than a single value
- Used to transform other tables
- Starts with a table, then transforms it (e.g., *FILTER function*)



Aggregation Functions

Aggregation Functions

- Common aggregation functions
 - SUM
 - AVERAGE
 - MAX / MIN
 - COUNT / COUNTDISTINCT
- Work only on numeric columns
- Aggregate only one column
 - ✓ SUM (Orders[Price])
 - ✗ SUM (Orders[Price] * Orders([Quantity])

Aggregation Functions

When used in a Measure:

- Can be used to get the aggregation of any column.

```
[Total Sales] =  
SUM ( tblSales[Sale Amount] )
```

Can be used to create formulas that manipulate multiple columns.

```
[Total Profit] =  
SUM ( tblSales[Sale] ) - SUM ( tblSales[Cost] )
```


ALL (REMOVEFILTERS) Function



- Return all rows from a table, or all rows in a column, ignoring any filters that might have been applied.
- Useful for clearing filters and creating calculations on all the rows in a table.
- Can be used to clear the Filter Context from:

- 1 Column

`ALL(Product[Color])`

- Multiple Columns

`ALL(Product[Color], Customer[City])`



TIP:
*Apply an ALL function to the **Fact Table** to clear all filters from all Dimension Tables.*



Iterator Functions

The 'X' Aggregation Functions



- Iterators: perform row-wise operations before they summarize
 - SUMX
 - AVERAGEX
 - MINX
 - MAXX
 - RANKX
 - COUNTX
 - COUNTAX
 - PRODUCTX
 - VARX.P / VARX.S
 - STDEV.P / STDEV.S
 - CONCATENATEX
 - MEDIANX
 - PERCENTILEX.EXC
 - PERCENTILEX.INC
 - GEOMEANX
- Iterator functions require:
 - A table which to iterate over
 - An expression to process row-by-row
- Performs aggregation on row-level results

Example of SUMX

What is the total profit?

	A	B	C	D	E
1	SalesRep	Date	Product	Cost	Sale
2	Joe Marks	2/8/2022	Golf Balls	\$ 2,409.00	\$ 2,519.00
3	Janice Faraco	4/30/2022	Basketballs	\$ 1,579.00	\$ 2,004.00
4	Ernest Feldgus	5/27/2022	Rowing Machines	\$ 1,090.00	\$ 2,170.00
5	Sandy Brady	6/18/2023	Tennis Balls	\$ 1,406.00	\$ 1,796.00
6	Terry Caracio	7/17/2023	Golf Balls	\$ 2,454.00	\$ 3,034.00
7					

Example of SUMX

What is the total profit?

	A	B	C	D	E	F	
1	SalesRep	Date	Product	Cost	Sale	Profit	
2	Joe Marks	2/8/2022	Golf Balls	\$ 2,409.00	\$ 2,519.00		
3	Janice Faraco	4/30/2022	Basketballs	\$ 1,579.00	\$ 2,004.00		
4	Ernest Feldgus	5/27/2022	Rowing Machines	\$ 1,090.00	\$ 2,170.00		
5	Sandy Brady	6/18/2023	Tennis Balls	\$ 1,406.00	\$ 1,796.00		
6	Terry Caracio	7/17/2023	Golf Balls	\$ 2,454.00	\$ 3,034.00		
7							

Profit =

Example of SUMX

What is the total profit?

	A	B	C	D	E	F
1	SalesRep	Date	Product	Cost	Sale	Profit
2	Joe Marks	2/8/2022	Golf Balls	\$ 2,409.00	\$ 2,519.00	\$ 110.00
3	Janice Faraco	4/30/2022	Basketballs	\$ 1,579.00	\$ 2,004.00	\$ 425.00
4	Ernest Feldgus	5/27/2022	Rowing Machines	\$ 1,090.00	\$ 2,170.00	\$ 1,080.00
5	Sandy Brady	6/18/2023	Tennis Balls	\$ 1,406.00	\$ 1,796.00	\$ 390.00
6	Terry Caracio	7/17/2023	Golf Balls	\$ 2,454.00	\$ 3,034.00	\$ 580.00
7						

$$\text{Profit} = [\text{Sale}] - [\text{Cost}]$$

Example of SUMX

What is the total profit?

	A	B	C	D	E	F
1	SalesRep	Date	Product	Cost	Sale	Profit
2	Joe Marks	2/8/2022	Golf Balls	\$ 2,409.00	\$ 2,519.00	\$ 110.00
3	Janice Faraco	4/30/2022	Basketballs	\$ 1,579.00	\$ 2,004.00	\$ 425.00
4	Ernest Feldgus	5/27/2022	Rowing Machines	\$ 1,090.00	\$ 2,170.00	\$ 1,080.00
5	Sandy Brady	6/18/2023	Tennis Balls	\$ 1,406.00	\$ 1,796.00	\$ 390.00
6	Terry Caracio	7/17/2023	Golf Balls	\$ 2,454.00	\$ 3,034.00	\$ 580.00
7					Total Profit	

$\text{Total Profit} = \text{SUMX}([\text{Sale}] - [\text{Cost}])$

Example of SUMX

What is the total profit?

Data

	A	B	C	D	E	F
1	SalesRep	Date	Product	Cost	Sale	Profit
2	Joe Marks	2/8/2022	Golf Balls	\$ 2,409.00	\$ 2,519.00	\$ 110.00
3	Janice Faraco	4/30/2022	Basketballs	\$ 1,579.00	\$ 2,004.00	\$ 425.00
4	Ernest Feldgus	5/27/2022	Rowing Machines	\$ 1,090.00	\$ 2,170.00	\$ 1,080.00
5	Sandy Brady	6/18/2023	Tennis Balls	\$ 1,406.00	\$ 1,796.00	\$ 390.00
6	Terry Caracio	7/17/2023	Golf Balls	\$ 2,454.00	\$ 3,034.00	\$ 580.00
7					Total Profit	\$ 2,585.00

$$\text{Total Profit} = \text{SUMX}(\text{Data}, [\text{Sale}] - [\text{Cost}])$$

Example of SUMX

What is the total profit?

Data

	A	B	C	D	E	F
1	SalesRep	Date	Product	Cost	Sale	Profit
2	Joe Marks	2/8/2022	Golf Balls	\$ 2,409.00	\$ 2,519.00	\$ 110.00
3	Janice Faraco	4/30/2022	Basketballs	\$ 1,579.00	\$ 2,004.00	\$ 425.00
4	Ernest Feldgus	5/27/2022	Rowing Machines	\$ 1,090.00	\$ 2,170.00	\$ 1,080.00
5	Sandy Brady	6/18/2023	Tennis Balls	\$ 1,406.00	\$ 1,796.00	\$ 390.00
6	Terry Caracio	7/17/2023	Golf Balls	\$ 2,454.00	\$ 3,034.00	\$ 580.00
7					Total Profit	\$ 2,585.00

$$\text{Total Profit} = \text{SUMX}(\text{Data}, \text{Data}[\text{Sale}] - \text{Data}[\text{Cost}])$$

Example of SUMX

Total Profit = SUMX (Data, Data[Sale] - Data[Cost])

For each row in the Data table,
evaluate the expression,
then SUM all the results





Evaluation Context

Two Types of Evaluation Context



- Context – The data that is available to a given calculation
- Row Context
 - Evaluates information contained on a single row.
 - Used when evaluating Calculated Columns.

$$\text{Tax} = \text{Sales}[\text{Subtotal}] * 10\%$$

- Filter Context
 - Evaluates the entire data model via external (*and internal*) filters.
 - Used when evaluating Measures and Calculated Tables

$$\text{Tax} = \text{CALCULATE}(\text{Sales}, [\text{LineTotal}] * 10\%)$$

Filtering Process



1. Start with all data from the entire Data Model.
2. Apply any Slicer filters and/or visual filters.
3. Apply any row/column filters from a Pivot Table.
4. Apply any in-formula filters derived from functions
(*ex: FILTER, ALL, CALCULATE*)

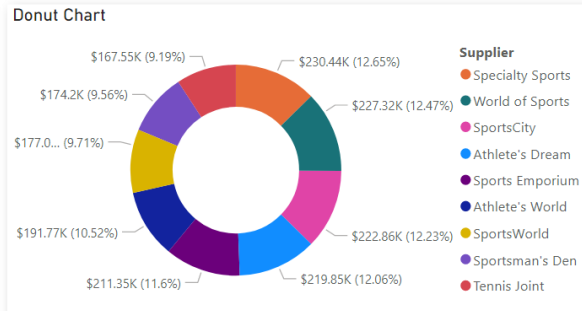
1 - Start with the Entire Data Model



Year
☐ 2016
☐ 2017
☐ 2018
☐ 2019

Product	Central	Northeast	Northwest	Southeast	Southwest	Total
Baseballs	\$66,140	\$81,822		\$81,871		\$229,833
Basketballs					\$158,460	\$158,460
Exercise Machines			\$59,450			\$59,450
Footballs	\$81,044		\$69,743			\$150,787
Gloves	\$74,461	\$204,865			\$97,204	\$376,530
Golf Balls		\$159,249			\$135,428	\$294,677
Rowing Machines	\$53,210			\$67,901	\$75,588	\$196,699
Stepper Machines		\$65,612	\$134,118			\$199,730
Tennis Balls		\$81,400		\$74,799		\$156,199
Total	\$274,855	\$592,948	\$263,311	\$224,571	\$466,680	\$1,822,365

Month
☐ January
☐ February
☐ March
☐ April
☐ May
☐ June
☐ July
☐ August
☐ September
☐ October
☐ November
☐ December



\$1,822,365

Sales

Sales = \$1,822,365

Sales Representative	Date	Year	Month	Product	State	Region	Supplier	Cost	Sales
Aaron Baker	1/25/2017	2017	January	Gloves	Nebraska	Southwest	Athlete's Dream	\$7,500	\$8,865
Adolph Marx	4/13/2019	2019	April	Stepper Machines	Kentucky	Northwest	SportsWorld	\$1,593	\$2,698
Albert George	8/20/2017	2017	August	Footballs	New York	Northwest	World of Sports	\$3,387	\$4,081
Alexandra Zuck	9/18/2018	2018	September	Exercise Machines	Washington	Northwest	Athlete's World	\$150	\$196
Alfredo Coccozza	5/9/2018	2018	May	Baseballs	North Dakota	Northeast	Sportsman's Den	\$2,203	\$4,354
Alfredo James	2/12/2019	2019	February	Rowing Machines	Ohio	Southeast	Sportsman's Den	\$7,442	\$8,663
Alice Abramas	4/11/2018	2018	April	Golf Balls	Vermont	Northeast	Athlete's World	\$696	\$2,465
Alice Abramas	6/9/2017	2017	June	Gloves	Alabama	Northeast	SportsWorld	\$4,318	\$4,768
Alice Abramas	3/1/2017	2017	March	Gloves	Arizona	Northeast	Tennis Joint	\$1,116	\$1,766
Alicia Christian	1/29/2017	2017	January	Baseballs	Delaware	Southeast	Athlete's Dream	\$3,406	\$4,256
Allen Konigsberg	3/11/2019	2019	March	Exercise Machines	Connecticut	Northwest	Specialty Sports	\$4,119	\$4,649
Alphonso D'Abruzzo	9/22/2017	2017	September	Gloves	Alaska	Southwest	Sports Emporium	\$6,200	\$6,991
Andrew Warhola	3/17/2017	2017	March	Golf Balls	New York	Northeast	Athlete's Dream	\$4,441	\$6,236
Angeline Brown	10/20/2016	2016	October	Baseballs	Pennsylvania	Southeast	Sportsman's Den	\$3,959	\$5,316
Annie Mae	7/20/2017	2017	July	Gloves	South Dakota	Northeast	Sportsman's Den	\$1,250	\$1,698
Anthony Dominick	7/19/2018	2018	July	Stepper Machines	South Carolina	Northwest	SportsCity	\$5,364	\$7,586

2 - Filter for Years

Year

☐ 2016

☒ 2017

☒ 2018

☐ 2019

Product	Central	Northeast	Northwest	Southeast	Southwest	Total
Baseballs	\$41,099	\$61,027		\$49,630		\$151,756
Basketballs					\$107,020	\$107,020
Exercise Machines			\$27,942			\$27,942
Footballs	\$58,031		\$45,783			\$103,814
Gloves	\$52,647	\$137,220			\$64,953	\$254,820
Golf Balls		\$106,702				\$106,702
Rowing Machines	\$25,562			\$46,704	\$45,027	\$117,293
Stepper Machines		\$33,760	\$81,613			\$115,373
Tennis Balls		\$54,636		\$43,271		\$97,907
Total	\$177,339	\$393,345	\$155,338	\$139,605	\$333,845	\$1,199,472

\$1,199,472

Sales

Sales = \$1,199,472

Month

☐ January

☐ February

☐ March

☐ April

☐ May

☐ June

☐ July

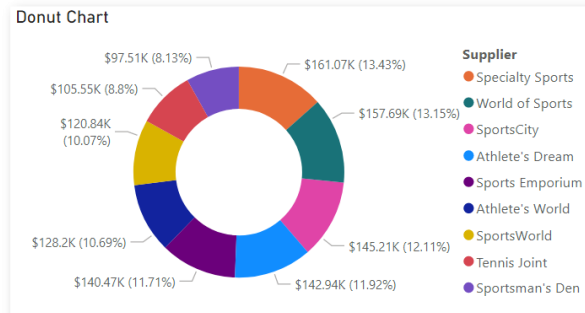
☐ August

☐ September

☐ October

☐ November

☐ December



[Year] = 2017 & 2018

Sales Representative	Date	Year	Month	Product	State	Region	Supplier	Cost	Sales
Aaron Baker	1/25/2017	2017	January	Gloves	Nebraska	Southwest	Athlete's Dream	\$7,500	\$8,865
Albert George	8/20/2017	2017	August	Footballs	New York	Northwest	World of Sports	\$3,387	\$4,081
Alexandra Zuck	9/18/2018	2018	September	Exercise Machines	Washington	Northwest	Athlete's World	\$150	\$196
Alfredo Cocozza	5/9/2018	2018	May	Baseballs	North Dakota	Northeast	Sportsman's Den	\$2,203	\$4,354
Alice Abramas	4/11/2018	2018	April	Golf Balls	Vermont	Northeast	Athlete's World	\$696	\$2,465
Alice Abramas	6/9/2017	2017	June	Gloves	Alabama	Northeast	SportsWorld	\$4,318	\$4,768
Alice Abramas	3/1/2017	2017	March	Gloves	Arizona	Northeast	SportsWorld	\$1,116	\$1,766
Alicia Christian	1/29/2017	2017	January	Baseballs	Delaware	Southeast	Athlete's Dream	\$3,406	\$4,256
Alphonso D'Abruzzo	9/22/2017	2017	September	Gloves	Alaska	Southwest	Sports Emporium	\$6,200	\$6,991
Andrew Warhola	3/17/2017	2017	March	Golf Balls	New York	Northeast	Athlete's Dream	\$4,441	\$6,236
Annie Mae	7/20/2017	2017	July	Gloves	South Dakota	Northeast	Sportsman's Den	\$1,250	\$1,698
Anthony Dominick	7/19/2018	2018	July	Stepper Machines	South Carolina	Northwest	SportsCity	\$5,364	\$7,586
Anthony Papaleo	6/20/2017	2017	June	Stepper Machines	Washington	Northwest	Tennis Joint	\$275	\$421
Aristotelis Harris	4/29/2017	2017	April	Golf Balls	Idaho	Southwest	Athlete's World	\$392	\$2,246
Asa Yoelson	1/13/2018	2018	January	Stepper Machines	Minnesota	Northeast	Athlete's Dream	\$4,367	\$5,151
Audrey Cotter	11/24/2017	2017	November	Stepper Machines	Washington	Northeast	SportsCity	\$4,609	\$6,928

3 - Filter for Months

Year

- ☐ 2016
- ☒ 2017
- ☒ 2018
- ☐ 2019

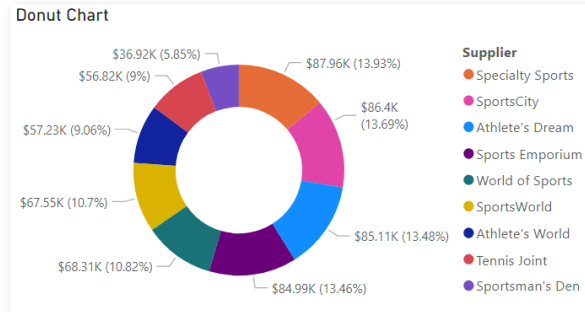
Product	Central	Northeast	Northwest	Southeast	Southwest	Total
Baseballs	\$25,737	\$15,409		\$18,895		\$60,041
Basketballs					\$58,320	\$58,320
Exercise Machines			\$15,783			\$15,783
Footballs	\$31,227		\$12,075			\$43,302
Gloves	\$28,599	\$81,513			\$39,643	\$149,755
Golf Balls		\$44,498			\$76,770	\$121,268
Rowing Machines	\$7,070			\$36,512	\$26,514	\$70,096
Stepper Machines		\$20,207	\$46,118			\$66,325
Tennis Balls		\$30,001		\$16,402		\$46,403
Total	\$92,633	\$191,628	\$73,976	\$71,809	\$201,247	\$631,293

\$631,293

Sales

Sales = \$631,293

- Month
- ☒ January
 - ☒ February
 - ☒ March
 - ☒ April
 - ☒ May
 - ☒ June
 - ☐ July
 - ☐ August
 - ☐ September
 - ☐ October
 - ☐ November
 - ☐ December



[Year] = 2017 & 2018
[Month] = "January" thru "June"

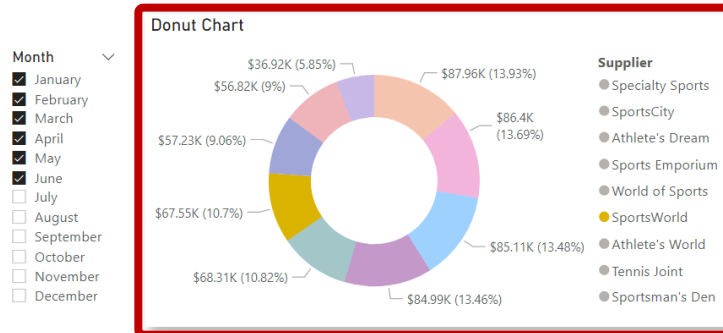
Sales Representative	Date	Year	Month	Product	State	Region	Supplier	Cost	Sales
Aaron Baker	1/25/2017	2017	January	Gloves	Nebraska	Southwest	Athlete's Dream	\$7,500	\$8,865
Alfredo Coccozza	5/9/2018	2018	May	Baseballs	North Dakota	Northeast	Sportsman's Den	\$2,203	\$4,354
Alice Abramas	4/11/2018	2018	April	Golf Balls	Vermont	Northeast	Athlete's World	\$696	\$2,465
Alice Abramas	6/9/2017	2017	June	Gloves	Alabama	Northeast	SportsWorld	\$4,318	\$4,768
Alice Abramas	3/1/2017	2017	March	Gloves	Arizona	Northeast	SportsWorld	\$1,116	\$1,766
Alicia Christian	1/29/2017	2017	January	Baseballs	Delaware	Southeast	Athlete's Dream	\$3,406	\$4,256
Andrew Warhola	3/17/2017	2017	March	Golf Balls	New York	Northeast	Athlete's Dream	\$4,441	\$6,236
Anthony Papaleo	6/20/2017	2017	June	Stepper Machines	Washington	Northwest	Tennis Joint	\$275	\$421
Aristotelis Harris	4/29/2017	2017	April	Golf Balls	Idaho	Southwest	Athlete's World	\$392	\$2,246
Asa Yoelson	1/13/2018	2018	January	Stepper Machines	Minnesota	Northeast	Athlete's Dream	\$4,367	\$5,151
Benjamin Geza	6/5/2018	2018	June	Footballs	Florida	Central	Athlete's World	\$4,026	\$6,436
Betty Joan	4/13/2017	2017	April	Gloves	Iowa	Northeast	SportsWorld	\$90	\$1,784
Burton Leon	2/27/2018	2018	February	Stepper Machines	Indiana	Northwest	Athlete's World	\$5,381	\$6,436
Byron Barr	1/11/2017	2017	January	Gloves	Kentucky	Northeast	SportsWorld	\$2,608	\$4,286
Camille Javal	1/20/2018	2018	January	Basketballs	Idaho	Southwest	Tennis Joint	\$935	\$2,892
Carlos Irwin	2/15/2018	2018	February	Tennis Balls	Georgia	Southeast	World of Sports	\$1,718	\$2,247

4 - Filter for Supplier

Year

- ☐ 2016
- ☒ 2017
- ☒ 2018
- ☐ 2019

Product	Central	Northeast	Northwest	Southwest	Total
Basketballs				\$8,893	\$8,893
Exercise Machines			\$7,728		\$7,728
Footballs			\$3,824		\$3,824
Gloves	\$5,363	\$18,374			\$23,737
Golf Balls		\$936		\$5,477	\$6,413
Rowing Machines				\$4,272	\$4,272
Stepper Machines			\$7,884		\$7,884
Tennis Balls		\$4,803			\$4,803
Total	\$5,363	\$24,113	\$19,436	\$18,642	\$67,554



\$67,554

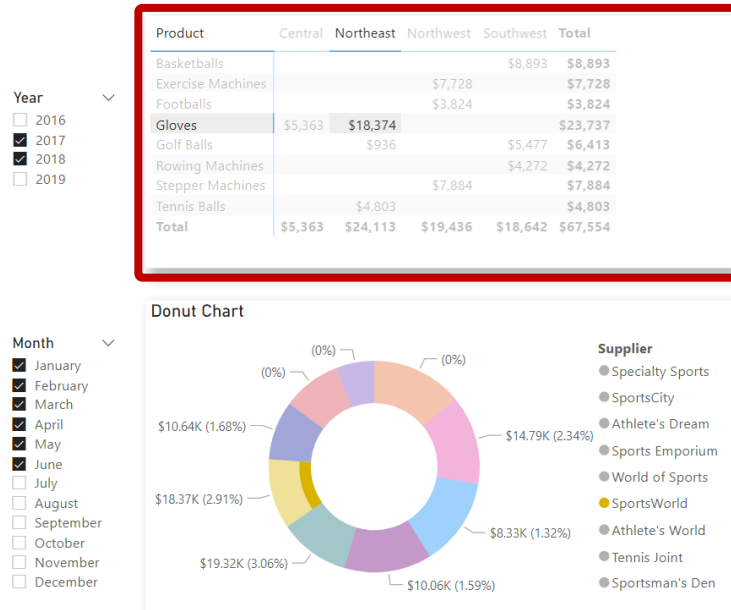
Sales

Sales = \$67,554

[Year] = 2017 & 2018
 [Month] = "January" thru "June"
 [Supplier] = "SportsWorld"

Sales Representative	Date	Year	Month	Product	State	Region	Supplier	Cost	Sales
Alice Abrams	6/9/2017	2017	June	Gloves	Alabama	Northeast	SportsWorld	\$4,318	\$4,768
Alice Abrams	3/1/2017	2017	March	Gloves	Arizona	Northeast	SportsWorld	\$1,116	\$1,766
Betty Joan	4/13/2017	2017	April	Gloves	Iowa	Northeast	SportsWorld	\$90	\$1,784
Byron Barr	1/11/2017	2017	January	Gloves	Kentucky	Northeast	SportsWorld	\$2,608	\$4,286
Delloreese Patricia	1/19/2018	2018	January	Tennis Balls	Utah	Northeast	SportsWorld	\$3,982	\$4,803
Eric Marlon	6/2/2018	2018	June	Footballs	Nevada	Northwest	SportsWorld	\$1,667	\$3,824
Ethel Zimmerman	2/13/2017	2017	February	Golf Balls	Maine	Northeast	SportsWorld	\$738	\$936
Francis Thomas	3/25/2018	2018	March	Golf Balls	Montana	Southwest	SportsWorld	\$4,226	\$5,477
Harold Lipshitz	5/17/2018	2018	May	Gloves	Minnesota	Central	SportsWorld	\$4,101	\$5,363
Jerome Silberman	2/18/2018	2018	February	Rowing Machines	Tennessee	Southwest	SportsWorld	\$3,712	\$4,272
John Carpenter	6/17/2017	2017	June	Gloves	New Hampshire	Northeast	SportsWorld	\$3,625	\$5,770
Maurice Micklewhite	1/24/2018	2018	January	Exercise Machines	Kansas	Northwest	SportsWorld	\$401	\$451
Susan Kerr	4/17/2017	2017	April	Exercise Machines	Illinois	Northwest	SportsWorld	\$7,167	\$7,277
Uma Karuna	4/25/2017	2017	April	Stepper Machines	New Mexico	Northwest	SportsWorld	\$7,690	\$7,884
Virginia McMath	3/31/2018	2018	March	Basketballs	Georgia	Southwest	SportsWorld	\$6,543	\$8,893

5 - Filter for Product & Region



\$18,374

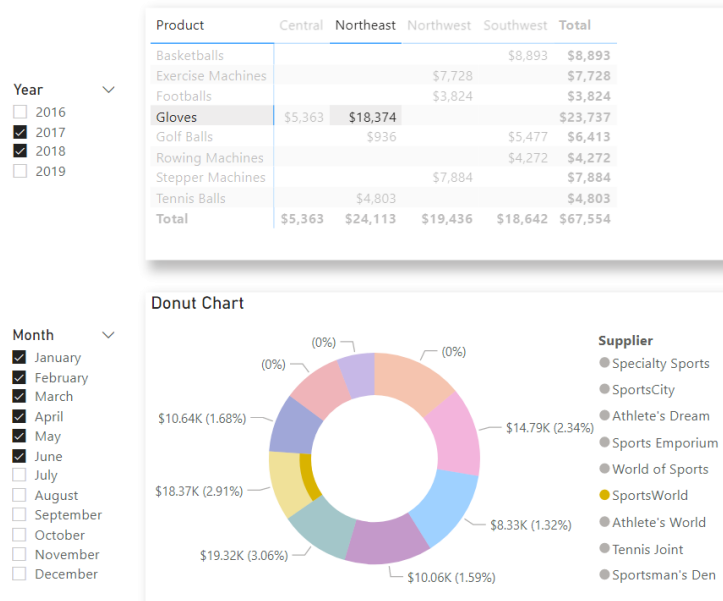
Sales

Sales = \$18,374

Sales Representative	Date	Year	Month	Product	State	Region	Supplier	Cost	Sales
Alice Abramas	6/9/2017	2017	June	Gloves	Alabama	Northeast	SportsWorld	\$4,318	\$4,768
Alice Abramas	3/1/2017	2017	March	Gloves	Arizona	Northeast	SportsWorld	\$1,116	\$1,766
Betty Joan	4/13/2017	2017	April	Gloves	Iowa	Northeast	SportsWorld	\$90	\$1,784
Byron Barr	1/11/2017	2017	January	Gloves	Kentucky	Northeast	SportsWorld	\$2,608	\$4,286
John Carpenter	6/17/2017	2017	June	Gloves	New Hampshire	Northeast	SportsWorld	\$3,625	\$5,770

[Year] = 2017 & 2018
 [Month] = "January" thru "June"
 [Supplier] = "SportsWorld"
 [Product] = "Gloves"
 [Region] = "Northeast"

6 - Filter for Sales Representative



\$6,534

AA Sales

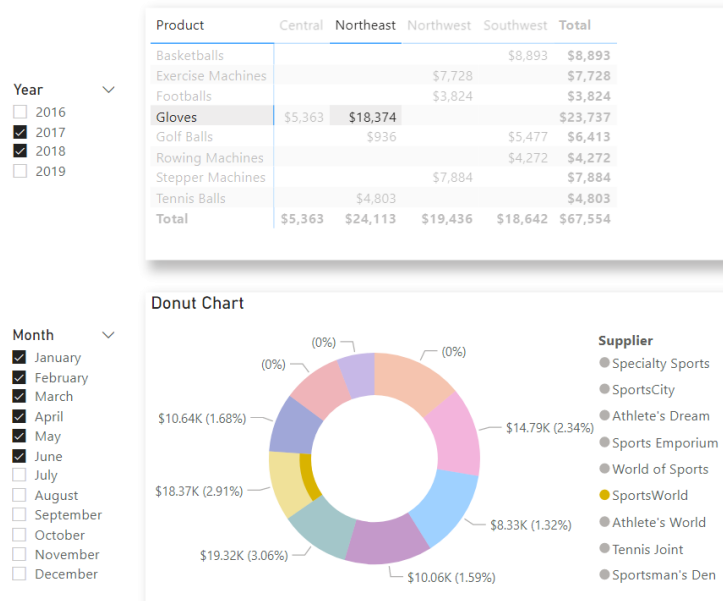
Sales = \$6,534

Sales Representative	Date	Year	Month	Product	State	Region	Supplier	Cost	Sales
Alice Abrams	6/9/2017	2017	June	Gloves	Alabama	Northeast	SportsWorld	\$4,318	\$4,768
Alice Abrams	3/1/2017	2017	March	Gloves	Arizona	Northeast	SportsWorld	\$1,116	\$1,766

[Year] = 2017 & 2018
 [Month] = "January" thru "June"
 [Supplier] = "SportsWorld"
 [Product] = "Gloves"
 [Region] = "Northeast"
 [Sales Representative] = "Alice Abrams"

AA Sales =
 CALCULATE (
 SUM (tblSales[Sales]),
 tblSales[Sales Representative] = "Alice Abrams"
)

6 - Perform the Expression



\$6,534

AA Sales

Sales = \$6,534

Sales Representative	Date	Year	Month	Product	State	Region	Supplier	Cost	Sales
Alice Abrams	6/9/2017	2017	June	Gloves	Alabama	Northeast	SportsWorld	\$4,318	\$4,768
Alice Abrams	3/1/2017	2017	March	Gloves	Arizona	Northeast	SportsWorld	\$1,116	\$1,766

AA Sales =

CALCULATE (

SUM (tblSales[Sales])

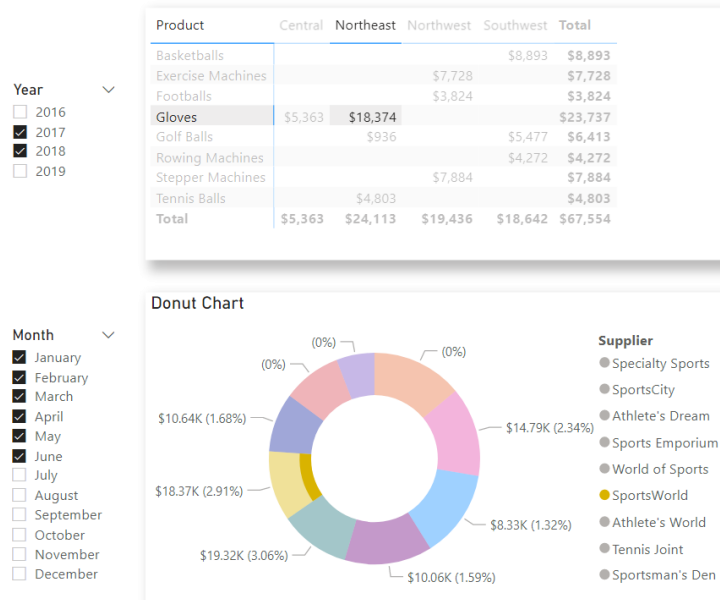
tblSales[Sales Representative] = "Alice Abrams"

)

7 - Performed as a Single Expression



Sales = \$6,534



Sales Representative	Date	Year	Month	Product	State	Region	Supplier	Cost	Sales
Alice Abrams	6/9/2017	2017	June	Gloves	Alabama	Northeast	SportsWorld	\$4,318	\$4,768
Alice Abrams	3/1/2017	2017	March	Gloves	Arizona	Northeast	SportsWorld	\$1,116	\$1,766

\$6,534

AA Sales

AA Sales =

```
CALCULATE (
    SUM ( tblSales[Sales] ),
    tblSales[Year] IN { 2017, 2018 },
    tblSales[Month] IN { "January", "February", "March", "April", "May", "June" },
    tblSales[Product] = "Gloves",
    tblSales[Region] = "Northeast",
    tblSales[Supplier] = "SportsWorld",
    tblSales[Sales Representative] = "Alice Abrams"
)
```



CALCULATE Function

The CALCULATE Function



- The most important, most powerful DAX function
- The only DAX function that can completely change the filters coming from the Filter Context
- Can overwrite or supplement existing filters

`CALCULATE(<expression> [, <filter1> [, <filter2> [, ...]]])`

`=CALCULATE([Total Sales], Product[Category] = “Bikes”)`

CALCULATE's Processing Order



1. Visual filters are applied (*ex: pivot table rows/columns, slicers, etc.*)
2. Filters from CALCULATE are added
3. Conflicting filters between visual and CALCULATE are replaced with CALCULATE's filters
4. Expression is calculated

Syntax Sugar

- The act of reducing a complex formula into a simpler formula
- Hides the complexity from the user

```
=  
CALCULATE (  
    [Total Sales],  
    FILTER ( ALL ( Product[Category] ), Product[Category] = "Bikes" )  
)
```

```
=  
CALCULATE ( [Total Sales], Product[Category] = "Bikes" )
```

```
=  
[Total Sales] ( Product[Category] = "Bikes" )
```





Date/Time Functions

Date and Time Functions



- Used to process date/time data in various ways:
 - Convert to and from strings
 - Compute offsets
 - Switch timezones
- Simplifies the process of working with dates and the peculiarities associated with calculating and formatting them.

Time Intelligence Functions



- Used to make working with common date-based measures easier.
 - Year to Date
 - Quarter to Date
 - Month to Date
 - Same Period Last Year
 - Previous Period
- Are extremely mission-specific.
- Flexibility is sacrificed for ease of use.

Examples of Time Intelligence



```
Sales YTD = CALCULATE(  
    [Sales Total],  
    DATESYTD(Dates[Date]) )
```

To account for fiscal years, define the end of the FY

```
Sales YTD = CALCULATE(  
    [Sales Total],  
    DATESYTD(Dates[Date], "06-30") )
```

Examples of Time Intelligence



Sales PY = CALCULATE([Sales Total],
SAMEPERIODLASTYEAR(Dates[Date]))

Sales PM = CALCULATE([Sales Total],
PREVIOUSMONTH(Dates[Date]))

Sales PY = CALCULATE([Sales Total],
DATEADD(Dates[Date], -1, YEAR))

Sales PM = CALCULATE([Sales Total],
DATEADD(Dates[Date], -1, MONTH))

Examples of Time Intelligence



Calculate the running total for [Sales Total] without ever resetting

```
RT_Sales =  
CALCULATE (  
    SUM ( [Sales Total] ),  
    FILTER ( ALL ( Dates[Date] ),  
            Dates[Date] <= MAX ( Dates[Date] )  
    )  
)
```

Date Tables



Required to have a Date Table when using time intelligence functions.

- 1 row per day
- Days must be consecutive (*no skipped days*)
- Must have a column defined as a DATE data type
- Must be marked as a Date Table

