COMP1111

Week 9

Crosstab Queries

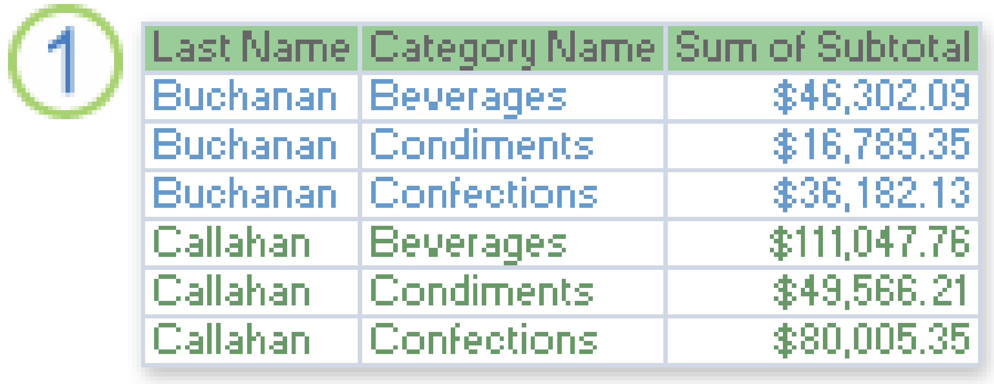
- To make summary data easier to read and understand, consider using a crosstab query.
- A crosstab query calculates a sum, average, or other aggregate function, and then groups the results by two sets of values—one set on the side of the datasheet and the other set across the top.
- On the ribbon, click **Create**, and then in the **Queries** group, click **Query Wizard**.
- In the **New Query** dialog box, double-click **Crosstab Query Wizard**.

Crosstab Queries

- A crosstab query is a type of select query used to arrange summary data – sums, averages, counts, and similar functions – into two category schemes, one of which is often time periods.
- For example, you might use a crosstab to show total sales by region per month.
- The way the results are arranged in a crosstab query can make it easier to read than a simple select query that displays the same data, as shown in the following illustration.

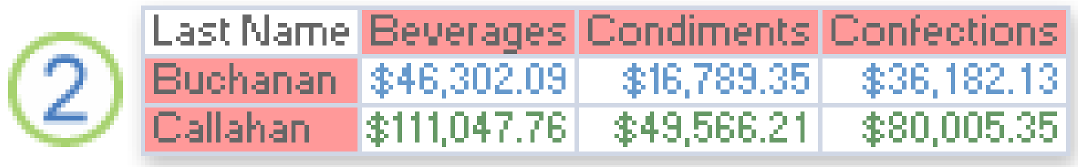
Crosstab Queries

- 1. This select query groups summary data vertically by employee and category.
- 2. This crosstab query displays the same data, grouped both horizontally and vertically.



1

Last Name	Category Name	Sum of Subtotal
Buchanan	Beverages	$46,302.09
Buchanan	Condiments	$16,789.35
Buchanan	Confections	$36,182.13
Callahan	Beverages	$111,047.76
Callahan	Condiments	$49,566.21
Callahan	Confections	$80,005.35



2

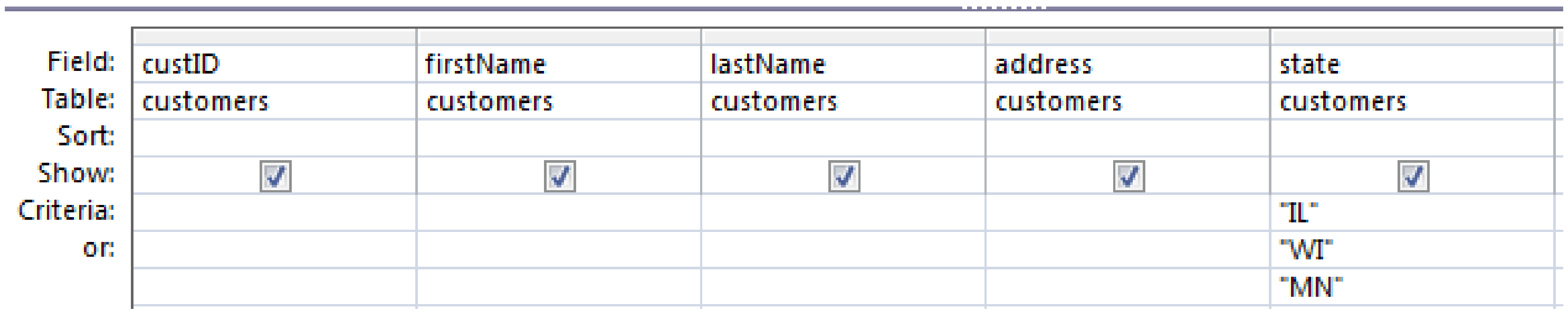
Last Name	Beverages	Condiments	Confections
Buchanan	$46,302.09	$16,789.35	$36,182.13
Callahan	$111,047.76	$49,566.21	$80,005.35

IN Clause

- The IN clause can be used in your WHERE clause to specify which data to include or exclude.
- You can use the IN clause by specifying the criteria to include in your query.

IN Clause

- For example, let's say you want a list of all orders in IL, WI and MN. You could do it this way:

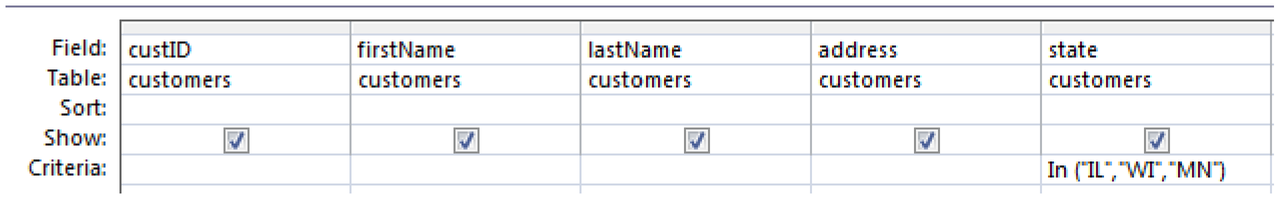


Field:	custID	firstName	lastName	address	state
Table:	customers	customers	customers	customers	customers
Sort:					
Show:	✓	✓	✓	✓	✓
Criteria:					"IL"
or:					"WI"
					"MN"

- The problem with the above statement is adding a ton of states makes for a long SQL Statement. Instead you can use the IN clause as a shortcut for the equivalent expression above:

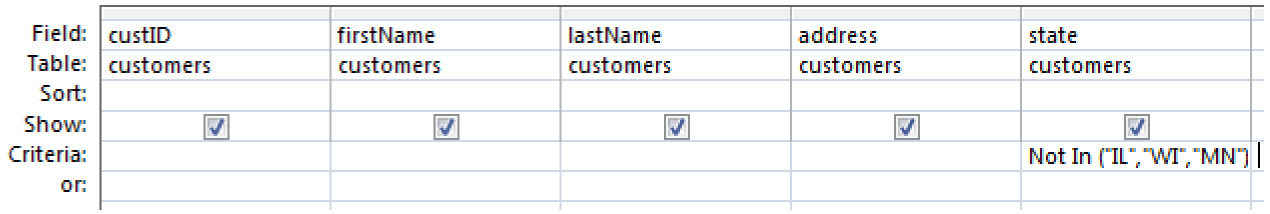
IN Clause

- Query using the IN clause



Field:	custID	firstName	lastName	address	state
Table:	customers	customers	customers	customers	customers
Sort:					
Show:	✓	✓	✓	✓	✓
Criteria:					In ("IL","WI","MN")

- Now, if you wanted customers that were NOT from those states, you can combine the IN clause with the NOT operator.



Field:	custID	firstName	lastName	address	state
Table:	customers	customers	customers	customers	customers
Sort:					
Show:	✓	✓	✓	✓	✓
Criteria:					Not In ("IL","WI","MN")
or:					

IIF Function

- You can use **IIF** anywhere you can use expressions.
- You use **IIF** to determine if another expression is true or false.
- If the expression is true, **IIF** returns one value; if it is false, **IIF** returns another.
- You specify the values **IIF** returns.
- **Syntax**
  - **IIF (** ***expr* , *truepart* , *falsepart*** **)**

IIF Function

- **Use IIF on a form or report**
- Suppose you have a Customers table that contains a field named CountryRegion.
- In a form, you want to denote whether Italian is the first language of the contact.
- You can add a control and use **IIF** in its **Control Source** property, like so:
- =IIF([CountryRegion]="Italy", "Italian", "Some other language")

IIF Function

- **Use IIF in complex expressions**
- You can use any expression as any part of an **IIF** statement.
- You can also "nest" **IIF** expressions, allowing you to evaluate a series of dependent expressions.
- To continue with the preceding example, you might want to test for several different CountryRegion values, and then display the appropriate language depending on which value exists:
- =IIF([CountryRegion]="Italy", "Italian", IIF([CountryRegion]="France", "French", IIF([CountryRegion]="Germany", "German", "Some other language")))

IIF Function

- The text "Some other language" is the ***falsepart*** argument of the innermost **IIF** function.
- Since each nested **IIF** function is the ***falsepart*** argument of the **IIF** function that contains it, the text "Some other language" is only returned if all the ***expr*** arguments of all the **IIF** functions evaluate to False.

IIF Function

- For another example, suppose you work at a library.
- The library database has a table named Check Outs that contains a field, named Due Date, that contains the date a particular book is due back.
- You can create a form that indicates the status of a checked out item in a control by using the **IIF** function in that control’s **Control Source** property, like so:
- =IIF([Due Date]<Date(),"OVERDUE",IIF([Due Date]=Date(),"Due today","Not Yet Due"))
- When you open the form in Form view, the control displays "OVERDUE" if the value of Due Date is less than the current date, "Due today" if it is equal to the current date, and "Not Yet Due" otherwise.

IIF Function

- **Use IIF in a query**
- The **IIF** function is frequently used to create calculated fields in queries.
- The syntax is the same, with the exception that in a query, you must preface the expression with a field alias and a colon (**:**) instead of an equal sign (**=**).
- To use the preceding example, you would type the following in the **Field** row of the query design grid:
- Language: IIF([CountryRegion]="Italy", "Italian", "Some other language")
- In this case, "Language:" is the field alias.

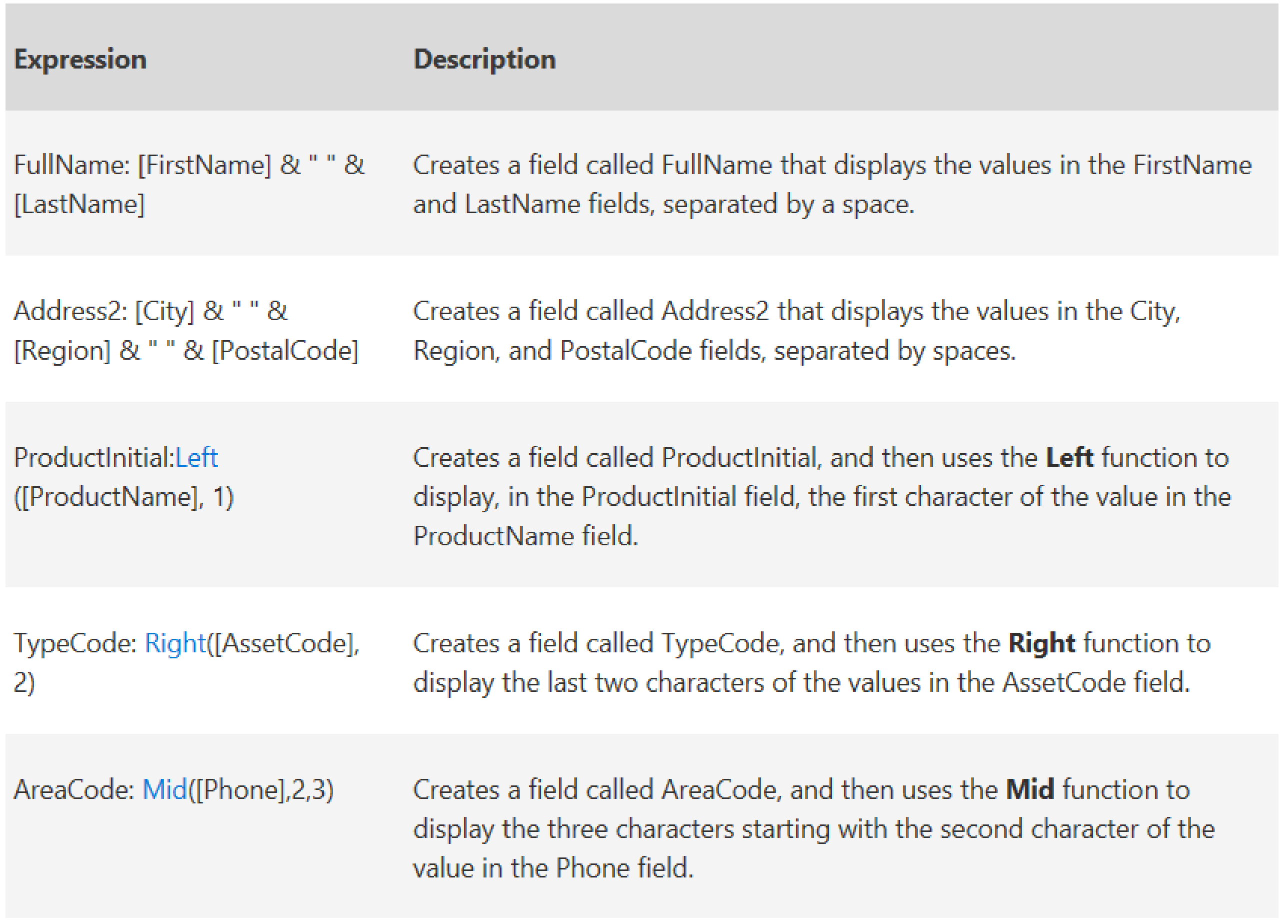
Expressions

- In Access, the term **expression** is synonymous with **formula**.
- An expression consists of a number of possible elements that you can use, alone or in combination, to produce a result.
- Those elements include:
- Identifiers — the names of table fields or controls on forms or reports, or the properties of those fields or controls
- Operators, such as **+** (plus) or **-** (minus)
- Functions, such as **SUM** or **AVG**
- Constants — values that do not change — such as strings of text, or numbers that are not calculated by an expression.
- You can use expressions in a number of ways — to perform a calculation, retrieve the value of a control, or supply criteria to a query, just to name a few.

Expressions

- **Expressions that manipulate text in a query or filter**
- The expressions in the following table use the **&** and **+** operators to combine text strings, use built-in functions to operate on a text string, or otherwise operate on text to create a calculated field.

Text Manipulation

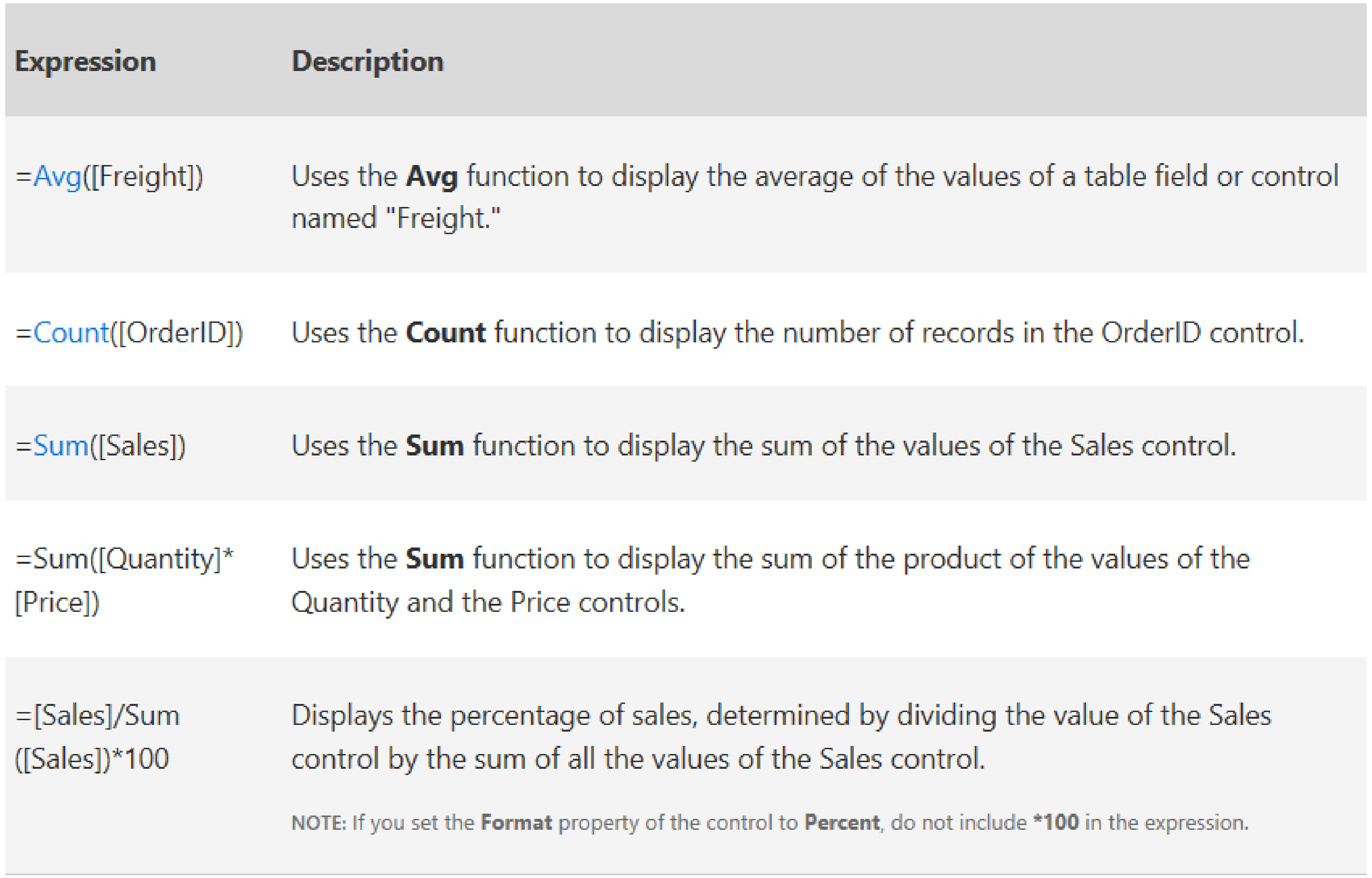


Expression	Description
FullName: [FirstName] & " " & [LastName]	Creates a field called FullName that displays the values in the FirstName and LastName fields, separated by a space.
Address2: [City] & " " & [Region] & " " & [PostalCode]	Creates a field called Address2 that displays the values in the City, Region, and PostalCode fields, separated by spaces.
ProductInitial:Left ([ProductName], 1)	Creates a field called ProductInitial, and then uses the **Left** function to display, in the ProductInitial field, the first character of the value in the ProductName field.
TypeCode: Right([AssetCode], 2)	Creates a field called TypeCode, and then uses the **Right** function to display the last two characters of the values in the AssetCode field.
AreaCode: Mid([Phone],2,3)	Creates a field called AreaCode, and then uses the **Mid** function to display the three characters starting with the second character of the value in the Phone field.

Expressions

- **Expressions that count, sum, and average values**
- You can use a type of function called an aggregate function to calculate values for one or more fields or controls.
- For example, you can calculate a group total for the group footer in a report, or an order subtotal for line items on a form. You can also count the number of items in one or more fields or calculate an average value.

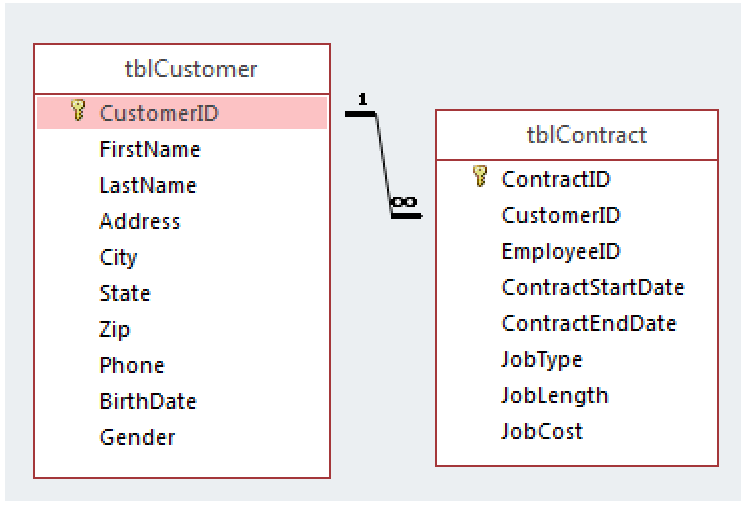
Aggregate Functions



Expression	Description
=Avg([Freight])	Uses the **Avg** function to display the average of the values of a table field or control named "Freight."
=Count([OrderID])	Uses the **Count** function to display the number of records in the OrderID control.
=Sum([Sales])	Uses the **Sum** function to display the sum of the values of the Sales control.
=Sum([Quantity]*[Price])	Uses the **Sum** function to display the sum of the product of the values of the Quantity and the Price controls.
=[Sales]/Sum([Sales])*100	Displays the percentage of sales, determined by dividing the value of the Sales control by the sum of all the values of the Sales control. NOTE: If you set the **Format** property of the control to **Percent**, do not include ***100** in the expression.

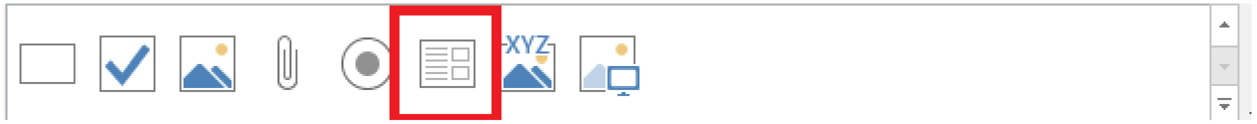
Subform Row Count

- Using the Form Wizard, create a form as the **mainform**.
- This form will use the **one (1)** side of your relationship.
- In the case of the example below, we will use **tblCustomer**



Subform Row Count

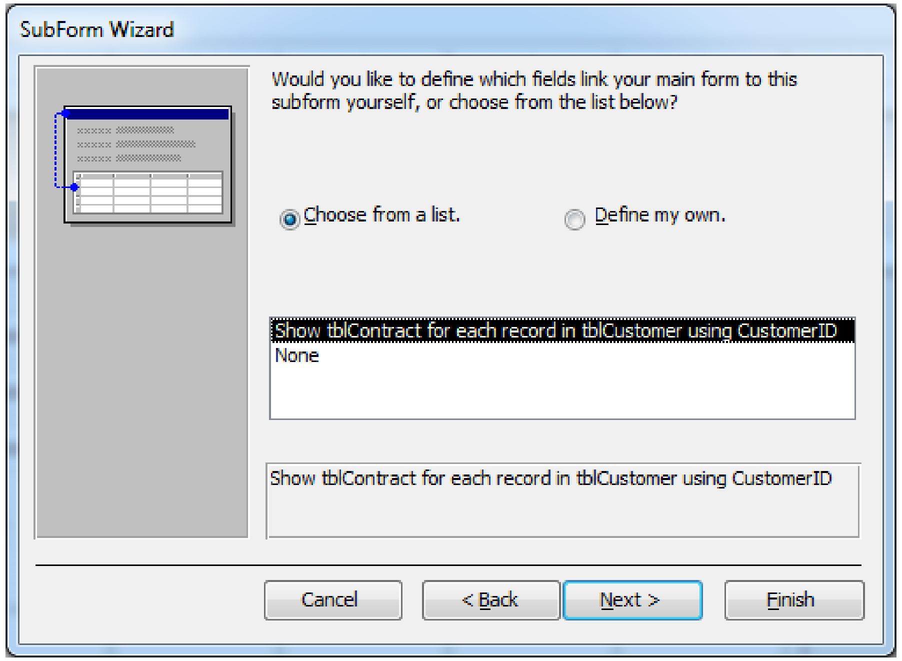
- Add the fields you think you will require
- Good examples are customerID, firstName, and lastName
- Next we will add a subform to our mainForm



- The subForm Wizard will appear.
- Select the fields you would like from the **many** table, in this case **tblContracts**
- Some fields to use here would be contractID, contractStartDate, contractEndDate, etc

Subform Row Count

- One the next screen, make sure the relationship is set correctly.
- For our example, show all contracts for each record in tblCustomer


SubForm Wizard

Would you like to define which fields link your main form to this subform yourself, or choose from the list below?

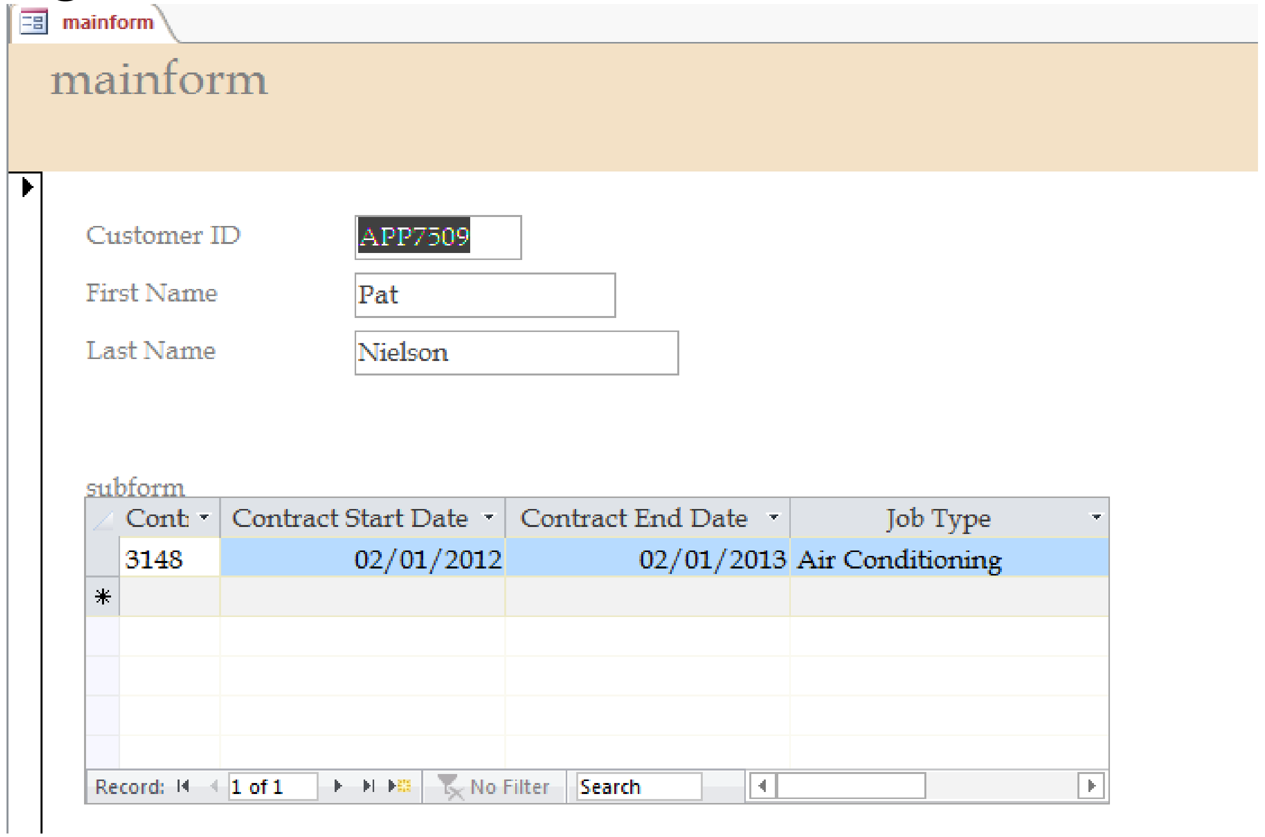
[✓ Choose from a list.] Define my own.

Show tblContract for each record in tblCustomer using CustomerID
None

Show tblContract for each record in tblCustomer using CustomerID

Subform Row Count

- At this point you should have a mainForm and subForm that looks something like this.



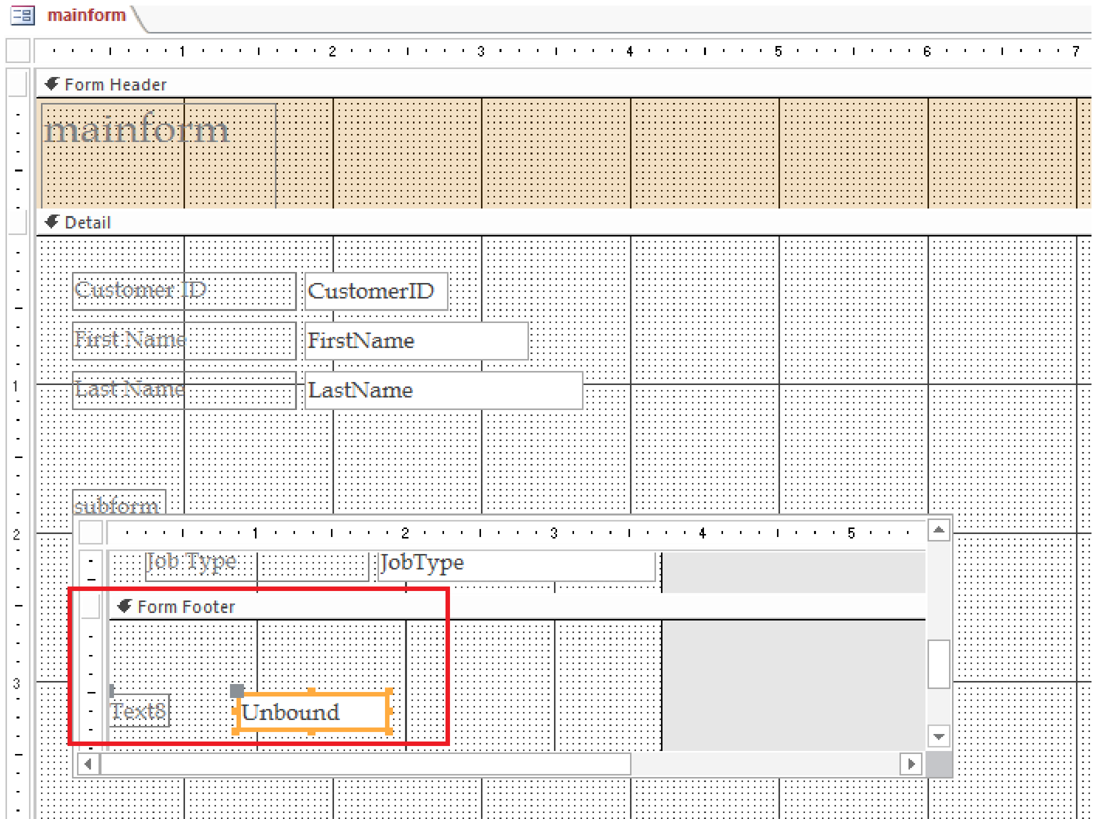
Customer ID	APP7509
First Name	Pat
Last Name	Nielson

subform

Cont	Contract Start Date	Contract End Date	Job Type
3148	02/01/2012	02/01/2013	Air Conditioning

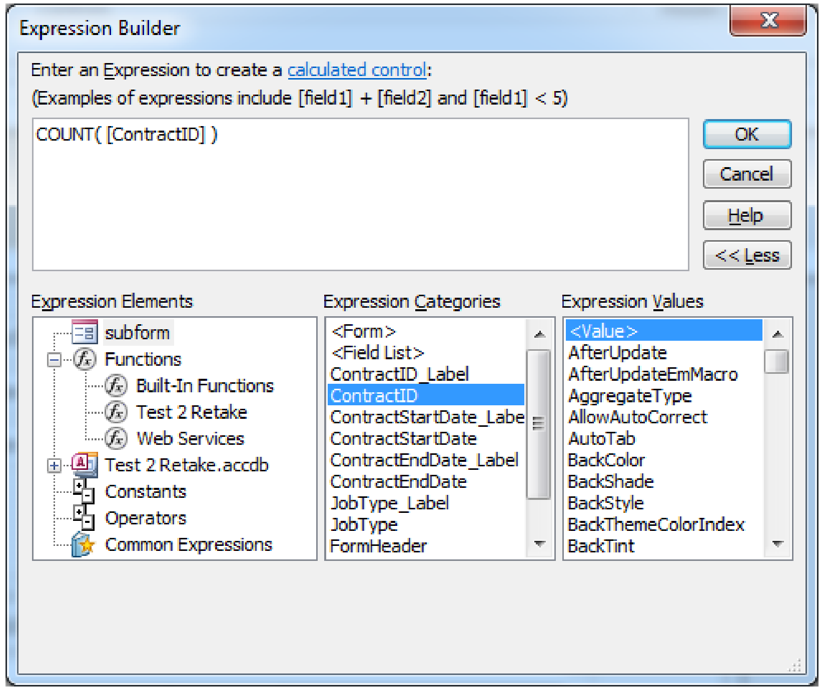
Subform Row Count

- Now, for the “tricky” part.
- From Design view, we are going to add a textbox in the **footer** of our **subform**



Subform Row Count

- Set your control source on your textbox
- **Property Sheet, Data Tab, Control Source**
- Set the expression, we will use the **COUNT** function to return the number of rows.

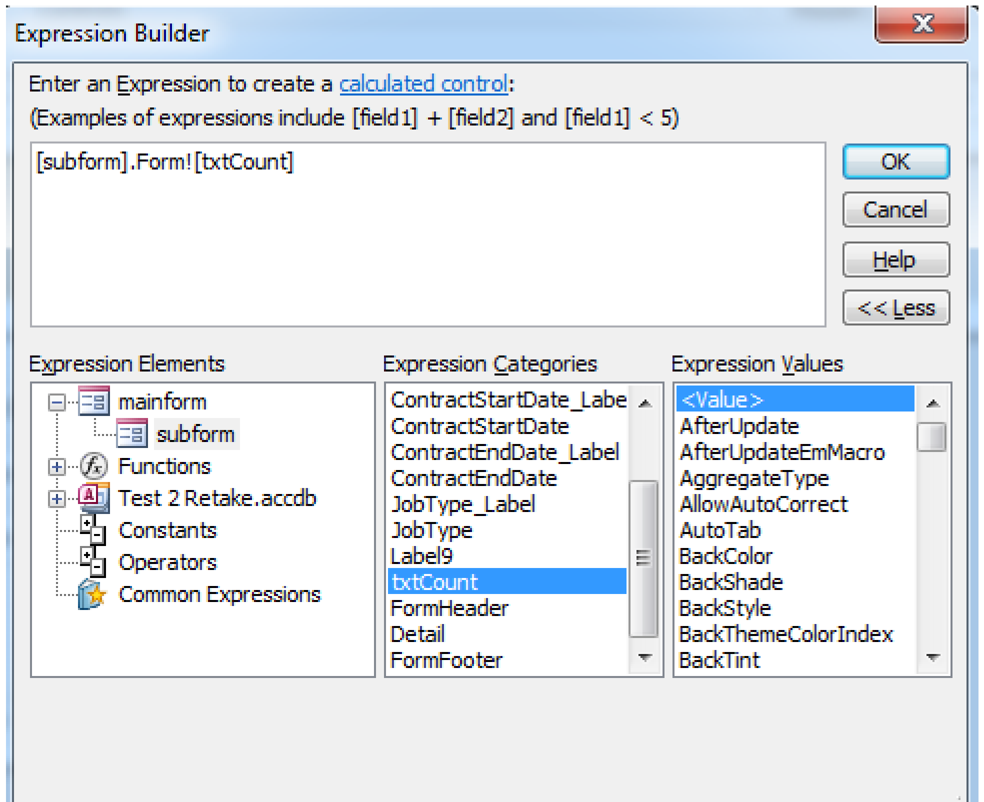

Expression Builder
Enter an Expression to create a calculated control:
(Examples of expressions include [field1] + [field2] and [field1] < 5)
OK
Cancel
Help
<< Less
Expression Elements
Expression Categories
Expression Values

COUNT([ContractID])

Expression Elements	Expression Categories	Expression Values
subform	<Form>	<Value>
Functions	<Field List>	AfterUpdate
Built-In Functions	ContractID_Label	AfterUpdateEmMacro
Test 2 Retake	ContractID	AggregateType
Web Services	ContractStartDate_Labe	AllowAutoCorrect
Test 2 Retake.accdb	ContractStartDate	AutoTab
Constants	ContractEndDate_Label	BackColor
Operators	ContractEndDate	BackShade
Common Expressions	JobType_Label	BackStyle
	JobType	BackThemeColorIndex
	FormHeader	BackTint

Subform Row Count

- Now, add a textbox somewhere on your mainForm.
- This textbox will link back to the value from the textbox in the subForm footer.
- Set the Control Source



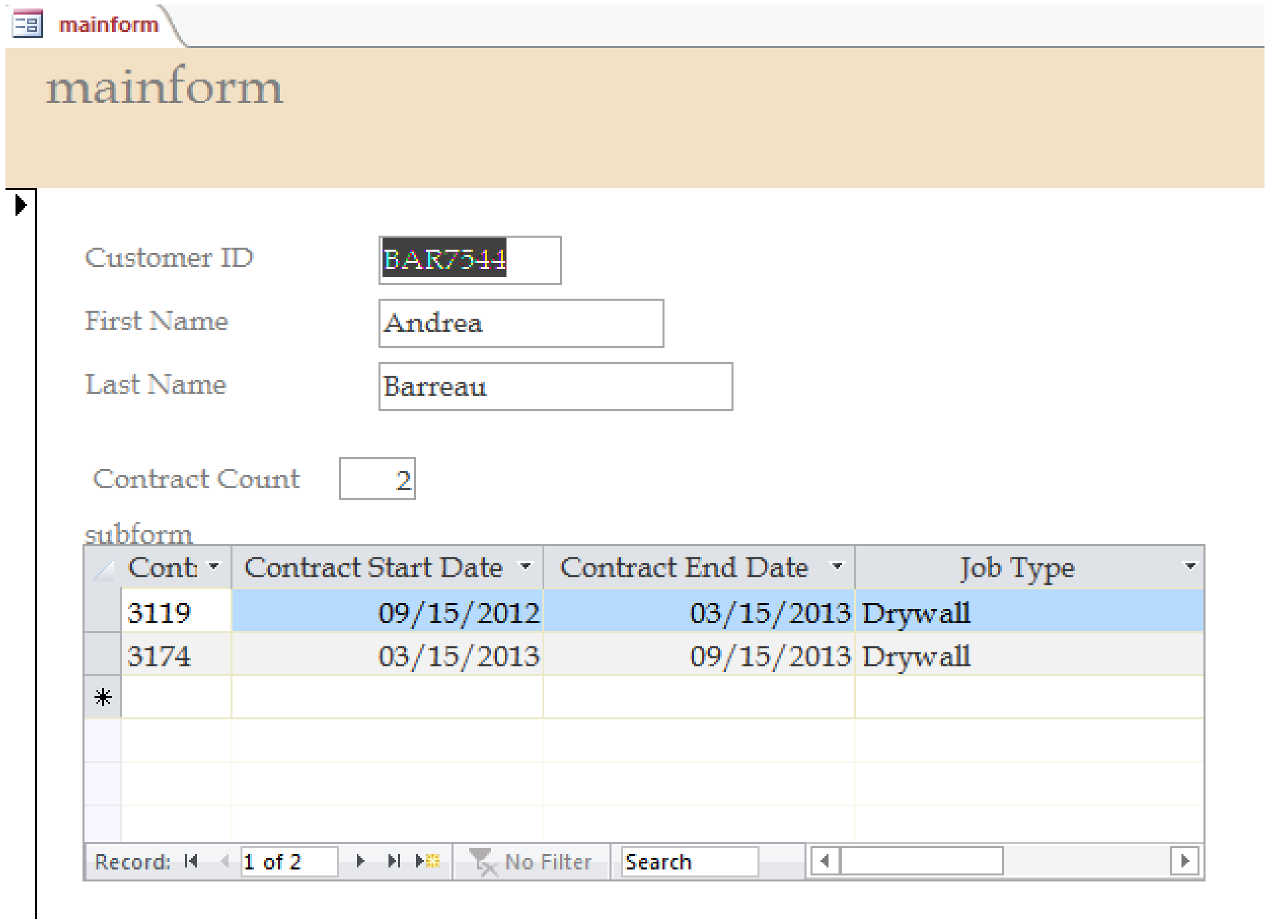
Expression Builder

Enter an Expression to create a calculated control:
(Examples of expressions include [field1] + [field2] and [field1] < 5)

```
[subform].Form![txtCount]
```

Expression Elements	Expression Categories	Expression Values
mainform	ContractStartDate_Labe	<Value>
subform	ContractStartDate	AfterUpdate
Functions	ContractEndDate_Label	AfterUpdateEmMacro
Test 2 Retake.accdb	ContractEndDate	AggregateType
Constants	JobType_Label	AllowAutoCorrect
Operators	JobType	AutoTab
Common Expressions	Label9	BackColor
	txtCount	BackShade
	FormHeader	BackStyle
	Detail	BackThemeColorIndex
	FormFooter	BackTint

Subform Row Count



mainform

mainform

Customer ID	BAR7544
First Name	Andrea
Last Name	Barreau
Contract Count	2

subform

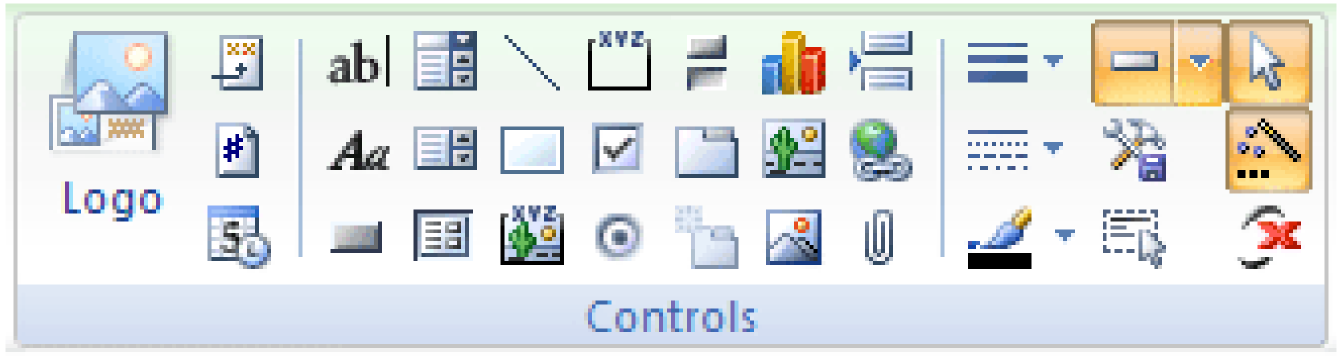
Cont	Contract Start Date	Contract End Date	Job Type
3119	09/15/2012	03/15/2013	Drywall
3174	03/15/2013	09/15/2013	Drywall
*			

Expressions

- Full list of expressions and how they work here:
- https://support.office.com/en-us/article/Examples-of-expressions-D3901E11-C04E-4649-B40B-8B6EC5AED41F

Calculated Controls

- This procedure helps you create a calculated control without using a control wizard.
- Right-click the form or report in the Navigation Pane, and then click **Design View**
- On the **Design** tab, in the **Controls** group, click the tool for the type of control you want to create.



Calculated Controls

- Position the pointer where you want the control to be placed on the form or report, and then click on the form or report to insert the control.
- If a control wizard starts, click **Cancel** to close it.
- Select the control, press F4 to display the property sheet, and then type an expression in the **Control Source** property box. To use the Expression Builder to create the expression, click … next to the **Control Source** property box.
- Switch to Form view or Report view and verify that the calculated control works as you expect.

Calculated Controls

- Precede each expression with the **=** operator. For example: **=[UnitPrice]*.75**.
- If you need more room to type an expression in the **Control Source** property box, press SHIFT+F2 to open the **Zoom** box.
- If your form or report is based on a query, you might want to put the expression in the query instead of in a calculated control. Doing this can improve performance and, if you are going to calculate totals for groups of records, it is easier to use the name of a calculated field in an aggregate function.
- When you sort on a calculated control in a form or report, ensure that the **Format** property of the control is set appropriately. Otherwise, calculated numeric or date values might sort alphabetically instead of numerically.

Calculated Controls

- If the control does not display the data you want (for example, if Access displays **#Name?** in the control), check the record source of the form or report to ensure that all the fields you used in the expression are available.
- If the record source is a query, you might need to add one or more fields to the query before the expression will work.

Calculated Controls

- Text boxes are the most popular choice for a calculated control because they can display so many different types of data.
- However, any control that has a **Control Source** property can be used as a calculated control.
- In many cases, it doesn't make sense to use a certain control type as a calculated control, because you can't update that control the way you can update a bound or unbound control.
- For example, if you place a check box control on a form and then enter an expression in the **Control Source** property of the check box, you can no longer select or clear the check box by clicking it.
- The check box appears selected or cleared, based on the results of the expression
- On a report, however, it may be useful to base a check box control on the results of a calculation, because controls on reports are used only to display information.

GCFLearnFree

- Information from http://www.gcflearnfree.org/access2016 and other web resources