

# Designing a Data Model with Power BI Desktop



# Agenda

- Creating Table Relationships
  - Creating Calculated Columns and Measure
  - Creating Tables using DAX Expressions
  - Configuring Fields for Geographic Mapping



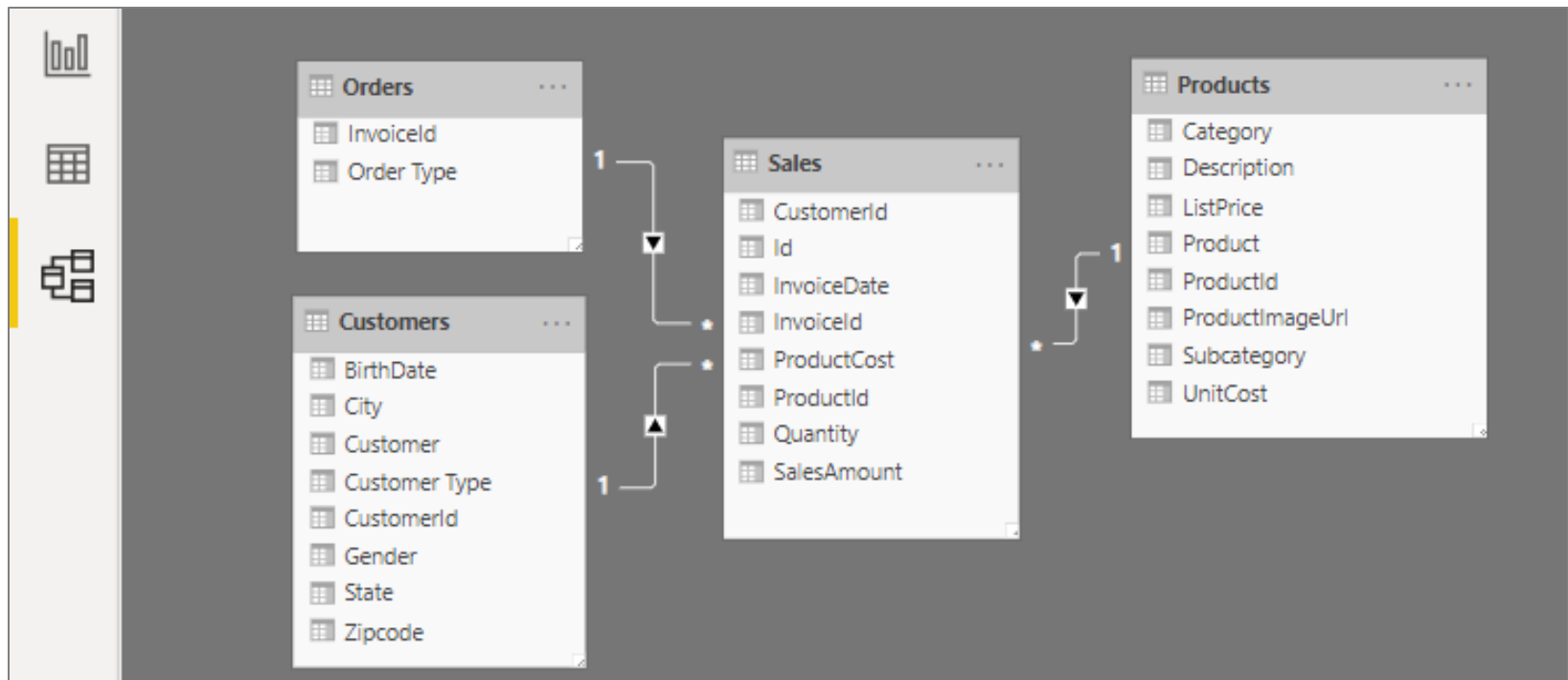
# Data Modeling with Power BI Desktop

- Steps to create a data model with Power Pivot
  - Create relationships between tables
  - Modify native columns (e.g. set formatting and data category)
  - Create calculated columns
  - Create measures
  - Create dimensional hierarchies
  - Add Calendar table(s)



# Table Relationships

- Tables in data model associated with relationships
  - Relationships based on single columns
  - Tabular model supports [1-to-1] and [1-to-many] relationships



# Relationship Properties

- Cardinality

Cardinality

Many to One (\*:1)

Many to One (\*:1)

One to One (1:1)

One to Many (1:\*)

- Cross filter direction

Cross filter direction

Both

Single

Both

Edit Relationship

Select tables and columns that relate to one another.

Sales

Id	Quantity	SalesAmount	InvoiceId	ProductId	CustomerId	PurchaseDate	ProductCost
2899	100	100	1457	14	888	Thursday, June 21, 2012	\$8
3824	100	100	1901	14	1137	Saturday, July 21, 2012	\$8
3968	100	100	1969	14	1173	Wednesday, July 25, 2012	\$8

Customers

CustomerId	City	State	ZipCode	Gender	BirthDate	Customer	CustomerType
55	San Jose	CA	95110	Female	Thursday, March 10, 1949	Jewell Ryan	Repeat Customer
73	San Jose	CA	95123	Male	Thursday, May 9, 1985	Granville Perry	Repeat Customer
74	San Jose	CA	95122	Female	Tuesday, June 19, 1979	Sheri Mercado	Repeat Customer

Cardinality

Many to One (\*:1)

Cross filter direction

Both

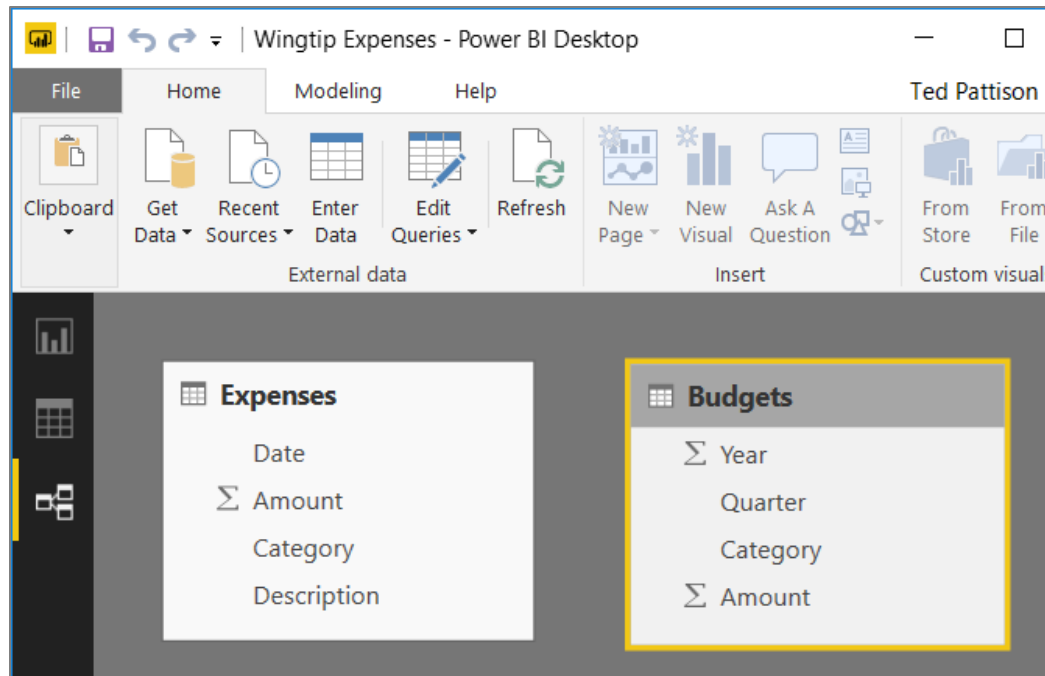
☒ Make this relationship active

OK Cancel



# How Do You Create a Relationship Here?

- Two tables don't have fields to create relationship
  - The solution is to create two new calculated columns



# Creating Composite Key Fields

- Create composite key column in Budgets

The screenshot shows the Power BI interface with a table view of Budgets. The formula bar at the top displays the formula: `Budget Key = [Year] & "-" & [Quarter] & "-" & [Category]`. The table has five columns: Year, Quarter, Category, Amount, and Budget Key. The Budget Key column contains values like "2017-Q1-Marketing". On the right, the FIELDS pane shows the Budgets table with fields Amount, Budget Key (highlighted), and Category.

Year	Quarter	Category	Amount	Budget Key
2017	Q1	Marketing	\$5,000	2017-Q1-Marketing
2017	Q1	Office Supplies	\$8,000	2017-Q1-Office Supplies
2017	Q1	Operations	\$8,000	2017-Q1-Operations
2017	Q1	Research & Development	\$5,000	2017-Q1-Research & Development
2017	Q2	Marketing	\$6,000	2017-Q2-Marketing
2017	Q2	Office Supplies	\$4,000	2017-Q2-Office Supplies
2017	Q2	Operations	\$7,000	2017-Q2-Operations

- Create composite key column in Expenses

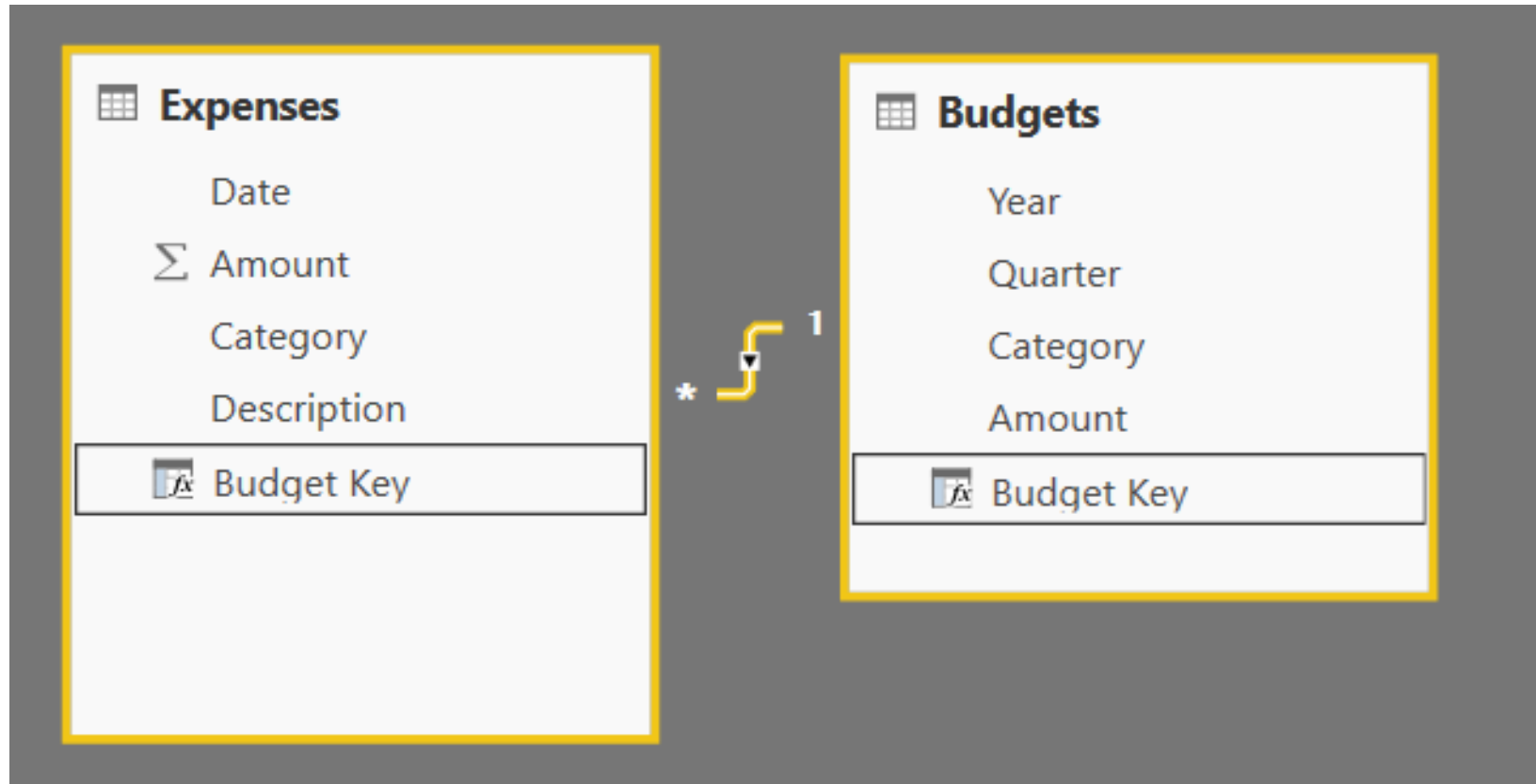
The screenshot shows the Power BI interface with a table view of Expenses. The formula bar at the top displays the formula: `Budget Key = VAR BudgetYear = YEAR([Date]) VAR BudgetMonth = "Q" & FORMAT([Date], "q") RETURN BudgetYear & "-" & BudgetMonth & "-" & [Category]`. The table has five columns: Date, Amount, Category, Description, and Budget Key. The Budget Key column contains values like "2017-Q2-Operations". On the right, the FIELDS pane shows the Expenses table with fields Amount, Budget Key (highlighted), and Category.

Date	Amount	Category	Description	Budget Key
Sunday, April 2, 2017	\$925	Operations	Verizon - Telephone and Internet	2017-Q2-Operations
Monday, April 3, 2017	\$142	Office Supplies	Postage Stamps	2017-Q2-Office Supplies
Wednesday, April 5, 2017	\$294	Operations	Electricity Bill	2017-Q2-Operations
Wednesday, April 5, 2017	\$120.25	Office Supplies	Coffee Supplies	2017-Q2-Office Supplies
Thursday, April 13, 2017	\$1,200	Operations	Cleaning Service	2017-Q2-Operations





# Create Relationship Using Composite Keys





# Agenda

- ✓ Creating Table Relationships
- Creating Calculated Columns and Measure
  - Creating Tables using DAX Expressions
  - Configuring Fields for Geographic Mapping



# Formatting Columns

- Each column has its own formatting properties
  - Formatting propagates to reports and visuals
  - Visuals automatically display values using format properties

Wingtip Sales Analysis - Power BI Desktop

File Home Modeling Help

Manage Relationships Relationships

New Measure Calculations

New Column

New Table

New Parameter What If

Sort by Column Sort

Data type: Fixed decimal number

Format: \$ English (United States)

Home Table:

Property: Uncategorized

Default Summarization: Sum

Formatting Properties

Id	Quantity	SalesAmount	InvoiceId	ProductId	InvoiceDate	CustomerId
95	9	\$179.55	46	6	Saturday, February 4, 2012	46
96	9	\$179.55	47	6	Saturday, February 4, 2012	47



# Working with DAX

- DAX is the language used to create data models
  - DAX stands for "Data Analysis Expression Language"
- DAX expressions are similar to Excel formulas
  - They always start with an equal sign (=)
  - DAX provides many built-in functions similar to Excel
- DAX Expressions are unlike Excel formulas...
  - DAX expressions cannot reference cells (e.g. A1 or C4)
  - Instead DAX expressions reference columns and tables

```
=SUM('Sales' [SalesAmount])
```



# Writing DAX Expressions

- Some DAX expressions are simple

```
Sales Revenue = Sum(Sales[SalesAmount])
```

- Some DAX expressions are far more complex

```
Sales Growth PM = IF(
  ( ISFILTERED(Calendar[Month]) && ISFILTERED(Calendar[Date]) = FALSE() ),
  DIVIDE(
    SUM(Sales[SalesAmount]) -
    CALCULATE(
      SUM(Sales[SalesAmount]),
      PREVIOUSMONTH(Calendar[Date])
    ),
    CALCULATE(
      SUM(Sales[SalesAmount]),
      PREVIOUSMONTH(Calendar[Date])
    )
  ),
  BLANK()
)
```



# Creating Variables in DAX Expressions

- Variables can be added at start of expression
  - Use **VAR** keyword once for each variable
  - Use **RETURN** keyword to return expression value

```
Budget Key =  
    VAR BudgetYear = YEAR([Date])  
    VAR BudgetMonth = "Q" & FORMAT([Date], "q")  
    RETURN  
    BudgetYear & "-" & BudgetMonth & "-" & [Category]
```



# Calculated Columns vs Measures

- Calculated Columns (aka Columns)
  - Evaluated based on context of a single row
  - Evaluated when data is loaded into memory

`Column1 = <DAX expression>`

- Measures
  - Evaluated at query time based on current filter context
  - Commonly used for aggregations (e.g. SUM, AVG, etc.)
  - Used more frequently than calculated columns

`Measure1 = <DAX expression>`



# When to Create Calculated Columns

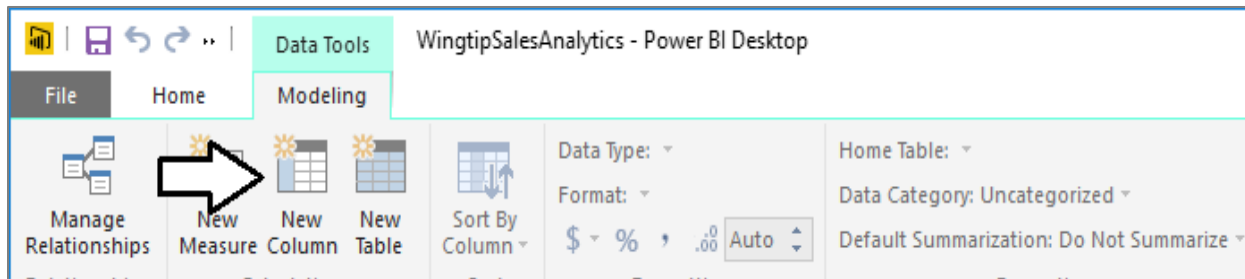
- Measures often better choice than calculate columns
  - Don't create calculated column when you need a measure
  - Prefer to create calculated columns only in specific scenarios
- When should you create calculated columns?
  - To create headers for row labels or column labels
  - To place calculated results in a slicer for filtering
  - Define an expression strictly bound to current row
  - Categories text or numbers (e.g. customer age groups)





# Creating Calculated Columns

- Edited in formula bar of Power Pivot data view
  - Start with name and then equals (=) sign
  - Enter a valid DAX expression
  - Clicking on column adds it into expression



1 Age = Floor((TODAY()-Customers[BirthDate])/365, 1)									
CustomerId	City	State	Zipcode	Gender	BirthDate	Customer	Customer Type	Age	
760	San Jose	CA	95133	Female	3/16/1968	Lucile Blake	One-time Customer	51	
881	San Jose	CA	95133	Female	7/19/1942	Rochelle Owen	One-time Customer	77	
940	San Jose	CA	95133	Female	3/7/1943	Corinne Finch	One-time Customer	76	
1119	San Jose	CA	95133	Female	9/3/1990	Twila Massey	One-time Customer	29	



# Calculated Column for Customer Age Group

## 1. Calculate customer age from birthdate

1 Age = Floor((TODAY()-Customers[BirthDate])/365, 1)									
Customerid	City	State	Zipcode	Gender	BirthDate	Customer	Customer Type	Age	
760	San Jose	CA	95133	Female	3/16/1968	Lucile Blake	One-time Customer	51	
881	San Jose	CA	95133	Female	7/19/1942	Rochelle Owen	One-time Customer	77	
940	San Jose	CA	95133	Female	3/7/1943	Corinne Finch	One-time Customer	76	
1119	San Jose	CA	95133	Female	9/3/1990	Twila Massey	One-time Customer	29	

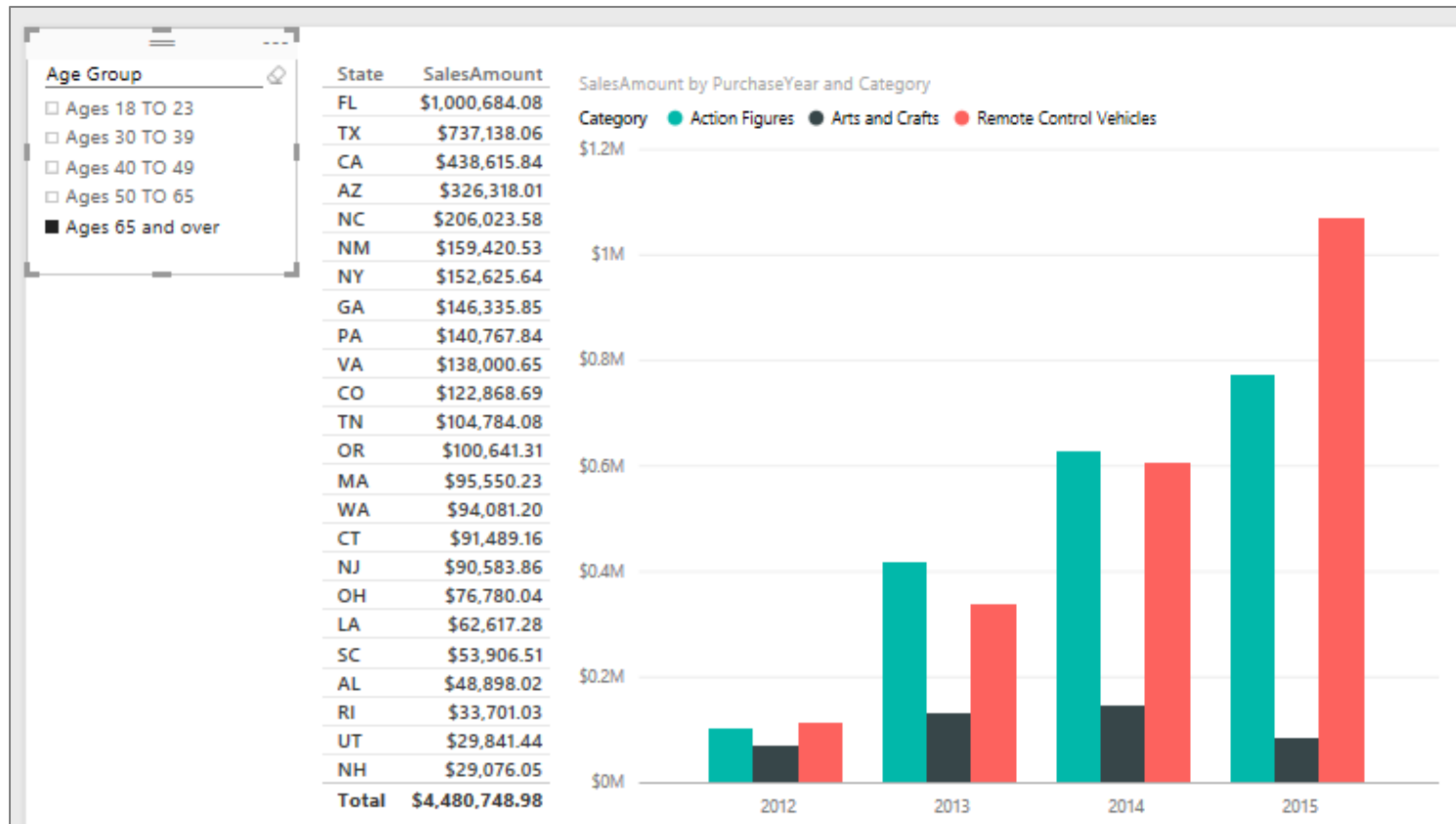
## 2. Calculate age groups using calculated column

1 Age Group = 2 SWITCH( 3 TRUE(), 4 [Age] >= 65, "65 and over", 5 [Age] >= 50, "50 to 64", 6 [Age] >= 40, "40 to 49", 7 [Age] >= 30, "30 to 39", 8 [Age] >= 18, "18 to 29", 9 [Age] < 18, "Under 18" 10 )										
Customerid	City	State	Zipcode	Gender	BirthDate	Customer	Customer Type	Age	Age Group	
760	San Jose	CA	95133	Female	3/16/1968	Lucile Blake	One-time Customer	51	50 to 64	
881	San Jose	CA	95133	Female	7/19/1942	Rochelle Owen	One-time Customer	77	65 and over	
940	San Jose	CA	95133	Female	3/7/1943	Corinne Finch	One-time Customer	76	65 and over	
1119	San Jose	CA	95133	Female	9/3/1990	Twila Massey	One-time Customer	29	18 to 29	



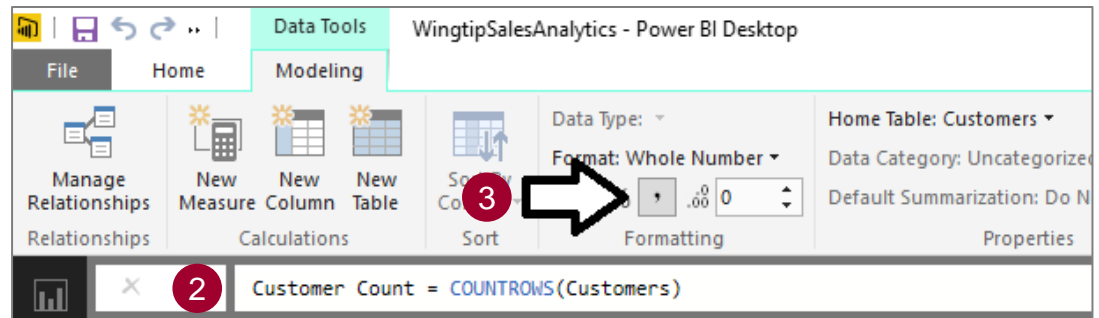
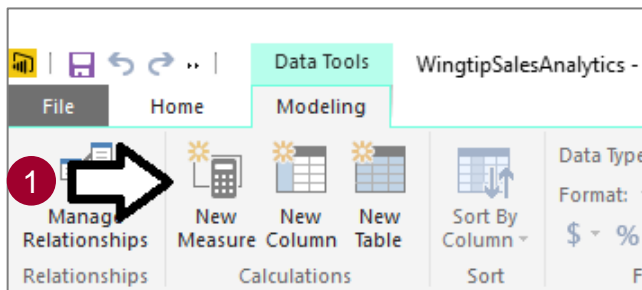
# Calculated Column used in a Slicer

- Calculated column can populate slicer values



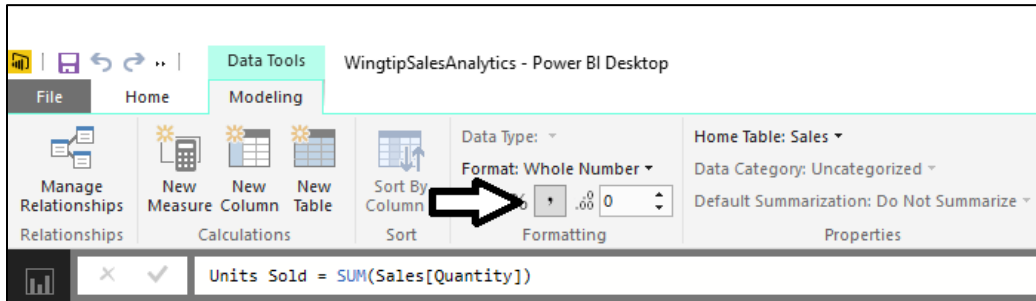
# Creating Measures

- Measures have advantage over calculated columns
  - They are evaluated based on the current evaluation context
- Creating a measure with Power BI Desktop
  1. Click New Measure button
  2. Give measure a name and write DAX expressions
  3. Configure formatting

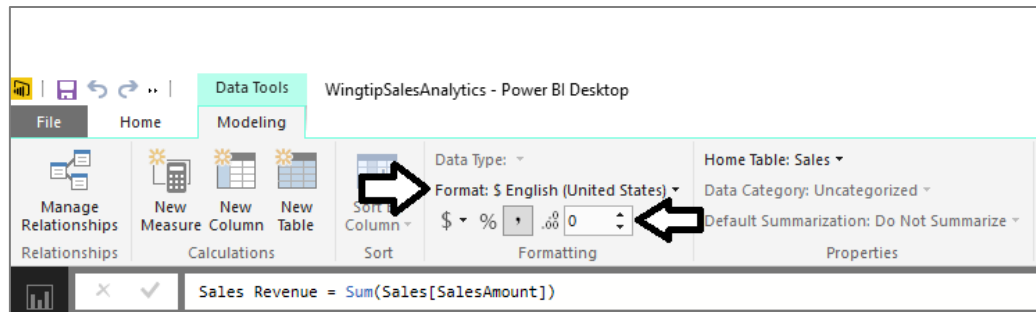


# Formatting Measures

- Format as whole number



- Format as currency

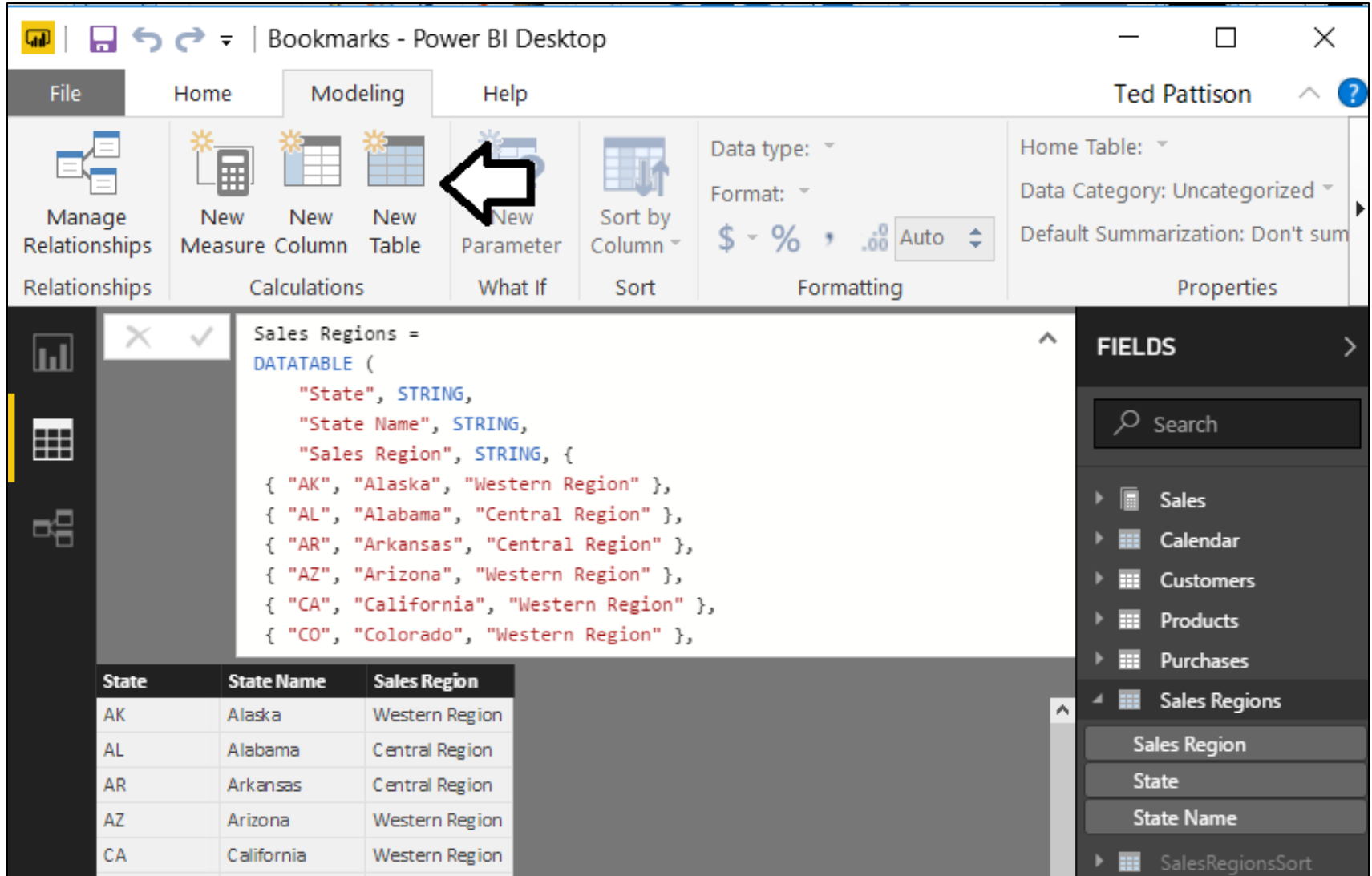


# Agenda

- ✓ Creating Table Relationships
- ✓ Creating Calculated Columns and Measure
- Creating Tables using DAX Expressions
- Configuring Fields for Geographic Mapping



# Creating Tables Dynamically using DAX



The screenshot shows the Power BI Desktop interface. The 'Modeling' ribbon is active, and the 'New Table' button is highlighted with a large black arrow. The 'Sales Regions' table is being created using the following DAX formula:

```
Sales Regions =  
DATATABLE (  
    "State", STRING,  
    "State Name", STRING,  
    "Sales Region", STRING, {  
        { "AK", "Alaska", "Western Region" },  
        { "AL", "Alabama", "Central Region" },  
        { "AR", "Arkansas", "Central Region" },  
        { "AZ", "Arizona", "Western Region" },  
        { "CA", "California", "Western Region" },  
        { "CO", "Colorado", "Western Region" },  
    }  
)
```

The resulting table is displayed below the formula:

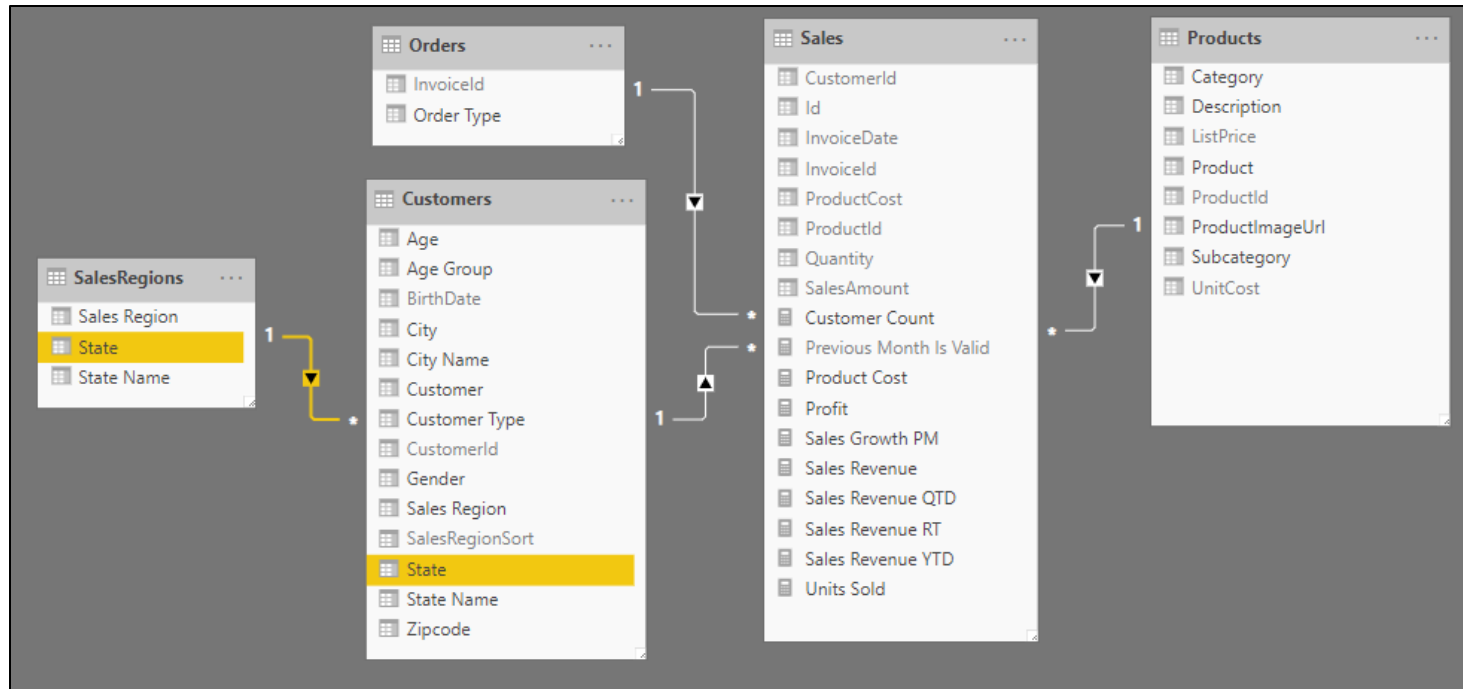
State	State Name	Sales Region
AK	Alaska	Western Region
AL	Alabama	Central Region
AR	Arkansas	Central Region
AZ	Arizona	Western Region
CA	California	Western Region

The 'FIELDS' pane on the right shows the 'Sales Regions' table selected, with its columns 'Sales Region', 'State', and 'State Name' listed.



# Integrating the Lookup Table into the Data Model

- Lookup table must be integrated into data model
  - Accomplished by creating relationship to one or more tables



# The RELATED Function

- RELATED function performs cross-table lookup
  - Effectively replaces older VLOOKUP function
  - Used in many-side table to look up value from one-side
  - Used to pull data from lookup table into primary table

1 Sales Region = RELATED(SalesRegions[Sales Region])											
CustomerId	City	State	Zipcode	Gender	BirthDate	Customer	Customer Type	Age	Age Group	Sales Region	
760	San Jose	CA	95133	Female	3/16/1968	Lucile Blake	One-time Customer	51	50 to 64	Western Region	
881	San Jose	CA	95133	Female	7/19/1942	Rochelle Owen	One-time Customer	77	65 and over	Western Region	
949	San Jose	CA	95133	Female	3/7/1943	Corinne Finch	One-time Customer	76	65 and over	Western Region	

1 State Name = RELATED(SalesRegions[State Name])										
Zipcode	Gender	BirthDate	Customer	Customer Type	Age	Age Group	Sales Region	State Name		
95133	Female	3/16/1968	Lucile Blake	One-time Customer	51	50 to 64	Western Region	California		
95133	Female	7/19/1942	Rochelle Owen	One-time Customer	77	65 and over	Western Region	California		
95133	Female	3/7/1943	Corinne Finch	One-time Customer	76	65 and over	Western Region	California		



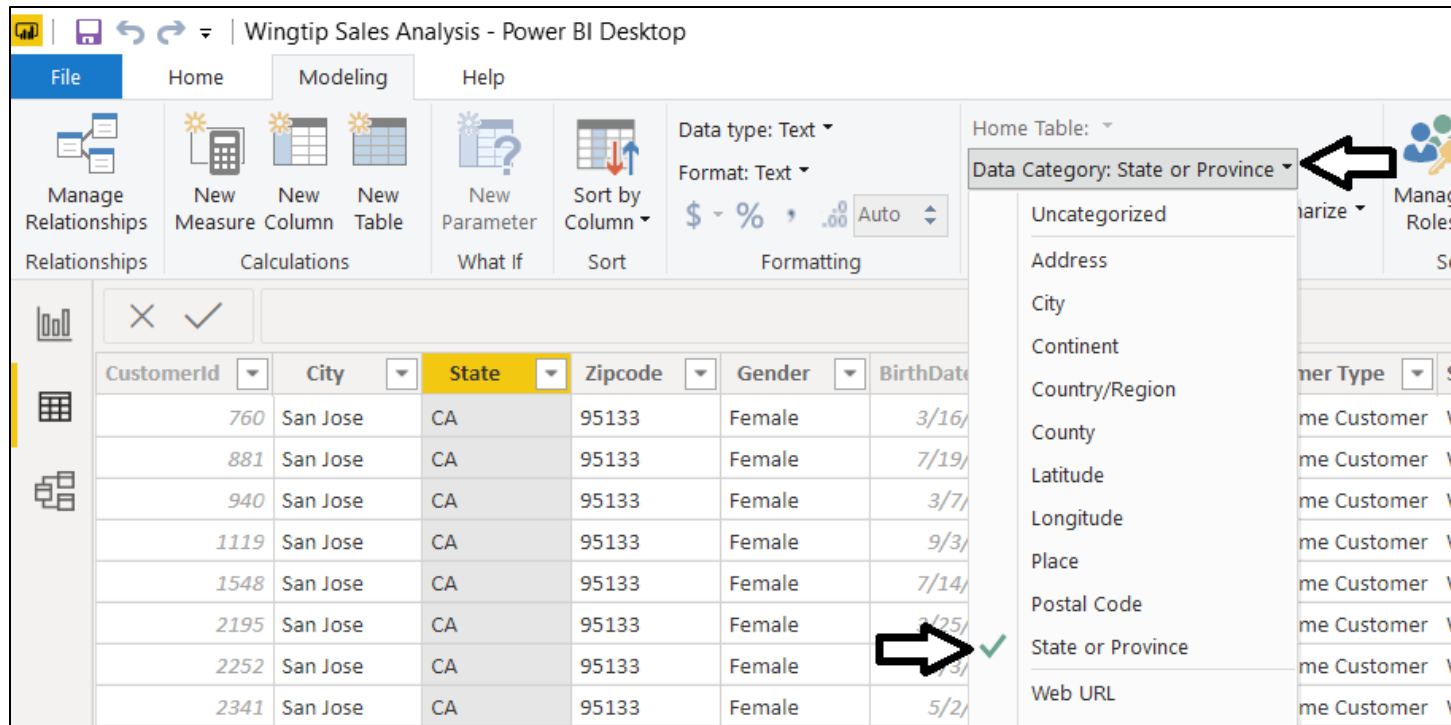
# Agenda

- ✓ Creating Table Relationships
- ✓ Creating Calculated Columns and Measure
- ✓ Creating Tables using DAX Expressions
- Configuring Fields for Geographic Mapping



# Geographic Field Metadata

- Fields in data model have metadata properties
  - Metadata used by visuals and reporting tools
  - Used as hints to Bing Mapping service



# Eliminate Geographic Ambiguity

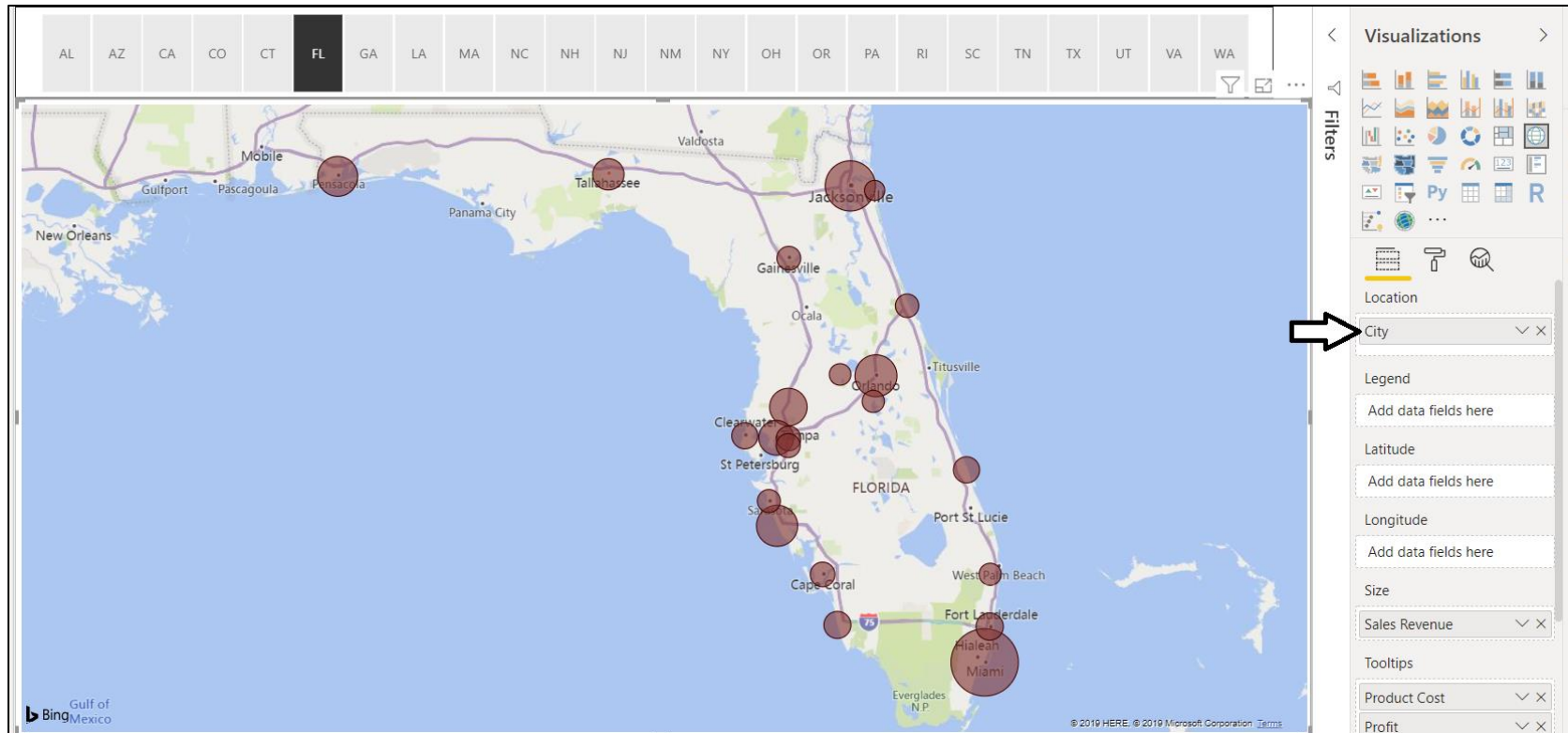
- City name alone is ambiguous
  - "Athens" defaults to Greece not Georgia
  - Concatenate city name with state to disambiguate

1 City = [City Name] & ", " & [State]								
BirthDate	Customer	Customer Type	Age	Age Group	Sales Region	State Name	SalesRegionSort	City
3/16/1968	Lucile Blake	One-time Customer	51	50 to 64	Western Region	California	1	San Jose, CA
7/19/1942	Rochelle Owen	One-time Customer	77	65 and over	Western Region	California	1	San Jose, CA
3/7/1943	Corinne Finch	One-time Customer	76	65 and over	Western Region	California	1	San Jose, CA
9/3/1990	Twila Massey	One-time Customer	29	18 to 29	Western Region	California	1	San Jose, CA
7/14/1955	Kellie Yang	One-time Customer	64	50 to 64	Western Region	California	1	San Jose, CA
3/25/1951	Megan Martin	One-time Customer	68	65 and over	Western Region	California	1	San Jose, CA



# Using Map Visual with a Geographic Field

- Map Visual shows distribution over geographic area
  - Visual automatically updates when filtered



# Summary

- ✓ Creating Table Relationships
- ✓ Creating Calculated Columns and Measure
- ✓ Creating Tables using DAX Expressions
- ✓ Configuring Fields for Geographic Mapping

